Ran Canetti
Juan A. Garay (Eds.)

LNCS 8042

Advances in Cryptology – CRYPTO 2013

33rd Annual Cryptology Conference
Santa Barbara, CA, USA, August 2013
Proceedings, Part I

Springer

Lecture Notes in Computer Science 8042

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Ran Canetti Juan A. Garay (Eds.)

Advances in Cryptology – CRYPTO 2013

33rd Annual Cryptology Conference
Santa Barbara, CA, USA, August 18-22, 2013
Proceedings, Part I

Springer

Volume Editors

Ran Canetti
Boston University and Tel Aviv University
111 Cummington Street
Boston, MA 02215, USA
E-mail: canetti@bu.edu

Juan A. Garay
AT&T Labs – Research
180 Park Avenue
Florham Park, NJ 07932, USA
E-mail: garay@research.att.com

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-40040-7 e-ISBN 978-3-642-40041-4
DOI 10.1007/978-3-642-40041-4
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2013944216

CR Subject Classification (1998): E.3, F.2, K.6.5, G.2.1, D.4.6, C.2.0, J.1

LNCS Sublibrary: SL 4 – Security and Cryptology

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

CRYPTO 2013, the 33rd Annual International Cryptology Conference, was held August 18–22, 2013, on the campus of the University of California, Santa Barbara. The event was sponsored by the International Association for Cryptologic Research (IACR) in cooperation with the UCSB Computer Science Department and the IEEE Computer Society's Technical Committee on Security and Privacy.

The program represents the recent significant advances in all areas of cryptology. Sixty-one papers were included in the program, a record number for IACR flagship conferences. This two-volume proceedings contains the revised versions of all the papers. One pair of papers shared a single presentation slot in the program. There were also two invited talks. On Monday, Cindy Cohn from the Electronic Frontier Foundation gave a talk entitled "Crypto Wars Part 2 Have Begun." On Wednesday, Adam Langley from Google spoke about "Why the Web Still Runs on RC4," in a joint session with CHES 2013. To accommodate the increase in the number of papers, sessions were held throughout Tuesday and Thursday afternoons. The rump session took place as usual on Tuesday evening, and was chaired by Dan Bernstein and Tanja Lange.

For the Best Paper Award, the Program Committee (PC) unanimously selected the paper "On the Function Field Sieve and the Impact of Higher Splitting Probabilities" by Faruk Gologlu, Robert Granger, Gary McGuire and Jens Zumbragel.

This year we also awarded a *Best Young-Author Paper Award.* To be eligible for the award, all authors of the paper had to either be full-time students or have received their PhDs in 2011 or later. The award was given to the paper "Counter-Cryptanalysis: Reconstructing Flame's New Variant Collision Attack" by Marc Stevens.

Faced with a large number of high-quality submissions, the PC decided to significantly increase the number of papers in the program from last year's 48 papers, at the price of making the program longer and keeping the paper presentations short (20 minutes per paper, including questions and answers). Another option that was seriously considered was to move to parallel sessions on some of the days of the conference. This would have allowed for somewhat longer paper presentations, and an early adjourn on Thursday. In the end, we opted to retain the single-session format, with the hope of keeping the community more unified by allowing participants to attend all talks.

The papers were reviewed by a PC consisting of 40 leading researchers in the field, in addition to the two co-chairs. Each PC member was allowed to submit one paper, plus an additional one if co-authored with a student. PC-authored papers were held to higher standards during the review process. Papers were reviewed in a double-blind fashion. Initially, each paper was assigned to three reviewers (four for PC-authored papers). During the discussion phase, when

necessary, extra reviews were solicited. As part of the paper discussion phase, we held a two-day PC meeting on May 2 and 3, at the AT&T building in downtown Manhattan.

We strived to ensure that all papers received a fair and objective evaluation by experts as well as a broader group of PC members. The final decisions were made based on the reviews and discussion, and taking other factors such as balance of the program into account.

This year we initiated an early review and rebuttal process, where authors received preliminary reviews on their submissions about midway through the review period, and were given the option to comment on the reviews within a window of several days. The authors' comments were then taken into account in the discussions within the PC and in the final reviews. This process was labor-intensive; however, we feel it was worthwhile, as it resulted in a significantly better understanding of many submissions.

We would like to sincerely thank the authors of all submissions—those whose papers made it into the program and those whose papers did not. Our sincere gratitude also goes out to the PC members, who have invested an incredible amount of work in reviewing papers, interacting with the authors via the rebuttal mechanism, and participating in so many discussions on papers, their contribution, and the state of the art in their fields of expertise. We also sympathize with the occasional frustration from seeing decisions go against personal recommendations and preferences, in spite of the hard work invested.

We are also indebted to the many external reviewers, who significantly contributed to the comprehensive evaluation of papers. A list of PC members and external reviewers appears after this note. Despite all our efforts, the list of external reviewers may have errors or omissions; we apologize for that in advance.

We would like to thank Helena Handschuh, the General Chair, for working closely with us throughout the whole process, providing the much-needed support in every step, including creating and maintaining the website, and taking care of all aspects of the conference's logistics.

Special thanks are due to Shai Halevi, who provided us with unlimited support of his *websubrev* software, which we used for the whole conference planning, paper evaluation, and interaction with PC members and authors. Josh Benaloh, was our IACR point of contact, always providing timely and informative answers to our questions. Alfred Hofmann and his colleagues at Springer provided a meticulous service for the timely production of this volume.

Finally, we would like to thank Qualcomm, Microsoft, Google, Good Technologies, and Cryptography Research Inc. for their generous support.

August 2013

Ran Canetti
Juan A. Garay

Crypto 2013
The 33rd International Cryptology Conference

Sponsored by *the International Association for Cryptologic Research*

General Chair

Helena Handschuh	Cryptography Research Inc. and K.U. Leuven

Program Co-chairs

Ran Canetti	Boston University and Tel Aviv University
Juan A. Garay	AT&T Labs — Research

Program Committee

Masayuki Abe	NTT, Japan
Mihir Bellare	UCSD, USA
Zvika Brakerski	Stanford University, USA
Jan Camenisch	IBM Research, Zürich, Switzerland
David Cash	Rutgers University, USA
Kai-Min Chung	Cornell University, USA and Academia Sinica, Taiwan
Jean-Sebastien Coron	University of Luxembourg
Dana Dachman-Soled	Microsoft Research, USA
Stefan Dziembowski	University of Warsaw, Poland and University of Rome I, Italy
Iftach Haitner	Tel Aviv University, Israel
Shai Halevi	IBM Research, USA
Goichiro Hanaoka	AIST, Japan
Dennis Hofheinz	KIT, Germany
Jonathan Katz	University of Maryland, USA
Lars R Knudsen	DTU, Denmark
Eyal Kushilevitz	Technion, Israel
Kristin Lauter	Microsoft Research, USA
Huijia Lin	MIT and Boston University, USA
Yehuda Lindell	Bar Ilan University, Israel
Vadim Lyubashevsky	ENS, France
John Mitchell	Stanford University, USA
Tal Moran	Inter-Disciplinary Center, Israel
Jesper B Nielsen	University of Aarhus, Denmark
Christof Paar	University of Bochum, Germany

Manoj M Prabhakaran	University of Illinois at Urbana-Champaign, USA
Tal Rabin	IBM Research, USA
Charles Rackoff	University of Toronto, Canada
Christian Rechberger	DTU, Denmark
Thomas Ristenpart	University of Wisconsin, USA
Guy Rothblum	Microsoft Research, USA
Rei Safavi	University of Calgary, Canada (advisory member)
Christian Schaffner	University of Amsterdam, The Netherlands
Hovav Shacham	UCSD, USA
Vitaly Shmatikov	UT Austin, USA
Nigel Smart	University of Bristol, UK
Adam Smith	Penn State University, USA
Martijn Stam	University of Bristol, UK
John P Steinberger	Tsinghua University, China
Frederik Vercauteren	K.U. Leuven, Belgium
Xiaoyun Wang	Tsinghua University, China
Daniel Wichs	Northeastern University, USA

External Reviewers

Divesh Aggarwal	Ignacio Cascudo	Eiichiro Fujisaki
Adi Akavia	Nishanth Chandran	Steven Galbraith
Martin Albrecht	Melissa Chase	Sanjam Garg
Elena Andreeva	Nathan Chenette	Ran Gelles
Benny Applebaum	Alessandro Chiesa	Rosario Gennaro
Gilad Asharov	Sherman S.M. Chow	Craig Gentry
Gilles van Assche	Craig Costello	Benedikt Gierlichs
Nuttapong Attrapadung	Scott Coull	Vipul Goyal
Paulo Barreto	Ivan Damgaard	Louis Granboulan
Timo Bartkewitz	Maria Dubovitskaya	Adam Groce
Raef Bassily	Leo Ducas	Jens Groth
Amos Beimel	Frédéric Dupuis	Kris Haralambiev
David Bernhard	Konrad Durnoga	Moritz Hardt
Dan Bernstein	Markus Drmuth	Carmit Hazay
Nir Bitansky	Keita Emura	Nadia Heninger
Andrey Bogdanov	Robert Enderlein	Jens Hermans
Joppe Bos	Sebastian Faust	Gottfried Herold
Christina Boura	Serge Fehr	Martin Hirt
Elette Boyle	Sean Hallgren	Viet Tung Hoang
Cas Cremers	Feng-Hao	Susan Hohenberger
Christophe De Cannire	Dario Fiore	Yuval Ishai
Anne Canteaut	Marc Fischlin	Tibor Jager
Angelo De Caro	Tore Kasper Frederiksen	Abhishek Jain

Thomas P. Jakobsen
Chen Jie
Charanjit Jutla
Seny Kamara
Tomasz Kazana
Aoki Kazumaro
Sriram Keelveedhi
Dmitry Khovratovich
Eike Kiltz
Ilya Kizhvatov
Markulf Kohlweiss
Venkata Koppula
Hugo Krawczyk
Stephan Krenn
Ranjit Kumaresan
Kaoru Kurosawa
Tanja Lange
Enrique Larraia
Martin M. Lauridsen
Gregor Leander
Chen-Kuei Lee
Anja Lehmann
Gaetan Leurent
Kevin Lewi
Allison Lewko
Feng-Hao Liu
Feng-hao Liu
Jake Loftus
Steve Lu
Edward Lui
Mohammad Mahmoody
Hemanta Maji
Takahiro Matsuda
Chrysanti Mavrotami
Travis Mayberry
Sarah Meiklejohn
Florian Mendel
Alexander Meurer
Daniele Micciancio
Eric Miles
Kazuhiko Minematsu
Ilya Mironov
Peter Montgomery
Amir Moradi
Paz Morillo
Kirill Morozov
Nicky Mouha
Naveed Muhammad
Michael Naehrig
Maria Naya-Plasencia
Gregory Neven
Phong Nguyen
Ryo Nishimaki
Kobbi Nissim
Adam O'Neill
Tatsuaki Okamoto
Claudio Orlandi
Rafi Ostrovsky
Omer Paneth
Bryan Parno
Anat Paskin-Cherniavsky
Christopher J. Peikert
Yuval Peres
Ludovic Perret
Eduoardo Persichetti
Joop van de Pol
Christopher Portmann
Emmanuel Prouff
Ananth Raghunathan
Kasper B. Rasmussen
Mariana Raykova
Oded Regev
Leonid Reyzin
Ben Riva
Matthieu Rivain
Phillip Rogaway
Mike Rosulek
Ron Rothblum
Yannis Rouselakis
Yusuke Sakai
Koichi Sakumoto
Louis Salvail
Palash Sarkar
Giannicola Scarpa
Dominique Schroeder
Peter Schwabe
Karn Seth
Ronen Shaltiel
Chih-Hao Shen
Thomas Shrimpton
Maciej Skórski
Graeme Smith
Fang Song
Damien Stehle
Ron Steinfeld
Koutarou Suzuki
Katsuyuki Takashima
Sidharth Telang
Aris Tentes
Isamu Teranishi
Seth Terashima
Stefano Tessaro
Susan Thomson
Mehdi Tibouch
Jean-Pierre Tillich
Tomas Toft
Eran Tromer
Max Tuengerthal
Madhur Tulsiani
Vinod Vaikuntanathan
Vesselin Velichkov
K. Venkata Muthuramakrishnan
Venkitasubramaniam
Thomas Vidick
Colin Walter
Meiqin Wang
Brent Waters
Dirk Westhoff
Carolyn Whitnall
Ronald de Wolf
David Wu
Xiaodi Wu
Keita Xagawa
Shota Yamada
Scott Yilek
Kazuki Yoneyama
Hongbo Yu
Greg Zaverucha
Maciej Zdanowicz
Mark Zhandry
Colin Jia Zheng
Joe Zimmerman
Angela Zottarel

Table of Contents – Part I

Session 4: Cryptanalysis II

Session 5: MPC – New Directions

Session 6: Leakage Resilience

Session 7: Symmetric Encryption and PRFs

Session 8: Key Exchange

Session 9: Multi Linear Maps

Session 10: Ideal Ciphers

Table of Contents – Part II

Session 11: Implementation-Oriented Protocols

Invited Talk: Why the Web Still Runs on RC4

Session 12: Number-Theoretic Hardness

Session 13: MPC — Foundations

Session 14: Codes and Secret Sharing

Session 15: Signatures and Authentication

Session 16: Quantum Security

Session 17: New Primitives

Session 18: Functional Encryption I

Session 19: Functional Encryption II

Practical Bootstrapping in Quasilinear Time*

Jacob Alperin-Sheriff and Chris Peikert

School of Computer Science, Georgia Institute of Technology

Abstract. Gentry's "bootstrapping" technique (STOC 2009) constructs a fully homomorphic encryption (FHE) scheme from a "somewhat homomorphic" one that is powerful enough to evaluate its own decryption function. To date, it remains the only known way of obtaining unbounded FHE. Unfortunately, bootstrapping is computationally very expensive, despite the great deal of effort that has been spent on improving its efficiency. The current state of the art, due to Gentry, Halevi, and Smart (PKC 2012), is able to bootstrap "packed" ciphertexts (which encrypt up to a linear number of bits) in time only *quasilinear* $\tilde{O}(\lambda) = \lambda \cdot \log^{O(1)} \lambda$ in the security parameter. While this performance is *asymptotically* optimal up to logarithmic factors, the practical import is less clear: the procedure composes multiple layers of expensive and complex operations, to the point where it appears very difficult to implement, and its concrete runtime appears worse than those of prior methods (all of which have quadratic or larger asymptotic runtimes).

In this work we give *simple*, *practical*, and entirely *algebraic* algorithms for bootstrapping in quasilinear time, for both "packed" and "non-packed" ciphertexts. Our methods are easy to implement (especially in the non-packed case), and we believe that they will be substantially more efficient in practice than all prior realizations of bootstrapping. One of our main techniques is a substantial enhancement of the "ring-switching" procedure of Gentry et al. (SCN 2012), which we extend to support switching between two rings where neither is a subring of the other. Using this procedure, we give a natural method for homomorphically evaluating a broad class of structured linear transformations, including one that lets us evaluate the decryption function efficiently.

1 Introduction

Bootstrapping, a central technique from the breakthrough work of Gentry [8, 7] on fully homomorphic encryption (FHE), converts a sufficiently powerful "somewhat homomorphic" encryption (SHE) scheme into a fully homomorphic one.

* This material is based upon work supported by the National Science Foundation under CAREER Award CCF-1054495, by the Alfred P. Sloan Foundation, and by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under Contract No. FA8750-11-C-0098. The views expressed are those of the authors and do not necessarily reflect the official policy or position of the National Science Foundation, the Sloan Foundation, DARPA or the U.S. Government.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 1–20, 2013.

(An SHE scheme can support a bounded number of homomorphic operations on freshly generated ciphertexts, whereas an FHE scheme has no such bound.) In short, bootstrapping works by *homomorphically* evaluating the SHE scheme's decryption function on a ciphertext that cannot support any further homomorphic operations. This has the effect of "refreshing" the ciphertext, i.e., it produces a new one that encrypts the same message and can handle more homomorphic operations. Bootstrapping remains the only known way to achieve *unbounded* FHE, i.e., a scheme that can homomorphically evaluate any efficient function using keys and ciphertexts of a fixed size.[1]

In order to be "bootstrappable," an SHE scheme must be powerful enough to homomorphically evaluate its own decryption function, using whatever homomorphic operations it supports. For security reasons, the key and ciphertext sizes of all known SHE schemes grow with the *depth* and, to a lesser extent, the *size* of the functions that they can homomorphically evaluate. For instance, under plausible hardness conjectures, the key and ciphertext sizes of the most efficient SHE scheme to date [3] grow quasilinearly in both the supported multiplicative depth d and the security parameter λ, i.e., as $\tilde{O}(d \cdot \lambda)$. Clearly, the runtime of bootstrapping must also grow with the sizes of the keys, ciphertexts, and decryption function. This runtime is perhaps the most important measure of efficiency for FHE, because bootstrapping is currently the biggest bottleneck by far in instantiations, both in theory and in practice.

The past few years have seen an intensive study of different forms of decryption procedures for SHE schemes, and their associated bootstrapping operations [8, 7, 18, 10, 4, 9, 3, 13]. The first few bootstrapping methods had moderate polynomial runtimes in the security parameter λ, e.g., $\tilde{O}(\lambda^4)$. Brakerski, Gentry, and Vaikuntanathan [3] gave a major efficiency improvement, reducing the runtime to $\tilde{O}(\lambda^2)$. They also gave an amortized method that bootstraps $\tilde{\Omega}(\lambda)$ ciphertexts at once in $\tilde{O}(\lambda^2)$ time, i.e., quasilinear runtime per ciphertext. However, these results apply only to "non-packed" ciphertexts, i.e., ones that encrypt essentially just one bit each, which combined with the somewhat large runtimes makes these methods too inefficient to be used very much in practice. Most recently, Gentry, Halevi, and Smart [12] achieved bootstrapping for "packed" ciphertexts (i.e., ones that encrypt up to $\tilde{\Omega}(\lambda)$ bits each) in *quasilinear* $\tilde{O}(\lambda)$ runtime, which is asymptotically optimal in space and time, up to polylogarithmic factors. For this they relied on a general "compiler" from another work of theirs [13], which achieved SHE/FHE for sufficiently wide circuits with polylogarithmic multiplicative "overhead," i.e., cost relative to evaluating the circuit "in the clear."

Bootstrapping and FHE in quasi-optimal time and space is a very attractive and powerful theoretical result. However, the authors of [13, 12] caution that their constructions may have limited potential for use in practice, for two main

[1] This stands in contrast with *leveled* FHE schemes, which can homomorphically evaluate a function of any *a priori* bounded depth, but using keys and ciphertexts whose sizes depend on the bound. Leveled FHE can be constructed without resorting to bootstrapping [3].

reasons: first, the runtimes, while asymptotically quasilinear, include very large polylogarithmic factors. For realistic values of the security parameter, these polylogarithmic terms exceed the rather small (but asymptotically worse) quasilinear overhead obtained in [3]. The second reason is that their bootstrapping operation is algorithmically very complex and difficult to implement (see the next paragraphs for details). Indeed, while there are now a few working implementations of bootstrapping (e.g., [10, 6]) that follow the templates from [8, 7, 18, 3], we are not aware of any attempt to implement any method having subquadratic runtime.

Is quasilinear efficient? The complexity and large practical overhead of the constructions in [13, 12] arise from two kinds of operations. First, the main technique from [13] is a way of homomorphically evaluating any sufficiently shallow and wide arithmetic circuit on a "packed" ciphertext that encrypts a high-dimensional vector of plaintexts in multiple "slots." It works by first using ring automorphisms and key-switching operations [4, 3] to obtain a small, fixed set of "primitive" homomorphic permutations on the slots. It then composes those permutations (along with other homomorphic operations) in a log-depth permutation network, to obtain any permutation. Finally, it homomorphically evaluates the desired circuit by combining appropriate permutations with relatively simple homomorphic slot-selection and ring operations.

In the context of bootstrapping, one of the key observations from [12] is that a main step of the decryption procedure can be evaluated using the above technique. Specifically, they need an operation that moves the coefficients of an encrypted plaintext polynomial, reduced modulo a cyclotomic polynomial $\Phi_m(X)$, into the slots of a packed ciphertext (and back again). Once the coefficients are in the slots, they can be rounded in a batched (SIMD) fashion, and then mapped back to coefficients of the plaintext. The operations that move the coefficients into slots and vice-versa can be expressed as $O(\log \lambda)$-depth arithmetic circuits of size $O(\lambda \log \lambda)$, roughly akin to the classic FFT butterfly network. Hence they can be evaluated homomorphically with polylogarithmic overhead, using [13]. However, as the authors of [12] point out, the decryption circuit is quite large and complex – especially the part that moves the slots back to the coefficients, because it involves reduction modulo $\Phi_m(X)$ for an m having several prime divisors. This modular reduction is the most expensive part of the decryption circuit, and avoiding it is one of the main open problems given in [12]. However, even a very efficient decryption circuit would still incur the large polylogarithmic overhead factors from the techniques of [13].

1.1 Our Contributions

We give a new bootstrapping algorithm that runs in *quasilinear* $\tilde{O}(\lambda)$ time per ciphertext with *small* polylogarithmic factors, and is algorithmically much simpler than previous methods. It is easy to implement, and we believe that it will be substantially more efficient in practice than all prior methods. We provide a unified bootstrapping procedure that works for both "non-packed" ciphertexts

(which encrypt integers modulo some p, e.g., bits) and "packed" ciphertexts (which encrypt elements of a high-dimensional ring), and also interpolates between the two cases to handle an intermediate concept we call "semi-packed" ciphertexts.

Our procedure for non-packed ciphertexts is especially simple and efficient. In particular, it can work very naturally using only cyclotomic rings having power-of-two index, i.e., rings of the form $\mathbb{Z}[X]/(1 + X^{2^k})$, which admit very fast implementations. This improves upon the method of [3], which achieves quasilinear *amortized* runtime when bootstrapping $\tilde{\Omega}(\lambda)$ non-packed ciphertexts at once. Also, while that method can also use power-of-two cyclotomics, it can only do so by emulating $\mathbb{Z}_2$ (bit) arithmetic within $\mathbb{Z}_p$ for some moderately large prime p, which translates additions in $\mathbb{Z}_2$ into much more costly multiplications in $\mathbb{Z}_p$. By contrast, our method works "natively" with any plaintext modulus.

For packed ciphertexts, our procedure draws upon high-level ideas from [13, 12], but our approach is conceptually and technically very different. Most importantly, it completely avoids the two main inefficiencies from those works: first, unlike [13], we do not use permutation networks or any explicit permutations of the plaintext slots, nor do we rely on a general-purpose compiler for homomorphically evaluating arithmetic circuits. Instead, we give direct, practically efficient procedures for homomorphically mapping the coefficients of an encrypted plaintext element into slots and vice-versa. In particular, our procedure does not incur the large cost or algorithmic complexity of homomorphically reducing modulo $\Phi_m(X)$, which was the main bottleneck in the decryption circuit of [12].

At a higher level, our bootstrapping method has two other attractive and novel features: first, it is entirely "algebraic," by which we mean that the full procedure (including generation of all auxiliary data it uses) can be described as a short sequence of elementary operations from the "native instruction set" of the SHE scheme. By contrast, all previous methods at some point invoke rather generic arithmetic circuits, e.g., for modular addition of values represented as bit strings, or reduction modulo a cyclotomic polynomial $\Phi_m(X)$. Of course, arithmetic circuits can be evaluated using the SHE scheme's native operations, but we believe that the distinction between "algebraic" and "non-algebraic" is an important qualitative one, and it certainly affects the simplicity and concrete efficiency of the bootstrapping procedure.

The second nice feature of our method is that it completely decouples the algebraic structure of the SHE plaintext ring from that which is needed by the bootstrapping procedure. In previous methods that use amortization (or "batching") for efficiency (e.g., [17, 3, 12]), the ring and plaintext modulus of the SHE scheme must be chosen so as to provide many plaintext slots. However, this structure may not always be a natural match for the SHE application's efficiency or functionality requirements. For example, the lattice-based pseudorandom function of [1] works very well with a ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ where both q and n are powers of two, but for such parameters R_q has only *one* slot. Our method can bootstrap even for this kind of plaintext ring (and many others), while still using batching to achieve quasilinear runtime.

1.2 Techniques

At the heart of our bootstrapping procedure are two novel homomorphic operations for SHE schemes over cyclotomic rings: for non-packed (or semi-packed) ciphertexts, we give an operation that *isolates the message-carrying coefficient(s)* of a high-dimensional ring element; and for (semi-)packed ciphertexts, we give an operation that *maps coefficients to slots* and vice-versa.

Isolating coefficients. Our first homomorphic operation is most easily explained in the context of non-packed ciphertexts, which encrypt single elements of the quotient ring $\mathbb{Z}_p$ for some small modulus p, using ciphertexts over some cyclotomic quotient ring $R_q = R/qR$ of moderately large degree $d = \deg(R/\mathbb{Z}) = \tilde{O}(\lambda)$. We first observe that a ciphertext to be bootstrapped can be reinterpreted as an encryption of an R_q-element, one of whose $\mathbb{Z}_q$-coefficients (with respect to an appropriate basis of the ring) "noisily" encodes the message, and whose other coefficients are just meaningless noise terms. We give an simple and efficient homomorphic operation that preserves the meaningful coefficient, and maps all the others to zero. Having isolated the message-encoding coefficient, we can then homomorphically apply an efficient integer "rounding" function (see [12] and the full version) to recover the message from its noisy encoding, which completes the bootstrapping procedure. (Note that it is necessary to remove the meaningless noise coefficients first, otherwise they would interfere with the correct operation of the rounding function.)

Our coefficient-isolating procedure works essentially by applying the *trace function* $\mathrm{Tr}_{R/\mathbb{Z}} \colon R \to \mathbb{Z}$ to the plaintext. The trace is the "canonical" $\mathbb{Z}$-linear function from R to $\mathbb{Z}$, and it turns out that for the appropriate choice of $\mathbb{Z}$-basis of R used in decryption, the trace simply outputs (up to some scaling factor) the message-carrying coefficient we wish to isolate. One simple and very efficient way of applying the trace homomorphically is to use the "ring-switching" technique of [11], but unfortunately, this requires the ring-LWE problem [15] to be hard over the target ring $\mathbb{Z}$, which is clearly not the case. Another way follows from the fact that $\mathrm{Tr}_{R/\mathbb{Z}}$ equals the sum of all d automorphisms of R; therefore, it can be computed by homomorphically applying each automorphism and summing the results. Unfortunately, this method takes at least *quadratic* $\Omega(\lambda^2)$ time, because applying each automorphism homomorphically takes $\Omega(\lambda)$ time, and there are $d = \Omega(\lambda)$ automorphisms.

So, instead of inefficiently computing the trace by summing all the automorphisms at once, we consider a *tower* of cyclotomic rings $\mathbb{Z} = R^{(0)} \subseteq R^{(1)} \subseteq \cdots \subseteq R^{(r)} = R$, usually written as $R^{(r)}/\cdots/R^{(1)}/R^{(0)}$. Then $\mathrm{Tr}_{R/\mathbb{Z}}$ is the composition of the individual trace functions $\mathrm{Tr}_{R^{(i)}/R^{(i-1)}} \colon R^{(i)} \to R^{(i-1)}$, and these traces are equal to the sums of all automorphisms of $R^{(i)}$ that fix $R^{(i-1)}$ pointwise, of which there are exactly $d_i = \deg(R^{(i)}/R^{(i-1)}) = \deg(R^{(i)}/\mathbb{Z})/\deg(R^{(i-1)}/\mathbb{Z})$. We can therefore compute each $\mathrm{Tr}_{R^{(i)}/R^{(i-1)}}$ in time linear in λ and in d_i; moreover, the number of trace functions to apply is at most logarithmic in $d = \deg(R/\mathbb{Z}) = \tilde{O}(\lambda)$, because each one reduces the degree by a factor of at least two. Therefore, by ensuring that the degrees of $R^{(r)}, R^{(r-1)}, \ldots, R^{(0)}$

decrease gradually enough, we can homomorphically apply the full $\mathrm{Tr}_{R/\mathbb{Z}}$ in quasilinear time. For example, a particularly convenient choice is to let $R^{(i)}$ be the 2^{i+1}st cyclotomic ring $\mathbb{Z}[X]/(1+X^{2^i})$ of degree 2^i, so that every $d_i = 2$, and there are exactly $\log_2(d) = O(\log \lambda)$ trace functions to apply.

More generally, when bootstrapping a *semi-packed* ciphertext we start with a plaintext value in R_q that noisily encodes a message in S_p, for some subring $S \subseteq R$. (The case $S = \mathbb{Z}$ corresponds to a non-packed ciphertext.) We show that applying the trace function $\mathrm{Tr}_{R/S}$ to the R_q-plaintext yields a new plaintext *in* S_q that noisily encodes the message, thus isolating the meaningful part of the noisy encoding and vanishing the rest. We then homomorphically apply a rounding function to recover the S_p message from its noisy S_q encoding, which uses the technique described next.

Mapping coefficients to slots. Our second technique, and main technical innovation, is in bootstrapping (semi-)packed ciphertexts. We enhance the recent "ring-switching" procedure of [11], and use it to efficiently move "noisy" plaintext coefficients (with respect to an appropriate decryption basis) into slots for batch-rounding, and finally move the rounded slot values back to coefficients. We note that all previous methods for loading plaintext data into slots used the *same* ring for the source and destination, and so required the plaintext to come from a ring designed to have many slots. In this work, we use ring-switching to go from the SHE plaintext ring to a *different* ring having many slots, which is used only temporarily for batch-rounding. This is what allows the SHE plaintext ring to be decoupled from the rings used in bootstrapping, as mentioned above.

To summarize our technique, we first recall the ring-switching procedure of [11]. It was originally devised to provide moderate efficiency gains for SHE/FHE schemes, by allowing them to switch ciphertexts from high-degree cyclotomic rings to *subrings* of smaller degree (once enough homomorphic operations have been performed to make this secure). We generalize the procedure, showing how to switch between two rings where neither ring need be a subring of the other. The procedure has a very simple implementation, and as long as the two rings have a large *common subring*, it is also very efficient (e.g., quasilinear in the dimension). Moreover, it supports, as a side effect, the homomorphic evaluation of any function that is *linear over the common subring*. However, the larger the common subring is, the more restrictive this condition on the function becomes.

We show how our enhanced ring-switching can move the plaintext coefficients into the slots of the target ring (and back), which can be seen as just evaluating a certain $\mathbb{Z}$-linear function. Here we are faced with the main technical challenge: for efficiency, the common subring of the source and destination rings must be large, but then the supported class of linear functions is very restrictive, and certainly does not include the $\mathbb{Z}$-linear one we want to evaluate. We solve this problem by switching through a short sequence of "hybrid" rings, where adjacent rings have a large common subring, but the initial and final rings have only the integers $\mathbb{Z}$ in common. Moreover, we show that for an appropriately chosen sequence of hybrid rings, the $\mathbb{Z}$-linear function we want to evaluate *is* realizable by a sequence of allowed linear functions between adjacent hybrid rings. Very

critically, this decomposition requires the SHE scheme to use a *highly structured* basis of the ring for decryption. The usual representation of a cyclotomic ring as $\mathbb{Z}[X]/\Phi_m(X)$ typically does not correspond to such a basis, so we instead rely on the *tensorial decomposition* of the ring and its corresponding bases, as recently explored in [16]. At heart, this is what allows us to avoid the expensive homomorphic reduction modulo $\Phi_m(X)$, which is one of the main bottlenecks in previous work [12].[2]

Stepping back a bit, the technique of switching through hybrid rings and bases is reminiscent of standard "sparse decompositions" for linear transformations like the FFT, in that both decompose a complicated high-dimensional transform into a short sequence of simpler, structured transforms. (Here, the simple transforms are computed merely as a side-effect of passing through the hybrid rings.) Because of these similarities, we believe that the enhanced ring-switching procedure will be applicable in other domain-specific applications of homomorphic encryption, e.g., signal-processing transforms or statistical analysis.

Organization. Due to space restrictions, this version of the paper omits much of the algebraic background, several proofs, and some lower-level descriptions of our procedures; see the full version for complete details. Section 2.1 recalls some of the algebraic background required for our constructions, and Section 2.2 recalls a standard ring-based SHE scheme and some of its natural homomorphic operations. Section 3 defines the general bootstrapping procedure. Sections 4 and 5 respectively fill in further details of the two novel homomorphic operations used in the bootstrapping procedure.

The full version also documents a folklore transformation between two essentially equivalent ways of encoding messages in SHE schemes (namely, the "least/most significant bit" encodings), describes an integer rounding procedure that simplifies the one given in [12], and gives some concrete choices of rings that our method can use in practice.

Acknowledgments. We thank Oded Regev for helpful discussions during the early stages of this research, and the anonymous CRYPTO'13 reviewers for their thoughtful comments.

2 Preliminaries

For a positive integer k, we let $[k] = \{0, \ldots, k-1\}$. For an integer modulus q, we let $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ denote the quotient ring of integers modulo q. For integers q, q', we define the integer "rounding" function $\lfloor \cdot \rceil_{q'} \colon \mathbb{Z}_q \to \mathbb{Z}_{q'}$ as $\lfloor x \rceil_{q'} = \lfloor (q'/q) \cdot x \rceil \bmod q'$.

[2] The use of more structured representations of cyclotomic rings in [16] was initially motivated by the desire for simpler and more efficient algorithms for cryptographic operations. Interestingly, these representations yield moderate efficiency improvements for computations "in the clear," but dramatic benefits for their homomorphic counterparts!

2.1 Algebraic Background

Throughout this work, by "ring" we mean a commutative ring with identity. For two rings $R \subseteq R'$, an R-basis of R' is a set $B \subset R'$ such that every $r \in R'$ can be written uniquely as an R-linear combination of elements of B. For two rings R, S with a common subring E, an E-linear function $L\colon R \to S$ is one for which $L(r + r') = L(r) + L(r')$ for all $r, r' \in R$, and $L(e \cdot r) = e \cdot L(r)$ for all $e \in E, r \in R$. It is immediate that such a function is defined uniquely by its values on any E-basis of R.

Cyclotomic Rings. For a positive integer m called the *index*, let $\mathcal{O}_m = \mathbb{Z}[\zeta_m]$ denote the mth *cyclotomic ring*, where ζ_m is an abstract element of order m over $\mathbb{Q}$. (In particular, we do not view ζ_m as any particular complex root of unity.) The minimal polynomial of ζ_m over $\mathbb{Q}$ is the mth *cyclotomic polynomial* $\Phi_m(X) = \prod_{i \in \mathbb{Z}_m^*}(X - \omega_m^i) \in \mathbb{Z}[X]$, where $\omega_m = \exp(2\pi\sqrt{-1}/m) \in \mathbb{C}$ is the principal mth complex root of unity, and the roots $\omega_m^i \in \mathbb{C}$ range over all the *primitive* complex mth roots of unity. Therefore, $\mathcal{O}_m$ is a ring extension of degree $n = \varphi(m)$ over $\mathbb{Z}$. (In particular, $\mathcal{O}_1 = \mathcal{O}_2 = \mathbb{Z}$.) Clearly, $\mathcal{O}_m$ is isomorphic to the polynomial ring $\mathbb{Z}[X]/\Phi_m(X)$ by identifying ζ_m with X, and has the "power basis" $\{1, \zeta_m, \ldots, \zeta_m^{n-1}\}$ as a $\mathbb{Z}$-basis. However, for non-prime-power m the power basis can be somewhat cumbersome and inefficient to work with. In Section 2.1 we consider other, more structured bases that are essential to our techniques.

If $m|m'$, we can view the mth cyclotomic ring $\mathcal{O}_m$ as a subring of $\mathcal{O}_{m'} = \mathbb{Z}[\zeta_{m'}]$, via the ring embedding (i.e., injective ring homomorphism) that maps ζ_m to $\zeta_{m'}^{m'/m}$. The ring extension $\mathcal{O}_{m'}/\mathcal{O}_m$ has degree $d = \varphi(m')/\varphi(m)$, and also d automorphisms τ_i (i.e., automorphisms of $\mathcal{O}_{m'}$ that fix $\mathcal{O}_m$ pointwise), which are defined by $\tau_i(\zeta_{m'}) = \zeta_{m'}^i$ for each $i \in \mathbb{Z}_{m'}^*$ such that $i = 1 \pmod m$. The *trace* function $\mathrm{Tr} = \mathrm{Tr}_{\mathcal{O}_{m'}/\mathcal{O}_m}\colon \mathcal{O}_{m'} \to \mathcal{O}_m$ can be defined as the sum of these automorphisms:

$$\mathrm{Tr}_{\mathcal{O}_{m'}/\mathcal{O}_m}(a) = \sum_i \tau_i(a) \in \mathcal{O}_m.$$

Notice that Tr is $\mathcal{O}_m$-linear by definition. If $\mathcal{O}_{m''}/\mathcal{O}_{m'}/\mathcal{O}_m$ is a tower of ring extensions, then the trace satisfies the composition property $\mathrm{Tr}_{\mathcal{O}_{m''}/\mathcal{O}_m} = \mathrm{Tr}_{\mathcal{O}_{m'}/\mathcal{O}_m} \circ \mathrm{Tr}_{\mathcal{O}_{m''}/\mathcal{O}_{m'}}$.

An important element in the mth cyclotomic ring is

$$g := \prod_{\text{odd prime } p|m} (1 - \zeta_p) \in \mathcal{O}_m. \tag{1}$$

Also define $\hat{m} = m/2$ if m is even, otherwise $\hat{m} = m$, for any cyclotomic index m. It is known that $g|\hat{m}$ (see, e.g., [16, Section 2.5.4]). The following lemma shows how the elements g in different cyclotomic rings, and the ideals they generate, are related by the trace function. (See the full version for a proof.)

Lemma 2.1. *Let $m|m'$ be positive integers and let $g \in R = \mathcal{O}_m, g' \in R' = \mathcal{O}_{m'}$ and $\hat{m}, \hat{m}'$ be as defined above. Then $\mathrm{Tr}_{R'/R}(g'R') = (\hat{m}'/\hat{m}) \cdot gR$, and in particular, $\mathrm{Tr}_{R'/R}(g') = (\hat{m}'/\hat{m}) \cdot g$.*

Later on we use the *scaled* trace function $(\hat{m}/\hat{m}')\,\mathrm{Tr}_{R'/R}$, which by the above lemma maps the ideal $g'R$ to gR, and g' to g.

Tensorial Decomposition of Cyclotomics. An important fact from algebraic number theory, used centrally in this work (and in [16]), is the *tensorial decomposition* of cyclotomic rings (and their bases) in terms of subrings. Let $\mathcal{O}_{m_1}, \mathcal{O}_{m_2}$ be cyclotomic rings. Then their largest common subring is $\mathcal{O}_{m_1} \cap \mathcal{O}_{m_2} = \mathcal{O}_g$ where $g = \gcd(m_1, m_2)$, and their smallest common extension ring, called the *compositum*, is $\mathcal{O}_{m_1} + \mathcal{O}_{m_2} = \mathcal{O}_l$ where $l = \mathrm{lcm}(m_1, m_2)$. When considered as extensions of $\mathcal{O}_g$, the ring $\mathcal{O}_l$ is isomorphic to the *ring tensor product* of $\mathcal{O}_{m_1}$ and $\mathcal{O}_{m_2}$, written as (sometimes suppressing $\mathcal{O}_g$ when it is clear from context)

$$\mathcal{O}_l/\mathcal{O}_g \cong (\mathcal{O}_{m_1}/\mathcal{O}_g) \otimes (\mathcal{O}_{m_2}/\mathcal{O}_g).$$

On the right, the ring tensor product is defined as the set of all $\mathcal{O}_g$-linear combinations of *pure tensors* $a_1 \otimes a_2$, with ring operations defined by $\mathcal{O}_g$-bilinearity and the mixed-product property $(a_1 \otimes a_2) \cdot (b_1 \otimes b_2) = (a_1 b_1) \otimes (a_2 b_2)$. The isomorphism with $\mathcal{O}_l/\mathcal{O}_g$ then simply identifies $a_1 \otimes a_2$ with $a_1 \cdot a_2 \in \mathcal{O}_l$. Note that any $a_1 \in \mathcal{O}_{m_1}$ corresponds to the pure tensor $a_1 \otimes 1$, and similarly for any $a_2 \in \mathcal{O}_{m_2}$.

The following simple lemma will be central to our techniques.

Lemma 2.2. *Let $m_1, m_2 > 0$ be integers and $g = \gcd(m_1, m_2)$, $l = \mathrm{lcm}(m_1, m_2)$. Then for any $\mathcal{O}_g$-linear function $\bar{L}\colon \mathcal{O}_{m_1} \to \mathcal{O}_{m_2}$, there is an (efficiently computable) $\mathcal{O}_{m_2}$-linear function $L\colon \mathcal{O}_l \to \mathcal{O}_{m_2}$ that coincides with $\bar{L}$ on the subring $\mathcal{O}_{m_1} \subseteq \mathcal{O}_l$.*

Proof. Write $\mathcal{O}_l \cong \mathcal{O}_{m_1} \otimes \mathcal{O}_{m_2}$, where the common base ring $\mathcal{O}_g$ is implicit. Let $L\colon (\mathcal{O}_{m_1} \otimes \mathcal{O}_{m_2}) \to \mathcal{O}_{m_2}$ be the $\mathcal{O}_g$-linear function uniquely defined by $L(a_1 \otimes a_2) = \bar{L}(a_1) \cdot a_2 \in \mathcal{O}_{m_2}$ for all pure tensors $a_1 \otimes a_2$. Then because $(a_1 \otimes a_2) \cdot b_2 = a_1 \otimes (a_2 b_2)$ for any $b_2 \in \mathcal{O}_{m_2}$ by the mixed-product property, L is also $\mathcal{O}_{m_2}$-linear. Finally, for any $a_1 \in \mathcal{O}_{m_1}$ we have $L(a_1 \otimes 1) = \bar{L}(a_1)$ by construction.

Ideal Factorization and Plaintext Slots. In the full version we recall the unique factorization of prime integers into prime ideals in cyclotomic rings, and, following [17], how the Chinese remainder theorem can yield several plaintext "slots" that embed $\mathbb{Z}_q$ as a subring, even for composite q.

In brief, for any prime integer p and cyclotomic ring R, the ideal pR factors as $pR = \prod_i \mathfrak{p}_i^e$ for some distinct prime ideals $\mathfrak{p}_i$ and some $e \geq 1$. Moreover, for any power $q = p^r$ where $r \geq 1$, the quotient ring $R/\mathfrak{p}_i^{re}$ embeds $\mathbb{Z}_q$ as a subring. By the Chinese Remainder Theorem (CRT), the natural ring homomorphism from R_q to the product ring $\bigoplus_i (R/\mathfrak{p}_i^{re})$ is an isomorphism. When the natural plaintext space of a cryptosystem is R_q, we refer to the quotient rings $R/\mathfrak{p}_i^{re}$ as the plaintext "$\mathbb{Z}_q$-slots" (or just "slots"), and use them to store vectors of $\mathbb{Z}_q$-elements via the CRT isomorphism. With this encoding, ring operations in R_q induce "batch" (or "SIMD") component-wise operations on the corresponding

vectors of $\mathbb{Z}_q$ elements. We note that the CRT isomorphism is easy to compute in both directions. In particular, to map from a vector of $\mathbb{Z}_q$-elements to R_q just requires knowing a fixed "mod-q CRT set" $C = \{c_i\} \subset R$ for which $c_i = 1 \pmod{\mathfrak{p}_i^{r_e}}$ and $c_i = 0 \pmod{\mathfrak{p}_j^{r_e}}$ for all $j \neq i$. Such a set can be precomputed using, e.g., a generalization of the extended Euclidean algorithm.

Product Bases. Our bootstrapping technique relies crucially on certain highly structured bases and CRT sets, which we call "product bases (sets)," that arise from towers of cyclotomic rings. Let $\mathcal{O}_{m''}/\mathcal{O}_{m'}/\mathcal{O}_m$ be such a tower, let $B'' = \{b''_{j''}\} \subset \mathcal{O}_{m''}$ be any $\mathcal{O}_{m'}$-basis of $\mathcal{O}_{m''}$, and let $B' = \{b'_{j'}\} \subset \mathcal{O}_{m'}$ be any $\mathcal{O}_m$-basis of $\mathcal{O}_{m'}$. Then it follows immediately that the product set $B'' \cdot B' := \{b''_{j''} \cdot b'_{j'}\} \subset \mathcal{O}_{m''}$ is an $\mathcal{O}_m$-basis of $\mathcal{O}_{m''}$.[3] Of course, for a tower of several cyclotomic extensions and relative bases, we can obtain product bases that factor with a corresponding degree of granularity.

In the full version we show that the "powerful" and "decoding" bases of cyclotomic rings R, as defined in [16], admit "finest-possible" product structures, corresponding to any desired tower $R/\cdots/\mathbb{Z}$ of cyclotomic rings. (Other commonly used bases of $\mathcal{O}_m$, such as the power $\mathbb{Z}$-basis, do not admit such factorizations unless m is a prime power.) Similarly, we show how to construct CRT sets that have finest-possible factorizations.

2.2 Ring-Based Homomorphic Cryptosystem

Here we recall a somewhat-homomorphic encryption scheme whose security is based on the ring-LWE problem [15] in arbitrary cyclotomic rings. For our purposes we focus mainly on its decryption function, though below we also recall its support for "ring switching" [11]. For further details on its security guarantees, homomorphic properties, and efficient implementation, see [15, 5, 3, 14, 11, 16].

Let $R = \mathcal{O}_m \subseteq R' = \mathcal{O}_{m'}$ be respectively the mth and m'th cyclotomic rings, where $m|m'$. The plaintext ring is the quotient ring R_p for some integer p; ciphertexts are made up of elements of R'_q for some integer q, which for simplicity we assume is divisible by p; and the secret key is some $s \in R'$. The case $m = 1$ corresponds to "non-packed" ciphertexts, which encrypt elements of $\mathbb{Z}_p$ (e.g., single bits), whereas $m = m'$ corresponds to "packed" ciphertexts, and $1 < m < m'$ corresponds to what we call "semi-packed" ciphertexts. Note that without loss of generality we can treat any ciphertext as packed, since R'_p embeds R_p. But the smaller m is, the simpler and more practically efficient our bootstrapping procedure can be. Since our focus is on refreshing ciphertexts that have large noise rate, we can think of m' as being somewhat small (e.g., in the several hundreds) via ring-switching [11], and q also as being somewhat small (e.g., in the several thousands) via modulus-switching. Our main focus in this work is on a plaintext modulus p that is a power of two, though for generality we present all our techniques in terms of arbitrary p.

[3] Formally, this basis is a *Kronecker* product of the bases B'' and B', which is typically written using the $\otimes$ operator. We instead use $\cdot$ to avoid confusion with pure tensors in a ring tensor product, which the elements of $B'' \cdot B'$ may not necessarily be.

A ciphertext encrypting a message $\mu \in R_p$ under secret key $s' \in R'$ is some pair $c' = (c'_0, c'_1) \in R'_q \times R'_q$ satisfying the relation

$$c'_0 + c'_1 \cdot s' = \frac{q}{p} \cdot \mu + e' \pmod{qR'} \tag{2}$$

for some error (or "noise") term $e' \in R'$ such that $e' \cdot g' \in g'R'$ is sufficiently "short," where $g' \in R'$ is as defined in Equation (1).[4] Informally, the "noise rate" of the ciphertext is the ratio of the "size" of e' (or more precisely, the magnitude of its coefficients in a suitable basis) to q/p.

We note that Equation (2) corresponds to what is sometimes called the "most significant bit" (msb) message encoding, whereas somewhat-homomorphic schemes are often defined using "least significant bit" (lsb) encoding, in which p and q are coprime and $c'_0 + c'_1 s' = e' \pmod{qR'}$ for some error term $e' \in \mu + pR'$. For our purposes the msb encoding is more natural, and in any case the two encodings are essentially equivalent; see the full version for details.

Decryption. At a high level, the decryption algorithm works in two steps: the "linear" step simply computes $v' = c'_0 + c'_1 \cdot s' = \frac{q}{p} \cdot \mu + e' \in R'_q$, and the "non-linear" step outputs $\lfloor v' \rceil_p \in R_p$ using a certain "ring rounding function" $\lfloor \cdot \rceil_p \colon R'_q \to R_p$. As long as the error term e' is within the tolerance of the rounding function, the output will be $\mu \in R_p$. This is all entirely analogous to decryption in LWE-based systems, but here the rounding is n-dimensional, rather than just from $\mathbb{Z}_q$ to $\mathbb{Z}_p$.

Concretely, the ring rounding function $\lfloor \cdot \rceil_p \colon R'_q \to R_p$ is defined in terms of the integer rounding function $\lfloor \cdot \rceil_p \colon \mathbb{Z}_q \to \mathbb{Z}_p$ and a certain "decryption" $\mathbb{Z}$-basis $B' = \{b_j\}$ of R', as follows.[5] Represent the input $v' \in R'_q$ in the decryption basis as $v' = \sum_j v'_j \cdot b'_j$ for some coefficients $v'_j \in \mathbb{Z}_q$, then independently round the coefficients, yielding an element $\sum \lfloor v'_j \rceil_p \cdot b'_j \in R'_p$ that corresponds to the message $\mu \in R_p$ (under the standard embedding of R_p into R'_p).

Changing the Plaintext Modulus. We use two operations on ciphertexts that alter the plaintext modulus p and encrypted message $\mu \in R_p$. The first operation changes p to any multiple $p' = dp$, and produces an encryption of

[4] Quantitatively, "short" is defined with respect to the *canonical embedding* of R', whose precise definition is not needed in this work. The above system is equivalent to the one from [16] in which the message, error term, and ciphertext components are all taken over the "dual" fractional ideal $(R')^\vee = (g'/\hat{m}')R'$ in the m'th cyclotomic number field, and the error term has an essentially spherical distribution over $(R')^\vee$. In that system, decryption is best accomplished using a certain $\mathbb{Z}$-basis of $(R')^\vee$, called the *decoding basis*, which optimally decodes spherical errors. The above formulation is more convenient for our purposes, and simply corresponds with multiplying everything in the system of [16] by an $\hat{m}'/g'$ factor. This makes $e' \cdot g' \in g'R' = \hat{m}'(R')^\vee$ short and essentially spherical in our formulation. See [15, 16] for further details.

[5] In our formulation, the basis B' is $(\hat{m}'/g')$ times the decoding basis of $(R')^\vee$. See Section 2.1 and Footnote 4.

some $\mu' \in R'_{p'}$ such that $\mu' = \mu \pmod{pR'}$. To do this, it simply "lifts" the input ciphertext $c' = (c'_0, c'_1) \in (R'_q)^2$ to an arbitrary $c'' = (c''_0, c''_1) \in (R'_{q'})^2$ such that $c''_j = c'_j \pmod{qR'}$, where $q' = dq$. The second operation applies to an encryption of a message $\mu \in R_p$ that is known to be divisible by some divisor d of p, and produces an encryption of $\mu/d \in R_{p/d}$. The operation actually leaves the ciphertext c' unchanged; it just declares the associated plaintext modulus to be p/d (which affects how decryption is performed).

Ring Switching. We rely heavily on the cryptosystem's support for switching ciphertexts to a cyclotomic subring S' of R', which as a side-effect homomorphically evaluates any desired S'-linear function on the plaintext. Notice that the linear function L is applied to the plaintext as embedded in R'_p; this obviously applies the induced function on the true plaintext space R_p.

Proposition 2.3 ([11], full version). *Let $S' \subseteq R'$ be cyclotomic rings. Then the above-described cryptosystem supports the following homomorphic operation: given any S'-linear function $L\colon R'_p \to S'_p$ and a ciphertext over R'_q encrypting (with sufficiently small error term) a message $\mu \in R'_p$, the output is a ciphertext over S'_q encrypting $L(\mu) \in S'_p$.*

The security of the procedure described in Proposition 2.3 is based on the hardness of the ring-LWE problem in S', so the dimension of S' must be sufficiently large. The procedure itself is quite simple and efficient: it first switches to a secret key that lies in the subring S', then it multiplies the resulting ciphertext by an appropriate fixed element of R' (which is determined solely by the function L). Finally, it applies to the ciphertext the trace function $\mathrm{Tr}_{R'/S'}\colon R' \to S'$. All of these operations are quasi-linear time in the dimension of $R'/\mathbb{Z}$, and very efficient in practice. In particular, the trace is a trivial linear-time operation when elements are represented in any of the bases we use. The ring-switching procedure increases the effective error rate of the ciphertext by a factor of about the square root of the dimension of R', which is comparable to that of a single homomorphic multiplication. See [11] for further details.

3 Overview of Bootstrapping Procedure

Here we give a high-level description of our bootstrapping procedure. We present a unified procedure for non-packed, packed, and semi-packed ciphertexts, but note that for non-packed ciphertexts, Steps 3a and 3c (and possibly 1c) are null operations, while for packed ciphertexts, Steps 1b, 1c, and 2 are null operations.

Recalling the cryptosystem from Section 2.2, the plaintext ring is R_p and the ciphertext ring is R'_q, where $R = \mathcal{O}_m \subseteq R' = \mathcal{O}_{m'}$ are cyclotomic rings (so $m|m'$), and q is a power of p. The procedure also uses a larger cyclotomic ring $R'' = \mathcal{O}_{m''} \supseteq R'$ (so $m'|m''$) to work with ciphertexts that encrypt elements of the original ciphertext ring R'_q. We can choose m'' however we like, subject to the constraints below.

To obtain quasilinear runtimes and exponential security under standard hardness assumptions, our procedure imposes some mild conditions on the indices m, m', and m'':

- The dimension $\varphi(m'')$ of R'' must be quasilinear, so we can represent elements of R'' efficiently.
- For Steps 2 and 3, all the prime divisors of m and m' must be small (i.e., polylogarithmic).
- For Step 3, m and m''/m must be coprime, which implies that m and m'/m must be coprime also. Note that the former condition is always satisfied for non-packed ciphertexts (where $m = 1$). For packed ciphertexts (where $m = m'$), the latter condition is always satisfied, which makes it easy to choose a valid m''. For semi-packed ciphertexts (where $1 < m < m'$), we can always satisfy the latter condition either by increasing m (at a small expense in practical efficiency in Step 3), or by effectively decreasing m slightly (at a possible improvement in practical efficiency); see the full version for details.

The input to the procedure is a ciphertext $c' = (c'_0, c'_1) \in (R'_q)^2$ that encrypts some plaintext $\mu \in R_p$ under a secret key $s' \in R'$, i.e., it satisfies the relation

$$v' = c'_0 + c'_1 \cdot s' = \frac{q}{p} \cdot \mu + e' \pmod{qR'}$$

for some small enough error term $e' \in R'$. The procedure computes a new encryption of $\lfloor v \rceil_p = \mu$ (under some secret key, not necessarily s') that has substantially smaller noise rate than the input ciphertext. It proceeds as follows (explanatory remarks appear in italics):

1. Convert c' to a "noiseless" ciphertext c'' over a large ring R''_Q that encrypts a plaintext $(g'/g)u' \in R'_{q'}$, where $g' \in R', g \in R$ are as defined in Equation (1), $q' = (\hat{m}'/\hat{m})q$, and $u' = v' \pmod{qR'}$. This proceeds in the following substeps (see Section 3.1 for further details).
   Note that $g'/g \in R'$ by definition, and that $(g'/g)|(\hat{m}'/\hat{m})$.
   (a) Reinterpret c' as a noiseless encryption of $v' = \frac{q}{p} \cdot \mu + e' \in R'_q$ as a *plaintext*, noting that both the plaintext and ciphertext rings are now taken to be R'_q.
   This is purely a conceptual change in perspective, and does not involve any computation.
   (b) Using the procedure described in Section 2.2, change the plaintext (and ciphertext) modulus to $q' = (\hat{m}'/\hat{m})q$, yielding a noiseless encryption of some $u' \in R'_{q'}$ such that $u' = v' \pmod{qR'}$.
   Note that this step is a null operation if the original ciphertext was packed, i.e., if $m = m'$.
   We need to increase the plaintext modulus because homomorphically computing $\mathrm{Tr}_{R'/R}$ in Step 2 below introduces an $\hat{m}'/\hat{m}$ factor into the plaintext, which we will undo by scaling the plaintext modulus back down to q.
   (c) Multiply the ciphertext from the previous step by $(g'/g) \in R'$, yielding a noiseless encryption of plaintext $(g'/g)u' \in R'_{q'}$.

The factor $(g'/g) \in R'$ is needed when we homomorphically compute $\mathrm{Tr}_{R'/R}$ in Step 2 below. Note that $g'/g = 1$ if and only if every odd prime divisor of m' also divides m, e.g., if $m = m'$.

(d) Convert to a noiseless ciphertext c'' that still encrypts $(g'/g)u' \in R'_{q'}$, but using a large enough ciphertext ring R''_Q for some $R'' = \mathcal{O}_{m''} \supseteq R'$ and modulus $Q \gg q'$.

A larger ciphertext ring R''_Q is needed for security in the upcoming homomorphic operations, to compensate for the low noise rates that will need to be used. These operations will expand the initial noise rate by a quasipolynomial $\lambda^{O(\log \lambda)}$ factor in total, so the dimension of R'' and the bit length of Q can be $\tilde{O}(\lambda)$ and $\tilde{O}(1)$, respectively.

The remaining steps are described here only in terms of their effect on the *plaintext* value and ring. Using ring- and modulus-switching, the ciphertext ring R'' and modulus Q may be made smaller as is convenient, subject to the security and functionality requirements. (Also, the ciphertext ring implicitly changes during Steps 3a and 3c.)

2. Homomorphically apply the scaled trace function $(\hat{m}/\hat{m}')\,\mathrm{Tr}_{R'/R}$ to the encryption of $(g'/g)u' \in R'_{q'}$, to obtain an encryption of plaintext

$$u = \frac{\hat{m}}{\hat{m}'} \cdot \mathrm{Tr}_{R'/R}\Big(\frac{g'}{g} \cdot u'\Big) = \frac{q}{p} \cdot \mu + e \in R_q$$

for some suitably small error term $e \in R$. See Section 4 further details.

This step changes the plaintext ring from $R'_{q'}$ to R_q, and homomorphically isolates the noisy R_q-encoding of μ. It is a null operation if the original ciphertext was packed, i.e., if $m = m'$.

3. Homomorphically apply the ring rounding function $\lfloor \cdot \rceil_p \colon R_q \to R_p$, yielding an output ciphertext that encrypts $\lfloor u \rceil_p = \mu \in R_p$. This proceeds in three sub-steps, all of which are applied homomorphically (see Section 5 for details):

(a) Map the coefficients u_j of $u \in R_q$ (with respect to the decryption basis B of R) to the $\mathbb{Z}_q$-slots of a ring S_q, where S is a suitably chosen cyclotomic. *This step changes the plaintext ring from R_q to S_q. It is a null operation if the original ciphertext was non-packed (i.e., if $m = 1$), because we can let $S = R = \mathbb{Z}$.*

(b) Batch-apply the integer rounding function $\lfloor \cdot \rceil \colon \mathbb{Z}_q \to \mathbb{Z}_p$ to the $\mathbb{Z}_q$-slots of S_q, yielding a ciphertext that encrypts the values $\mu_j = \lfloor u_j \rceil_p \in \mathbb{Z}_p$ in its $\mathbb{Z}_p$-slots.

This step changes the plaintext ring from S_q to S_p. It constitutes the only non-linear operation on the plaintext, with multiplicative depth $\lceil \lg p \rceil \cdot (\log_p(q) - 1) \approx \log(q)$, and as such is the most expensive in terms of runtime, noise expansion, etc.

(c) Reverse the map from the step 3a, sending the values μ_j from the $\mathbb{Z}_p$-slots of S_p to coefficients with respect to the decryption basis B of R_p, yielding an encryption of $\mu = \sum_j \mu_j b_j \in R_p$.

This step changes the plaintext ring from S_p to R_p. Just like step 3a, it is a null operation for non-packed ciphertexts.

In the full version we describe a few minor variants and practical optimizations of our basic procedure.

3.1 Obtaining a Noiseless Ciphertext

Step 1 of our bootstrapping procedure is given as input a ciphertext $c' = (c'_0, c'_1)$ over R'_q that encrypts (typically with a high noise rate) a message $\mu \in R_p$ under key $s' \in R$, i.e., $v' = c'_0 + c'_1 \cdot s' = \frac{q}{p} \cdot \mu + e' \in R'_q$ for some error term e'. We first change our perspective and view c' as a "noiseless" encryption (still under s') of the plaintext value $v' \in R'_q$, taking both the plaintext and ciphertext rings to be R'_q. This view is indeed formally correct, because

$$c'_0 + c'_1 \cdot s' = \frac{q}{q} \cdot v' + 0 \pmod{qR'}.$$

Next, in preparation for the upcoming homomorphic operations we increase the plaintext (and ciphertext) modulus to q', and multiply the resulting ciphertext by g'/g. These operations clearly preserve noiselessness. Finally, we convert the ciphertext ring to R''_Q for a sufficiently large cyclotomic $R'' \supseteq R'$ and modulus $Q \gg q$ that is divisible by q. This is done simply by embedding R' into R'' and introducing extra precision, i.e., scaling the ciphertext up by a Q/q factor. It is easy to verify that these operations also preserve noiselessness.

4 Homomorphic Trace

Here we show how to perform Step 2 of our bootstrapping procedure, which homomorphically evaluates the scaled trace function $(\hat{m}/\hat{m}')\,\mathrm{Tr}_{R'/R}$ on an encryption of $(g'/g)u' \in R'_{q'}$, where recall that: $g' \in R', g \in R$ are as defined in Equation (1), and (g'/g) divides $(\hat{m}'/\hat{m})$; the plaintext modulus is $q' = (\hat{m}'/\hat{m})q$; and

$$u' = v' = \frac{q}{p} \cdot \mu + e' \pmod{qR'},$$

where $e' \cdot g' \in g'R'$ is sufficiently short. Our goal is to show that:

1. the scaled trace of the plaintext $(g'/g)u'$ is some $u = \frac{q}{p} \cdot \mu + e \in R_q$, where $e \cdot g \in gR$ is short, and
2. we can efficiently homomorphically apply the scaled trace on a ciphertext c'' over some larger ring $R'' = \mathcal{O}_{m''} \supseteq R'$.

4.1 Trace of the Plaintext

We first show the effect of the scaled trace on the plaintext $(g'/g)u' \in R'_{q'}$. By the above description of $u' \in R'_{q'}$ and the fact that $(g'/g)q$ divides $q' = (\hat{m}'/\hat{m})q$, we have

$$(g'/g)u' = (g'/g)v' = (g'/g)\left(\frac{q}{p} \cdot \mu + e'\right) \pmod{(g'/g)qR'}.$$

Therefore, letting $\mathrm{Tr} = \mathrm{Tr}_{R'/R}$, by R-linearity of the trace and Lemma 2.1, we have

$$\begin{aligned}\mathrm{Tr}((g'/g)u') &= \mathrm{Tr}(g'/g) \cdot \frac{q}{p} \cdot \mu + \mathrm{Tr}(e' \cdot g')/g \\ &= \frac{\hat{m}'}{\hat{m}}\left(\frac{q}{p} \cdot \mu + e\right) \pmod{q'R},\end{aligned}$$

where $e = (\hat{m}/\hat{m}')\,\mathrm{Tr}(e' \cdot g')/g \in R$. Therefore, after scaling down the plaintext modulus q' by an $\hat{m}'/\hat{m}$ factor (see Section 2.2), the plaintext is $\frac{q}{p} \cdot \mu + e \in R_q$.

Moreover, $e \cdot g = (\hat{m}/\hat{m}')\,\mathrm{Tr}(e' \cdot g') \in gR$ is short because $e' \cdot g' \in g'R'$ is short; see, e.g., [11, Corollary 2.2]. In fact, by basic properties of the decoding/decryption basis (as defined in [16]) under the trace, the coefficient vector of e with respect to the decryption basis of R is merely a subvector of the coefficient vector of e' with respect to the decryption basis of R'. Therefore, e is within the error tolerance of the rounding function on R_q, assuming e' is within the error tolerance of the rounding function on R'_q.

4.2 Applying the Trace

Now we show how to efficiently homomorphically apply the scaled trace function $(\hat{m}/\hat{m}')\,\mathrm{Tr}_{R'/R}$ to an encryption of any plaintext in $R'_{q'}$ that is divisible by (g'/g). Note that this condition ensures that the output of the trace is a multiple of $\hat{m}/\hat{m}'$ in $R_{q'}$ (see Lemma 2.1), making the scaling a well-defined operation that results in an element of R_q.

First recall that $\mathrm{Tr}_{R'/R}$ is the sum of all $\varphi(m')/\varphi(m)$ automorphisms of R'/R, i.e., automorphisms of R' that fix R pointwise. So as mentioned in the introduction, one way of homomorphically computing the scaled trace is to homomorphically apply the proper automorphisms, sum the results, and scale down the plaintext and its modulus. While this "sum-automorphisms" procedure yields the correct result, computing the trace in this way does not run in quasilinear time, unless the number $\varphi(m')/\varphi(m)$ of automorphisms is only polylogarithmic.

Instead, we consider a sufficiently fine-grained tower of cyclotomic rings

$$R^{(r)}/\cdots/R^{(1)}/R^{(0)},$$

where $R' = R^{(r)}$, $R = R^{(0)}$, and each $R^{(i)} = \mathcal{O}_{m_i}$, where m_i is divisible by m_{i-1} for $i > 0$. E.g., for the finest granularity we would choose the tower so that every m_i/m_{i-1} is prime. Notice that the scaled trace function $(\hat{m}/\hat{m}')\,\mathrm{Tr}_{R'/R}$ is the composition of the scaled trace functions $(\hat{m}_{i-1}/\hat{m}_i)\,\mathrm{Tr}_{R^{(i)}/R^{(i-1)}}$, and that g'/g is the product of all $g^{(i)}/g^{(i-1)}$ for $i = 1, \ldots, r$, where $g^{(i)} \in R^{(i)}$ is as defined in Equation (1). So, another way of homomorphically applying the full scaled trace is to apply the corresponding scaled trace in sequence for each level of the tower, "climbing down" from $R' = R^{(r)}$ to $R = R^{(0)}$. In particular, if we use the above sum-automorphisms procedure with a tower of finest granularity, then there are at most $\log_2(m'/m) = O(\log \lambda)$ levels, and since we have assumed

that every prime divisor of m'/m is bounded by polylogarithmic in the security parameter λ, the full procedure will run in quasilinear $\tilde{O}(\lambda)$ time.

In the full version we give all the details of the sum-automorphisms procedure sketched above, as well as an alternative procedure using ring-switching that is preferable in certain cases.

5 Homomorphic Ring Rounding

In this section we describe how to efficiently homomorphically evaluate the "ring rounding function" $\lfloor \cdot \rceil_p \colon R_q \to R_p$, where $R = \mathcal{O}_m$ is the mth cyclotomic ring. Conceptually, we follow the high-level strategy from [12], but instantiate it with very different technical components. Recall from Section 2.2 that the rounding function expresses its input u in the "decryption" $\mathbb{Z}$-basis $B = \{b_j\}$ of R, as $u = \sum_j u_j \cdot b_j$ for $u_j \in \mathbb{Z}_q$, and outputs $\lfloor u \rceil_p := \sum_j \lfloor u_j \rceil_p \cdot b_j \in R_p$. Unlike with integer rounding from $\mathbb{Z}_q$ to $\mathbb{Z}_p$, it is not clear whether this rounding function has a low-depth arithmetic formula using just the ring operations of R. One difficulty is that there are an *exponentially* large number of values in R_q that map to a given value in R_p, which might be seen as evidence that a corresponding arithmetic formula must have large depth. Fortunately, we show how to circumvent this issue by using an additional homomorphic operation, namely, an enhancement of ring-switching. In short, we reduce the homomorphic evaluation of the ring rounding function (from R_q to R_p) very simply and efficiently to that of several parallel (batched) evaluations of the integer rounding function (from $\mathbb{Z}_q$ to $\mathbb{Z}_p$).

Suppose we choose some cyclotomic ring $S = \mathcal{O}_\ell$ having a mod-q CRT set $C = \{c_j\} \subset S$ of cardinality exactly $|B|$. That is, we have a ring embedding from the product ring $\mathbb{Z}_q^{|B|}$ into S_q, given by $\mathbf{u} \mapsto \sum_j u_j \cdot c_j$. Note that the choice of the ring S is at our convenience, and need not have any relationship to the plaintext ring R_q. We express the rounding function $R_q \to R_p$ as a sequence of three steps:

1. Map $u = \sum_j u_j \cdot b_j \in R_q$ to $\sum_j u_j \cdot c_j \in S_q$, i.e., send the $\mathbb{Z}_q$-coefficients of u (with respect to the decryption basis B) to the $\mathbb{Z}_q$-slots of S_q.
2. Batch-apply the integer rounding function from $\mathbb{Z}_q$ to $\mathbb{Z}_p$ to the slot values u_j, to get $\sum_j \lfloor u_j \rceil_p \cdot c_j \in S_2$.
3. Invert the map from the first step to obtain $\lfloor u \rceil_p = \sum_j \lfloor u_j \rceil_2 \cdot b_j \in R_2$.

Using batch/SIMD operations [17], the second step is easily achieved using the fact that S_q embeds the product of several copies of the ring $\mathbb{Z}_q$, via the CRT elements c_j. That is, we can simultaneously round all the coefficients u_j to $\mathbb{Z}_p$, using just one evaluation of an arithmetic procedure over S corresponding to one for the integer rounding function from $\mathbb{Z}_q$ to $\mathbb{Z}_p$.

We now describe one way of expressing the first and third steps above, in terms of operations that can be evaluated homomorphically. The first simple observation is that the function mapping $u = \sum_j u_j \cdot b_j$ to $\sum_j u_j \cdot c_j$ is induced by a $\mathbb{Z}$-linear function $\bar{L} \colon R \to S$. Specifically, $\bar{L}$ simply maps each $\mathbb{Z}$-basis element b_j to c_j. Now suppose that we choose S so that its largest common

subring with R is $\mathbb{Z}$, i.e., the indices m, ℓ are coprime. Then letting $T = R + S = \mathcal{O}_{m\ell} \cong R \otimes S$ be the compositum ring, Lemma 2.2 yields an S-linear function $L\colon T \to S$ that coincides with $\bar{L}$ on $R \subseteq T$, and in particular on u. The ring-switching procedure from Proposition 2.3 can homomorphically evaluate any S-linear function from T to S, and in particular, the function L. Therefore, by simply embedding R into T, we can homomorphically evaluate $\bar{L}(x) = L(x)$ by applying the ring-switching procedure with L.

Unfortunately, there is a major problem with the efficiency of the above approach: the *dimension* (over $\mathbb{Z}$) of the compositum ring T is the product of those of R and S, which are each at least linear in the security parameter. Therefore, representing and operating on arbitrary elements in T requires at least quadratic time.

Efficiently Mapping from B to C. In hindsight, the quadratic runtime of the above approach should not be a surprise, because we treated $\bar{L}\colon R \to S$ as an arbitrary $\mathbb{Z}$-linear transformation, and B, C as arbitrary sets. To do better, $\bar{L}$, B, and C must have some structure we can exploit. Fortunately, they can—*if* we choose them carefully. We now describe a way of expressing the first and third steps above in terms of simple operations that can be evaluated homomorphically in quasilinear time.

The main idea is as follows: instead of mapping directly from R to S, we will express $\bar{L}$ as a *sequence* of linear transformations $\bar{L}_1, \ldots, \bar{L}_r$ through several "hybrid" cyclotomic rings $R = H^{(0)}, H^{(1)}, \ldots, H^{(r)} = S$. For sets B and C with an appropriate product structure, these transformations will respectively map $A_0 = B \subset H^{(0)}$ to some structured subset $A_1 \subset H^{(1)}$, then A_1 to some structured subset $A_2 \subset H^{(2)}$, and so on, finally mapping A_{r-1} to $A_r = C \subset H^{(r)}$. In contrast to the inefficient method described above, the hybrid rings will be chosen so that each compositum $T^{(i)} = H^{(i-1)} + H^{(i)}$ of adjacent rings has dimension just *slightly larger* (by only a polylogarithmic factor) than that of R. This is achieved by choosing the indices of $H^{(i-1)}, H^{(i)}$ to have large greatest common divisor, and hence small least common multiple. For example, the indices can share almost all the same prime divisors (with multiplicity), and have just one different prime divisor each. Of course, other tradeoffs between the number of hybrid rings and the dimensions of the compositums are also possible.

The flip side of this approach is that using ring-switching, we can homomorphically evaluate only $E^{(i)}$-*linear* functions $\bar{L}_i\colon H^{(i-1)} \to H^{(i)}$, where $E^{(i)} = H^{(i-1)} \cap H^{(i)}$ is the largest common subring of adjacent hybrid rings. Since each $E^{(i)}$ is large by design (to keep the compositum $T^{(i)}$ small), this requirement is quite strict, yet we still need to construct linear functions $\bar{L}_i$ that sequentially map $B = A_0$ to $C = A_r$. To achieve this, we construct all the sets A_i to have appropriate product structure. Specifically, we ensure that for each $i = 1, \ldots, r$, we have factorizations

$$A_{i-1} = A_{i-1}^{\text{out}} \cdot Z_i, \quad A_i = A_i^{\text{in}} \cdot Z_i \tag{3}$$

for some set $Z_i \subset E^{(i)}$, where both A_{i-1}^{out} and A_i^{in} are linearly independent over $E^{(i)}$. (Note that for $1 \leq i < r$, each A_i needs to factor in two ways over

two subrings $E^{(i-1)}$ and $E^{(i)}$, which is why we need two sets A_i^{in} and A_i^{out}.) Then, we simply define $\bar{L}_i$ to be an arbitrary $E^{(i)}$-linear function that bijectively maps A_{i-1}^{out} to A_i^{in}. (Note that A_{i-1}^{out} and A_i^{in} have the same cardinality, because A_{i-1} and A_i do.) It immediately follows that $\bar{L}_i$ bijectively maps A_{i-1} to A_i, because

$$\bar{L}_i(A_{i-1}) = \bar{L}_i(A_{i-1}^{\text{out}} \cdot Z_i) = \bar{L}_i(A_{i-1}^{\text{out}}) \cdot Z_i = A_i^{\text{in}} \cdot Z_i$$

by $E^{(i)}$-linearity and the fact that $Z_i \subset E^{(i)}$.

Summarizing the above discusion, we have the following theorem.

Theorem 5.1. *Suppose there are cyclotomic rings $R = H^{(0)}, H^{(1)}, \ldots, H^{(r)} = S$ and sets $A_i \subset H^{(i)}$ such that for all $i = 1, \ldots, r$, we have $A_{i-1} = A_{i-1}^{out} \cdot Z_i$ and $A_i = A_i^{in} \cdot Z_i$ for some sets $Z_i \subset E^{(i)} = H^{(i-1)} \cap H^{(i)}$ and A_{i-1}^{out}, A_i^{in} that are each $E^{(i)}$-linearly independent and of equal cardinality. Then there is a sequence of $E^{(i)}$-linear maps $\bar{L}_i \colon H^{(i-1)} \to H^{(i)}$, for $i = 1, \ldots, r$, whose composition $\bar{L}_r \circ \cdots \circ \bar{L}_1$ bijectively maps A_0 to A_r.*

So far we have described how our desired map between *plaintext* rings R and S can be expressed as a sequence of linear maps through hybrid rings. In the context of bootstrapping, for security these plaintext rings typically need to be embedded in some larger ciphertext rings, because the dimensions of R, S are not large enough to securely support the very small noise used in bootstrapping. For example, following Step 2 of our bootstrapping procedure (Section 3), we have a ciphertext over the ring R'' where $R'' = \mathcal{O}_{m''} \supseteq R$ for some m'' of our choice that is divisible by m. We need to choose the sequence of hybrid *ciphertext* rings so that they admit suitable linear functions that induce the desired ones on the corresponding *plaintext* rings. Achieving this is easy; see the full version for details.

Construction. In the full version we construct hybrid cyclotomic rings $R = H^{(0)}, H^{(1)}, \ldots, H^{(r)} = S$ and sets $A_i \subset H^{(i)}$ (where $A_0 = B$ and $A_r = C$) to satisfy the following two properties for each $i = 1, \ldots, r$:

1. Each compositum $T^{(i)} = H^{(i-1)} + H^{(i)}$ is not too large, i.e., its dimension is quasilinear.
2. The sets A_{i-1}, A_i factor as described in Equation (3).

The main ideas are as follows: view R as the top level of a fine-grained cyclotomic tower, and choose a target ring S as the top level of a fine-grained tower that has sufficiently many $\mathbb{Z}_q$-slots at each level. Consider finest-possible factorizations of the decryption basis B of R, and of a mod-q CRT set C of S. Then to define the hybrid rings and sets A_{i-1}, A_i, for each successive hybrid ring we "tear down" a level from the top of the R-tower and the corresponding component of B, and "build up" another level from the bottom of the S-tower and the corresponding component of the CRT set C.

References

[1] Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 719–737. Springer, Heidelberg (2012)

[2] Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical gapSVP. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 868–886. Springer, Heidelberg (2012)

[3] Brakerski, Z., Gentry, C., Vaikuntanathan, V. (Leveled) fully homomorphic encryption without bootstrapping. In: ICTS, pp. 309–325 (2012)

[4] Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: FOCS, pp. 97–106 (2011)

[5] Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-LWE and security for key dependent messages. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 505–524. Springer, Heidelberg (2011)

[6] Cheon, J.H., Coron, J.-S., Kim, J., Lee, M.S., Lepoint, T., Tibouchi, M., Yun, A.: Batch fully homomorphic encryption over the integers. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 315–335. Springer, Heidelberg (2013)

[7] Gentry, C.: A fully homomorphic encryption scheme. PhD thesis, Stanford University (2009), `http://crypto.stanford.edu/craig`

[8] Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC, pp. 169–178 (2009)

[9] Gentry, C., Halevi, S.: Fully homomorphic encryption without squashing using depth-3 arithmetic circuits. In: FOCS, pp. 107–109 (2011)

[10] Gentry, C., Halevi, S.: Implementing Gentry's fully-homomorphic encryption scheme. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 129–148. Springer, Heidelberg (2011)

[11] Gentry, C., Halevi, S., Peikert, C., Smart, N.P.: Ring switching in BGV-style homomorphic encryption. In: Visconti, I., De Prisco, R. (eds.) SCN 2012. LNCS, vol. 7485, pp. 19–37. Springer, Heidelberg (2012), `http://eprint.iacr.org/2012/240`

[12] Gentry, C., Halevi, S., Smart, N.P.: Better bootstrapping in fully homomorphic encryption. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 1–16. Springer, Heidelberg (2012)

[13] Gentry, C., Halevi, S., Smart, N.P.: Fully homomorphic encryption with polylog overhead. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 465–482. Springer, Heidelberg (2012)

[14] Gentry, C., Halevi, S., Smart, N.P.: Homomorphic evaluation of the AES circuit. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 850–867. Springer, Heidelberg (2012)

[15] Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. J. ACM (2013); To appear Preliminary version In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010)

[16] Lyubashevsky, V., Peikert, C., Regev, O.: A toolkit for ring-LWE cryptography. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 35–54. Springer, Heidelberg (2013)

[17] Smart, N.P., Vercauteren, F.: Fully homomorphic SIMD operations. Cryptology ePrint Archive, Report 2011/133 (2011), `http://eprint.iacr.org/`

[18] van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (2010)

Hardness of SIS and LWE with Small Parameters

Daniele Micciancio[1,⋆] and Chris Peikert[2,⋆⋆]

[1] University of California, San Diego
[2] School of Computer Science, Georgia Institute of Technology

Abstract. The Short Integer Solution (SIS) and Learning With Errors (LWE) problems are the foundations for countless applications in lattice-based cryptography, and are provably as hard as approximate lattice problems in the worst case. An important question from both a practical and theoretical perspective is how small their parameters can be made, while preserving their hardness.

We prove two main results on SIS and LWE with small parameters. For SIS, we show that the problem retains its hardness for moduli $q \geq \beta \cdot n^{\delta}$ for any constant $\delta > 0$, where β is the bound on the Euclidean norm of the solution. This improves upon prior results which required $q > \beta \cdot \sqrt{n \log n}$, and is close to optimal since the problem is trivially easy for $q \leq \beta$. For LWE, we show that it remains hard even when the errors are small (e.g., uniformly random from $\{0, 1\}$), provided that the number of samples is small enough (e.g., linear in the dimension n of the LWE secret). Prior results required the errors to have magnitude at least $\sqrt{n}$ and to come from a Gaussian-like distribution.

Keywords: Lattice cryptography, Computational hardness, SIS, LWE.

1 Introduction

In modern lattice-based cryptography, two average-case computational problems serve as the foundation of almost all cryptographic schemes: Short Integer Solution (SIS), and Learning With Errors (LWE). The SIS problem dates back to Ajtai's pioneering work [1], and is defined as follows. Let n and q be integers, where n is the primary security parameter and usually $q = \text{poly}(n)$, and let $\beta > 0$. Given a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ for some $m = \text{poly}(n)$, the goal is to find a nonzero integer vector $\boldsymbol{z} \in \mathbb{Z}^m$ such that $\mathbf{A}\boldsymbol{z} = \mathbf{0} \bmod q$ and $\|\boldsymbol{z}\| \leq \beta$ (where $\|\cdot\|$ denotes Euclidean norm). Observe that β should be set large enough to ensure that a solution exists (e.g., $\beta > \sqrt{n \log q}$ suffices), but

⋆ Supported by the TYPED project under the DARPA PROCEED program and the National Science Foundation under grant CNS-1117936.

⋆⋆ Supported by the National Science Foundation under CAREER Award CCF-1054495, by DARPA under agreement number FA8750-11-C-0096, and by the Alfred P. Sloan Foundation.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 21–39, 2013.

that $\beta \geq q$ makes the problem trivially easy to solve. Ajtai showed that for appropriate parameters, SIS enjoys a remarkable worst-case/average-case hardness property: solving it *on the average* (with any noticeable probability) is at least as hard as approximating several lattice problems on n-dimensional lattices *in the worst case*, to within $\text{poly}(n)$ factors.

The LWE problem was introduced in the celebrated work of Regev [26], and has the same parameters n and q, along with a "noise rate" $\alpha \in (0, 1)$. The problem (in its search form) is to find a secret vector $\boldsymbol{s} \in \mathbb{Z}_q^n$, given a "noisy" random linear system $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\boldsymbol{b} = \mathbf{A}^T \boldsymbol{s} + \boldsymbol{e} \bmod q$, where $\mathbf{A}$ is uniformly random and the entries of $\boldsymbol{e}$ are i.i.d. from a Gaussian-like distribution with standard deviation roughly αq. Regev showed that as long as $\alpha q \geq 2\sqrt{n}$, solving LWE on the average (with noticeable probability) is at least as hard as approximating lattice problems in the worst case to within $\tilde{O}(n/\alpha)$ factors using a *quantum* algorithm. Subsequently, similar results under *classical* (i.e., non quantum) hardness assumptions were proved in [23,7].

A significant line of research has been devoted to improving the tightness of worst-case/average-case connections for lattice problems. For SIS, a series of works [1,8,16,21,13] gave progressively better parameters that guarantee hardness, and smaller approximation factors for the underlying lattice problems. The state of the art (from [13], building upon techniques introduced in [21]) shows that for $q \geq \beta \cdot \omega(\sqrt{n \log n})$, finding a SIS solution with norm bounded by β is as hard as approximating worst-case lattice problems to within $\tilde{O}(\beta\sqrt{n})$ factors. (The parameter m does not play any significant role in the hardness results, and can be any polynomial in n.) For LWE, Regev's initial result remains the tightest, and the requirement that $q \geq \sqrt{n}/\alpha$ (i.e., that the errors have magnitude at least $\sqrt{n}$) is in some sense optimal: a clever algorithm due to Arora and Ge [2] solves LWE in time $2^{\tilde{O}(\alpha q)^2}$, so a proof of hardness for substantially smaller errors would imply a subexponential time (quantum) algorithm for approximate lattice problems, which would be a major breakthrough. Interestingly, the current modulus bound for LWE is in some sense better than the one for SIS by a $\tilde{\Omega}(\sqrt{n})$ factor: there are applications of LWE for $1/\alpha = \tilde{O}(1)$ and hence $q = \tilde{O}(\sqrt{n})$, whereas SIS is only useful for $\beta \geq \sqrt{n}$, and therefore requires $q \geq n$ according to the state-of-the-art reductions.

Further investigating the smallest parameters for which SIS and LWE remain provably hard is important from both a practical and theoretical perspective. On the practical side, improvements may lead to smaller cryptographic keys without compromising the theoretical security guarantees, or may provide greater confidence in more practical parameter settings that so far lack provable hardness. Also, proving the hardness of LWE for non-Gaussian error distributions (e.g., uniform over a small set) makes applications easier to implement. Theoretically, improvements may eventually shed light on related problems like Learning Parity with Noise (LPN), which can be seen as a special case of LWE for modulus $q = 2$, and which is widely used in coding-based cryptography, but which has no known proof of hardness based on worst-case complexity assumptions.

1.1 Our Results

We prove two complementary results on the hardness of SIS and LWE with small parameters. For SIS, we show that the problem retains its hardness for moduli q nearly equal to the solution bound β. For LWE, we show that it remains hard even when the errors are small (e.g., uniformly random from $\{0, 1\}$), provided that the number m of noisy equations is small enough. This qualification is necessary in light of the Arora-Ge attack [2], which for large enough m can solve LWE with binary errors in polynomial time. Details follow.

SIS with small modulus. Our first theorem says that SIS retains its hardness with a modulus as small as $q \geq \beta \cdot n^\delta$, for any $\delta > 0$. Recall that the best previous reduction [13] required $q \geq \beta \cdot \omega(\sqrt{n \log n})$, and that SIS becomes trivially easy for $q \leq \beta$, so the q obtained by our proof is close to optimal. It also essentially closes the gap between LWE and SIS, in terms of how small a useful modulus can be. More precisely, the following is a special case of our main SIS hardness theorem; see Section 2 for full details.

Theorem 1 (Corollary of Theorem 4). *Let n and $m = \mathrm{poly}(n)$ be integers, let $\beta \geq \beta_\infty \geq 1$ be reals, let $Z = \{\boldsymbol{z} \in \mathbb{Z}^m : \|\boldsymbol{z}\|_2 \leq \beta \text{ and } \|\boldsymbol{z}\|_\infty \leq \beta_\infty\}$, and let $q \geq \beta \cdot n^\delta$ for some constant $\delta > 0$. Then solving (on the average, with non-negligible probability)* SIS *with parameters n, m, q and solution set $Z \setminus \{\mathbf{0}\}$ is at least as hard as approximating lattice problems in the worst case on n-dimensional lattices to within $\gamma = \max\{1, \beta \cdot \beta_\infty / q\} \cdot \tilde{O}(\beta\sqrt{n})$ factors.*

Of course, the ℓ_∞ bound on the SIS solutions can be easily removed simply setting $\beta_\infty = \beta$, so that $\|\boldsymbol{z}\|_\infty \leq \|\boldsymbol{z}\|_2 \leq \beta$ automatically holds true. We include an explicit ℓ_∞ bound $\beta_\infty \leq \beta$ in order to obtain more precise hardness results, based on potentially smaller worst-case approximation factors γ. We point out that the bound β_∞ and the associated extra term $\max\{1, \beta \cdot \beta_\infty / q\}$ in the worst-case approximation factor is not present in previous results. Notice that this term can be as small as 1 (if we take $q \geq \beta \cdot \beta_\infty$, and in particular if $\beta_\infty \leq n^\delta$), and as large as β / n^δ (if $\beta_\infty = \beta$). This may be seen as the first theoretical evidence that, at least when using a small modulus q, restricting the ℓ_∞ norm of the solutions may make the SIS problem qualitatively harder than just restricting the ℓ_2 norm. There is already significant empirical evidence for this belief: the most practically efficient attacks on SIS, which use lattice basis reduction (e.g., [12,9]), only find solutions with bounded ℓ_2 norm, whereas combinatorial attacks such as [5,27] (see also [22]) or theoretical lattice attacks [10] that can guarantee an ℓ_∞ bound are much more costly in practice, and also require exponential space. Finally, we mention that setting $\beta_\infty \ll \beta$ is very natural in the usual formulations of one-way and collision-resistant hash functions based on SIS, where collisions correspond (for example) to vectors in $\{-1, 0, 1\}^m$, and therefore have ℓ_∞ bound $\beta_\infty = 1$, but ℓ_2 bound $\beta = \sqrt{m}$. Similar gaps between β_∞ and β can easily be enforced in other applications, e.g., digital signatures [13].

LWE with small errors. In the case of LWE, we prove a general theorem offering a trade-off among several different parameters, including the size of the errors, the

dimension and number of samples in the LWE problem, and the dimension of the underlying worst-case lattice problems. Here we mention just one instantiation for the case of prime modulus and uniformly distributed *binary* (i.e., 0-1) errors, and refer the reader to Section 3 and Theorem 6 for the more general statement and a discussion of the parameters.

Theorem 2 (Corollary of Theorem 6). *For any integers n and $m = n \cdot (1 + \Omega(1/\log n))$, and all sufficiently large polynomially bounded (prime) moduli $q \geq n^{O(1)}$,*[1] *solving* LWE *with parameters n, m, q and independent uniformly random binary errors (i.e., in $\{0,1\}$) is at least as hard as approximating lattice problems in the worst case on $\Theta(n/\log n)$-dimensional lattices within a factor $\gamma = \tilde{O}(\sqrt{n} \cdot q)$.*

We remark that our results (see Theorem 6 and discussion following it) apply to many other settings, including error vectors $\boldsymbol{e} \in X$ chosen from any (sufficiently large) subset $X \subseteq \{0,1\}^m$ of binary strings, as well as error vectors with larger entries. Interestingly, our hardness result for LWE with very small errors relies on the worst-case hardness of lattice problems in dimension $n' = O(n/\log n)$, which is smaller than (but still quasi-linear in) the dimension n of the LWE problem; however, this is needed only when considering very small error vectors. Theorem 6 also shows that if $\boldsymbol{e}$ is chosen uniformly at random with entries bounded by n^ϵ (which is still much smaller than $\sqrt{n}$), then the dimension of the underlying worst-case lattice problems (and the number $m - n$ of extra samples, beyond the LWE dimension n) can be linear in n.

The restriction that the number of LWE samples $m = O(n)$ be linear in the dimension of the secret can also be relaxed slightly. But some restriction is necessary, because LWE with small errors can be solved in polynomial time when given an arbitrarily large polynomial number of samples. We focus on linear $m = O(n)$ because this is enough for most (but not all) applications in lattice cryptography, including identity-based encryption and fully homomorphic encryption, when the parameters are set appropriately. (The one exception that we know of is the security proof for pseudorandom functions [3].)

We remark that state-of-the-art reductions from worst-case lattice problems [21,13,26,7] are not tight enough to provide useful estimates on the concrete security of lattice cryptography, and they are best interpreted as qualitative results showing that there is no structural flaw in cryptographic constructions/instantiations. Still, these reductions are very valuable because even small changes in parameters can easily lead to new avenues of attack, like the polynomial time algorithm of [2] to LWE with binary errors. Likewise, our work also

[1] Making the asymptotic notation explicit, the theorem asserts that for any constant $c_1 > 0$ there is a constant $c_2 > 0$ such that if $m = n \cdot (1 + c_1/\log n)$ and $q \geq n^{c_2}$, then LWE with binary errors is hard. Notice that this dependency of the modulus q on the number of samples m is necessary for the theorem to be nontrivial. In fact, the LWE function with binary errors maps $\log q^n + m$ input bits to $\log q^m$ output bits. When $m = n \cdot (1 + c_1/\log n)$ and $q = n^{c_2}$ the LWE function stretches the input by $\log q^m - (\log q^n + m) = (c_1 c_2 - 1 - o(1))n$ bits, and for the theorem to be useful/nontrivial (and give, e.g., a pseudorandom generator) one needs $c_1 c_2 > 1$.

provides results that are primarily qualitative, showing that SIS and LWE are asymptotically secure even when $q \approx \sqrt{n}$ and the errors are small (provided the number of samples is suitably restricted). The evaluation of the concrete level of security/efficiency offered by SIS and LWE for specific small parameter values still requires careful cryptanalysis, and consideration of the best known attacks. (See, for example, [22,15,9].)

1.2 Techniques and Comparison to Related Work

Our results for SIS and LWE are technically disjoint, and all they have in common is the goal of proving hardness results for smaller values of the parameters. So, we describe our technical contributions in the analysis of these two problems separately.

SIS with small modulus. For SIS, as a warm-up, we first give a proof for a special case of the problem where the input is restricted to vectors of a special form (e.g., binary vectors). For this restricted version of SIS, we are able to give a self-reduction (from SIS to SIS) which reduces the size of the modulus. So, we can rely on previous worst-case to average-case reductions for SIS as "black boxes," resulting in an extremely simple proof. However, this simple self-reduction has some drawbacks. Beside the undesirable restriction on the SIS inputs, our reduction is rather loose with respect to the underlying worst-case lattice approximation problem: in order to establish the hardness of SIS with small moduli q (and restricted inputs), one needs to assume the worst-case hardness of lattice problems for rather large polynomial approximation factors. (By contrast, previous hardness results for larger moduli [21,13] only assumed hardness for quasi-linear approximation factors.) We address both drawbacks by giving a direct reduction from worst-case lattice problems to SIS with small modulus. This is our main SIS result, and it combines ideas from previous work [21,13] with two new technical ingredients:

- All previous SIS hardness proofs [1,8,16,21,13] solved worst-case lattice problems by iteratively finding (sets of linearly independent) lattice vectors of shorter and shorter length. Our first new technical ingredient (inspired by the pioneering work of Regev [26] on LWE) is the use a different intermediate problem: instead of finding progressively shorter lattice vectors, we consider the problem of sampling lattice vectors according to Gaussian-like distributions of progressively smaller widths. To the best of our knowledge, this is the first use of Gaussian lattice sampling as an intermediate worst-case problem in the study of SIS, and it appears necessary to lower the SIS modulus below n. We mention that Gaussian lattice sampling has been used before to reduce the modulus in hardness reductions for SIS [13], but still within the framework of iteratively finding short vectors (used in [13] to generate fresh Gaussian samples for the reduction), which results in larger moduli $q > n$.
- The use of Gaussian lattice sampling as an intermediate problem within the SIS hardness proof yields linear combinations of several discrete Gaussian

samples with adversarially chosen coefficients. Our second technical ingredient, used to analyze these linear combinations, is a new convolution theorem for discrete Gaussians (Theorem 3), which strengthens similar ones previously proved in [24,6]. Here again, the strength of our new convolution theorem appears necessary to obtain hardness results for SIS with modulus smaller than n.

Our new convolution theorem may be of independent interest, and might find applications in the analysis of other lattice algorithms.

LWE with small errors. We now move to our results on LWE. For this problem, the best provably hard parameters to date were those obtained in the original paper of Regev [26], which employed Gaussian errors, and required them to have (expected) magnitude at least $\sqrt{n}$. These results were believed to be optimal due to a clever algorithm of Arora and Ge [2], which solves LWE in subexponential time when the errors are asymptotically smaller than $\sqrt{n}$. The possibility of circumventing this barrier by limiting the number of LWE samples was first suggested by Micciancio and Mol [18], who gave "sample preserving" search-to-decision reductions for LWE, and asked if LWE with small uniform errors could be proved hard when the number of available samples is sufficiently small. Our results provide a first answer to this question, and employ concepts and techniques from the work of Peikert and Waters [25] (see also [4]) on *lossy* (trapdoor) functions. In brief, a lossy function family is an indistinguishable pair of function families $\mathcal{F}, \mathcal{L}$ such that functions in $\mathcal{F}$ are injective and those in $\mathcal{L}$ are lossy, in the sense that they map their common domain to much smaller sets, and therefore lose information about the input. As shown in [25], from the indistinguishability of $\mathcal{F}$ and $\mathcal{L}$, it follows that the function families $\mathcal{F}$ and $\mathcal{L}$ are both one-way.

In the full version of this paper [20] we present a generalized framework for the study of lossy function families, which does not require the functions to have trapdoors, and applies to arbitrary (not necessarily uniform) input distributions. While the techniques we use are all standard, and our definitions are minor generalizations of the ones given in [25], we believe that our framework provides a conceptual simplification of previous work, relating the relatively new notion of lossy functions to the classic security definitions of second-preimage resistance and uninvertibility.

The lossy function framework is used to prove the hardness of LWE with small uniform errors and (necessarily) a small number of samples. Specifically, we use the standard LWE problem (with large Gaussian errors) to set up a lossy function family $\mathcal{F}, \mathcal{L}$. (Similar families with trapdoors were constructed in [25,4], but not for the parameterizations required to obtain interesting hardness results for LWE.) The indistinguishability of $\mathcal{F}$ and $\mathcal{L}$ follows directly from the hardness of the underlying LWE problem. The new hardness result for LWE (with small errors) is equivalent to the one-wayness of $\mathcal{F}$, and is proved by a relatively standard analysis of the second-preimage resistance and uninvertibility of certain subset-sum functions associated to $\mathcal{L}$.

Our results, as well as previous work based on lossyness arguments, relies to some extent on the entropy in the secret vector. For simplicity, we specialize our analysis to the uniform input distribution (over arbitrary sets of short vectors), which makes counting arguments and entrody arguments essentially the same. Our results do generalize without much difficulty to other non-uniform distributions having sufficient min-entropy (unpredictability) over input sets with small diameter.

Comparison to related work. In a recent and independent work Döttling and Müller-Quade [11] also used a lossyness argument to prove new hardness results for LWE. Beside the use of a similar high level lossyness argument, the technical details of the proof are quite different, and the results are different as well. Just like our work, [11] proves hardness for uniformly distributed errors, and requires the number of m of samples to be fixed in advance. However, [11] requires the noise bound to be bigger than $\sqrt{n}$ (in fact, at least $m\sqrt{n}$, where m is the number of samples,) while in our work the errors can be smaller than $\sqrt{n}$, or even binary. On the other hand, when the magnitude of the errors is large $\sqrt{n}m \gg \sqrt{n}$, [11] allows the number of samples $m = n^{O(1)}$ to be an arbitrary large polynomial, while here we require it to be linear $m = \Theta(n)$. Another (more technical) difference between the two results is that our proof is based on a fairly general counting argument that allows error distributions that are uniform over arbitrary sets (of short vectors), while [11] only applies to uniform distributions over more structured sets, e.g., all vectors within a regularly shaped convex region of space.

1.3 Notation and Background

We briefly recall the (mostly standard) notation and background used in this paper, and refer the reader to the full version [20] for a more detailed account. We use standard asymptotic notation $O, \tilde{O}, \Omega, o, \omega$, and write ω_n as an abbreviation for $\omega(\sqrt{\log n})$. We write $[\mathcal{X}]$ to denote the support of a probability distribution $\mathcal{X}$, and $\mathcal{U}(X)$ for the uniform distribution over a set X. We assume familiarity with the notion of *computational indistinguishability*, and standard notions of security for function families, like *collision resistance*, *one-wayness*, *uninvertibility*, *second-preimage resistance*, and *pseudorandomness*. A *lossy function family* (slightly generalizing the concept of lossy trapdoor functions introduced in [25]) is a pair of computationally indistinguishable probability distributions $\mathcal{L}, \mathcal{F}$ over (descriptions of) functions $F \subseteq X \to Y$, such that $\mathcal{L}$ is an uninvertible function family, and $\mathcal{F}$ is a second preimage resistant function family, both with respect to some input distribution $\mathcal{X}$ over the domain X. It easily follows from the definition that both $\mathcal{F}$ and $\mathcal{L}$ are one-way function families with respect to input $\mathcal{X}$. A function family $\mathcal{L}$ is uninvertible with respect to the uniform distribution $\mathcal{X} = \mathcal{U}(X)$ if (and only if) the expectation $\mathbb{E}_{f \leftarrow \mathcal{L}}[|f(X)|]/|X|$ is negligible. Moreover, if $\mathcal{F} : X \to Y$ is uninvertible (with respect to input distribution $\mathcal{X}$) and $\mathcal{G} : Y \to Z$ is an arbitrary function family, then the composition $\mathcal{G} \circ \mathcal{F} : X \to Z$

(defined in the obvious way, as the result of sampling two functions from $\mathcal{G}$ and $\mathcal{F}$ independently at random, and taking their function composition) is also uninvertible on input $\mathcal{X}$.

An n-dimensional (full rank) *lattice* is the set Λ of integer combinations of n linearly independent (basis) vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \mathbb{R}^n$. The (decision version of the) Shortest Vector Problem, GapSVP_γ asks to approximate within a factor γ the (Euclidean) length of the shortest nonzero vector in the lattice generated by an input basis $\mathbf{B} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n]$. The Shortest Independent Vectors Problem SIVP_γ asks to find n linearly independent lattice vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ of length $\max_i \|\boldsymbol{v}_i\| \leq \gamma\lambda_n$ within a factor γ from the optimum value λ_n. The *Gram-Schmidt* minimum of a lattice Λ is $\tilde{bl}(\Lambda) = \min_{\mathbf{B}} \|\widetilde{\mathbf{B}}\|$, where $\widetilde{\boldsymbol{b}}_i$ is projection of $\boldsymbol{b}_i$ orthogonal to $\mathrm{span}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{i-1})$, and $\mathbf{B} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n]$ ranges over all possible bases of Λ.

The *Gaussian function* $\rho_s(\boldsymbol{x}) = \exp(-\pi\|\boldsymbol{x}\|^2/s^2)$ defines a *discrete Gaussian distribution* $D_{\Lambda+\boldsymbol{c},s}$ over a lattice coset $\Lambda + \boldsymbol{c}$, which samples each element $\boldsymbol{x} \in \Lambda + \boldsymbol{c}$ with probability $\rho_s(\boldsymbol{x})/\rho_s(\Lambda + \boldsymbol{c})$. For any (typically negligible) $\epsilon > 0$, the *smoothing parameter* $\eta_\epsilon(\Lambda)$ [21] is the smallest $s > 0$ such that $\rho_{1/s}(\Lambda^* \setminus \{\mathbf{0}\}) \leq \epsilon$, where $\Lambda^* = \{\boldsymbol{x} : \forall \boldsymbol{y} \in \Lambda, \langle \boldsymbol{x}, \boldsymbol{y} \rangle \in \mathbb{Z}\}$ is the *dual lattice*. The tensor product of two lattices with bases $\mathbf{B}, \mathbf{B}'$ is the lattice with a basis $\mathbf{B} \otimes \mathbf{B}'$ obtained replacing each entry $b_{i,j}$ of $\mathbf{B}$ with the block $b_{i,j} \cdot \mathbf{B}'$, and it satisfies $\eta(\Lambda_1 \otimes \Lambda_2) \leq \tilde{bl}(\Lambda_1) \cdot \eta(\Lambda_2)$. Sampling $D_{\Lambda,\sigma}$ for some $\sigma \geq 2\eta(\Lambda)$ allows to solve SIVP_γ within a factor $\gamma = \tilde{O}(\sigma\sqrt{n})$.

The *Short Integer Solution* function family $\mathrm{SIS}(m, n, q, X)$ is the set of all functions $f_{\mathbf{A}}(\boldsymbol{x}) = \mathbf{A}\boldsymbol{x} \bmod q$ indexed by $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ with domain $X \subseteq \mathbb{Z}^m$ and range $Y = \mathbb{Z}_q^n$. The *Learning With Errors* function family $\mathrm{LWE}(m, n, q, X)$, is the set of all functions $g_{\mathbf{A}}(\boldsymbol{s}, \boldsymbol{x}) = \mathbf{A}^T\boldsymbol{s} + \boldsymbol{x} \bmod q$ indexed by $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ with domain $\mathbb{Z}_q^n \times X$ and range $Y = \mathbb{Z}_q^m$. We sometimes write $\mathrm{SIS}(m, n, q, \beta)$ for some real $\beta > 0$, to refer to the problem of finding a nonzero vector $\boldsymbol{z} \in \mathbb{Z}^m$ of length at most $\|\boldsymbol{z}\| \leq \beta$ such that $f_{\mathbf{A}}(\boldsymbol{z}) = \mathbf{0}$. The SIS input is usually chosen according to either the uniform distribution $\mathcal{U}(X)$ over the set $X = \{0, \ldots, s-1\}^m$ or $X = \{-s, \ldots, 0, \ldots, s\}^m$, or the discrete Gaussian distribution $D^m_{\mathbb{Z},s}$. For the LWE problem, the input is usually chosen according to distribution $\mathcal{U}(\mathbb{Z}_q^n) \times \mathcal{X}$, where $\mathcal{X}$ is one of the SIS input distributions. This makes the $\mathrm{LWE}(m, n, q)$ and $\mathrm{SIS}(m, m-n, q)$ function families equivalent via a lattice duality argument [17,18]]. The SIS and LWE functions can be shown to be collision resistant, one-way, uninvertible, pseudorandom, etc., for appropriate parameters and input distributions, based on the assumption that SIVP_γ and/or GapSVP_γ are (quantum) hard in the worst case [26,23,7].

2 Hardness of SIS with Small Modulus

As a warm-up, we first give a simplified proof that the $\mathrm{SIS}(m, n, q, \beta)$ function family is collision resistant for moduli q as small as $n^{1/2+\delta}$, by a reduction between SIS problems with different parameters. Previous hardness results based on worst-case lattice assumptions require the modulus q to be at least

$\beta \cdot \omega(\sqrt{n \log n})$ [13, Theorem 9.2], and $\beta \geq \sqrt{n \log q}$ is needed to guarantee that a nontrivial solution exists. For such parameters, SIS is collision resistant assuming the hardness of approximating worst-case lattice problems to within $\approx \beta\sqrt{n}$ factors.

The intuition behind our proof for smaller moduli is easily explained. We reduce SIS with modulus q^c and solution bound β^c (for any constant integer $c \geq 1$) to SIS with modulus q and bound β. Then as long as $(q/\beta)^c \geq \omega(\sqrt{n \log n})$, the former problem enjoys worst-case hardness, hence so does the latter. Thus we can take $q = \beta \cdot n^\delta$ for any constant $\delta > 0$, and $c > 1/(2\delta)$. Notice, however, that the underlying approximation factor for worst-case lattice problems is $\approx \beta^c\sqrt{n} \geq n^{1/2+1/(4\delta)}$, which, while still polynomial, degrades severely as δ approaches 0. In the next subsection we give a direct reduction from worst-case lattice problems to SIS with a small modulus, which does not have this drawback.

The above discussion is formalized in the following proposition. For technical reasons, we prove that $\mathrm{SIS}(m, n, q, X)$ is collision resistant assuming that the domain X has the property that all SIS solutions $\boldsymbol{z} \in (X - X) \setminus \{\mathbf{0}\}$ satisfy $\gcd(\boldsymbol{z}, q) = 1$. This restriction is satisfied in many (but not all) common settings, e.g., when $q > \beta$ is prime, or when $X \subseteq \{0,1\}^m$ is a set of binary vectors. For simplicity, we describe the reduction assuming an SIS oracle that finds collisions with overwhelming probability. The reduction can be easily adapted to oracles that solve SIS with only nonnegligible probability using standard repetition/amplification techniques.

Proposition 1. *Let n, q, m, β and $X \subseteq \mathbb{Z}^m$ be such that $\gcd(\boldsymbol{x} - \boldsymbol{x}', q) = 1$ and $\|\boldsymbol{x} - \boldsymbol{x}'\| \leq \beta$ for any distinct $\boldsymbol{x}, \boldsymbol{x}' \in X$. For any positive integer c, there is a deterministic reduction from collision-finding for $\mathrm{SIS}(m^c, n, q^c, \beta^c)$ to collision-finding for $\mathrm{SIS}(m, n, q, X)$ (in both cases, with overwhelming advantage). The reduction runs in time polynomial in its input size, and makes fewer than m^c calls to its oracle.*

Proof. Let $\mathcal{A}$ be an efficient algorithm that finds a collision for $\mathrm{SIS}(m, n, q, X)$ with overwhelming advantage. We use it to find a nonzero solution for $\mathrm{SIS}(m^c, n, q^c, \beta^c)$. Let $\mathbf{A} \in \mathbb{Z}_{q^c}^{n \times m^c}$ be an input SIS instance. Partition the columns of $\mathbf{A}$ into m^{c-1} blocks $\mathbf{A}_i \in \mathbb{Z}_{q^c}^{n \times m}$, and for each one, invoke $\mathcal{A}$ to find a collision modulo q, i.e., a pair of distinct vectors $\boldsymbol{x}_i, \boldsymbol{x}'_i \in X$ such that $\mathbf{A}_i \boldsymbol{z}_i = \mathbf{0} \bmod q$, where $\boldsymbol{z}_i = \boldsymbol{x}_i - \boldsymbol{x}'_i$ and $\|\boldsymbol{z}_i\| \leq \beta$.

For each i, since $\gcd(\boldsymbol{z}_i, q) = 1$ and $\mathbf{A}_i \boldsymbol{z}_i = \mathbf{0} \bmod q$, the vector $\boldsymbol{a}'_i = (\mathbf{A}_i \boldsymbol{z}_i)/q \in \mathbb{Z}_{q^{c-1}}^n$ is uniformly random, even after conditioning on $\boldsymbol{z}_i$ and $\mathbf{A}_i \bmod q$. So, the matrix $\mathbf{A}' \in \mathbb{Z}_{q^{c-1}}^{n \times m^{c-1}}$ made up of all these columns is uniformly random. By induction on c, using $\mathcal{A}$ we can find a nonzero solution $\boldsymbol{z}' \in \mathbb{Z}^{m^{c-1}}$ such that $\mathbf{A}' \boldsymbol{z}' = \mathbf{0} \bmod q^{c-1}$ and $\|\boldsymbol{z}'\| \leq \beta^{c-1}$. Then it is easy to verify that a nonzero solution for the original instance $\mathbf{A}$ is given by $\boldsymbol{z} = (z'_1 \cdot \boldsymbol{z}_1, \ldots, z'_{m^{c-1}} \cdot \boldsymbol{z}_{m^{c-1}}) \in \mathbb{Z}^{m^c}$, and that $\|\boldsymbol{z}\| \leq \|\boldsymbol{z}'\| \cdot \max_i \|\boldsymbol{z}_i\| \leq \beta^c$. Finally, the total number of calls to $\mathcal{A}$ is $\sum_{i=0}^{c-1} m^i < m^c$, as claimed. □

Direct Reduction. As mentioned above, the large worst-case approximation factor associated with the use of Proposition 1 is undesirable, as is (to a lesser extent) the restriction that $\gcd(X - X, q) = 1$. To eliminate these drawbacks, we next give a direct proof that SIS is collision resistant for small q, based on the assumed hardness of worst-case lattice problems. The underlying approximation factor for these problems can be as small as $\tilde{O}(\beta\sqrt{n})$, which matches the best known factors obtained by previous proofs (which require a larger modulus q). Our new proof combines ideas from [21,13] and Proposition 1, as well as a new convolution theorem for discrete Gaussians which strengthens similar ones previously proved in [24,6].

Our proof of the convolution theorem is substantially different and, we believe, technically simpler than the prior ones. In particular, it handles the sum of many Gaussian samples all at once, whereas previous proofs used induction from a base case of two samples. With the inductive approach, it is technically complex to verify that all the intermediate Gaussian parameters (which involve harmonic means) satisfy the hypotheses. Moreover, the intermediate parameters can depend on the order in which the samples are added in the induction, leading to unnecessarily strong hypotheses on the original parameters. Due to space constraints, the proof of the convolution theorem is given in the full version [20].

Theorem 3. *Let Λ be an n-dimensional lattice, $\boldsymbol{z} \in \mathbb{Z}^m$ a nonzero integer vector, $s_i \geq \sqrt{2}\|\boldsymbol{z}\|_\infty \cdot \eta(\Lambda)$, and $\Lambda + \boldsymbol{c}_i$ arbitrary cosets of Λ for $i = 1, \ldots, m$. Let $\boldsymbol{y}_i$ be independent vectors with distributions $D_{\Lambda+\boldsymbol{c}_i, s_i}$, respectively. Then the distribution of $\boldsymbol{y} = \sum_i z_i \boldsymbol{y}_i$ is statistically close to $D_{Y,s}$, where $Y = \gcd(\boldsymbol{z})\Lambda + \boldsymbol{c}$, $\boldsymbol{c} = \sum_i z_i \boldsymbol{c}_i$, and $s = \sqrt{\sum_i (z_i s_i)^2}$. In particular, if $\gcd(\boldsymbol{z}) = 1$ and $\sum_i z_i \boldsymbol{c}_i \in \Lambda$, then $\boldsymbol{y}$ is distributed statistically close to $D_{\Lambda,s}$.*

The convolution theorem implies the following simple but useful lemma, which shows how to convert samples having a broad range of parameters into ones having parameters in a desired narrow range.

Lemma 1. *There is an efficient algorithm which, given a basis $\mathbf{B}$ of some lattice Λ, some $R \geq \sqrt{2}$, and access to samples $(\boldsymbol{y}_i, s_i)$ where each $s_i \in [\sqrt{2}, R] \cdot \eta(\Lambda)$ is* arbitrary *and each $\boldsymbol{y}_i$ has distribution D_{Λ,s_i}, with overwhelming probability outputs a pair $(\boldsymbol{y}, s)$ such that $s \in [R, \sqrt{2}R] \cdot \eta(\Lambda)$ and $\boldsymbol{y}$ has distribution statistically close to $D_{\Lambda,s}$.*

Proof. Let $\omega_n = \omega(\sqrt{\log n})$ satisfy $\omega_n \leq \sqrt{n}$. The algorithm draws $2n^2$ input samples, and works as follows: if at least n^2 of the samples have parameters $s_i \leq R \cdot \eta(\Lambda)/(\sqrt{n} \cdot \omega_n)$, then with overwhelming probability they all have lengths bounded by $R \cdot \eta(\Lambda)/\omega_n$ and they include n linearly independent vectors. Using such vectors we can construct a basis $\mathbf{S}$ such that $\|\widetilde{\mathbf{S}}\| \leq R \cdot \eta(\Lambda)/\omega_n$, and with the sampling algorithm of [13, Theorem 4.1] we can generate samples having parameter $R \cdot \eta(\Lambda)$. Otherwise, at least n^2 of the samples $(\boldsymbol{y}_i, s_i)$ have parameters $s_i \geq \max\{R/n, \sqrt{2}\} \cdot \eta(\Lambda)$. Then by summing an appropriate subset of those $\boldsymbol{y}_i$, by the convolution theorem we can obtain a sample having parameter in the desired range. □

The next lemma is the heart of our reduction. The novel part, corresponding to the properties described in the second item, is a way of using a collision-finding oracle to reduce the Gaussian width of samples drawn from a lattice. The first item corresponds to the guarantees provided by previous reductions.

Lemma 2. *Let m, n be integers, $S = \{\boldsymbol{z} \in \mathbb{Z}^m \setminus \{\mathbf{0}\} \mid \|\boldsymbol{z}\| \leq \beta \wedge \|\boldsymbol{z}\|_\infty \leq \beta_\infty\}$ for some real $\beta \geq \beta_\infty > 0$, and q an integer modulus with at most $\mathrm{poly}(n)$ integer divisors less than β_∞. There is a probabilistic polynomial time reduction that, on input any basis $\mathbf{B}$ of a lattice Λ and sufficiently many samples $(\boldsymbol{y}_i, s_i)$, where each $s_i \geq \sqrt{2}q \cdot \eta(\Lambda)$ may be arbitrary and each $\boldsymbol{y}_i$ has distribution D_{Λ,s_i}, and given access to an $\mathrm{SIS}(m, n, q, S)$ oracle (that finds a solution $\boldsymbol{z} \in S$ with nonnegligible probability) outputs (with overwhelming probability) a sample $(\boldsymbol{y}, s)$ where $\min s_i/q \leq s \leq (\beta/q) \cdot \max s_i$, and $\boldsymbol{y} \in \Lambda$ is such that:*

- $\mathbb{E}[\|\boldsymbol{y}\|] \leq (\beta\sqrt{n}/q) \cdot \max s_i$, *and for any fixed subspace $H \subset \mathbb{R}^n$ of dimension at most $n-1$, we have $\Pr[\boldsymbol{y} \notin H] \geq 1/10$.*
- *Moreover, if each $s_i \geq \sqrt{2}\beta_\infty q \cdot \eta(\Lambda)$, then the distribution of $\boldsymbol{y}$ is statistically close to $D_{\Lambda,s}$*

Proof. Let $\mathcal{A}$ be the SIS oracle. Without loss of generality, we can assume that whenever $\mathcal{A}$ outputs a valid solution $\boldsymbol{z} \in S$, we have that $\gcd(\boldsymbol{z})$ divides q. This is because for any integer vector $\boldsymbol{z}$, if $\mathbf{A}\boldsymbol{z} = \mathbf{0} \bmod q$ then also $\mathbf{A}((g/d)\boldsymbol{z}) = \mathbf{0} \bmod q$, where $d = \gcd(\boldsymbol{z})$ and $g = \gcd(d, q)$. Moreover, $(g/d)\boldsymbol{z} \in S$ holds true and $\gcd((g/d)\boldsymbol{z}) = \gcd(\boldsymbol{z}, q)$ divides q. Let d be such that $\mathcal{A}$ outputs, with nonnegligible probability, a valid solution $\boldsymbol{z}$ satisfying $\gcd(\boldsymbol{z}) = d$. Such a d exists because $\gcd(\boldsymbol{z})$ is bounded by β_∞ and divides q, so by assumption there are only polynomially many possible values of d. Let $q' = q/d$, which is an integer. By increasing m and using standard amplification techniques, we can make the probability that $\mathcal{A}$ outputs such a solution (satisfying $\boldsymbol{z} \in S$, $\mathbf{A}\boldsymbol{z} = \mathbf{0} \pmod q$ and $\gcd(\boldsymbol{z}) = d$) exponentially close to 1.

Let $(\boldsymbol{y}_i, s_i)$ for $i = 1, \ldots, m$ be input samples, where $\boldsymbol{y}_i$ has distribution D_{Λ,s_i}. Write each $\boldsymbol{y}_i$ as $\boldsymbol{y}_i = \mathbf{B}\boldsymbol{a}_i \bmod q'\Lambda$ for $\boldsymbol{a}_i \in \mathbb{Z}_{q'}^n$. Since $s_i \geq q'\eta(\Lambda)$ the distribution of $\boldsymbol{a}_i$ is statistically close to uniform over $\mathbb{Z}_{q'}^n$. Let $\mathbf{A} = [\boldsymbol{a}_1 \mid \cdots \mid \boldsymbol{a}_m] \in \mathbb{Z}_q^{n \times m}$, and choose $\mathbf{A}' \in \mathbb{Z}_d^{n \times m}$ uniformly at random. Since $\mathbf{A}$ is statistically close to uniform over $\mathbb{Z}_{q'}^{n \times m}$, the matrix $\mathbf{A} + q'\mathbf{A}'$ is statistically close to uniform over $\mathbb{Z}_q^{n \times m}$. Call the oracle $\mathcal{A}$ on input $\mathbf{A} + q'\mathbf{A}'$, and obtain (with overwhelming probability) a nonzero $\boldsymbol{z} \in S$ with $\gcd(\boldsymbol{z}) = d$, $\|\boldsymbol{z}\| \leq \beta$, $\|\boldsymbol{z}\|_\infty \leq \beta_\infty$ and $(\mathbf{A} + q'\mathbf{A}')\boldsymbol{z} = \mathbf{0} \bmod q$. Notice that $q'\mathbf{A}'\boldsymbol{z} = q\mathbf{A}'(\boldsymbol{z}/d) = \mathbf{0} \bmod q$ because $(\boldsymbol{z}/d)$ is an integer vector. Therefore $\mathbf{A}\boldsymbol{z} = \mathbf{0} \bmod q$. Finally, the reduction outputs $(\boldsymbol{y}, s)$, where $\boldsymbol{y} = \sum_i z_i \boldsymbol{y}_i/q$ and $s = \sqrt{\sum_i (s_i z_i)^2}/q$. Notice that $z_i \boldsymbol{y}_i \in q\Lambda + \mathbf{B}(z_i \boldsymbol{a}_i)$ because $\gcd(\boldsymbol{z}) = d$, so $\boldsymbol{y} \in \Lambda$.

Notice that s satisfies the stated bounds because $\boldsymbol{z}$ is a nonzero integer vector. We next analyze the distribution of $\boldsymbol{y}$. For any fixed $\boldsymbol{a}_i$, the conditional distribution of each $\boldsymbol{y}_i$ is $D_{q'\Lambda + \mathbf{B}\boldsymbol{a}_i, s_i}$, where $s_i \geq \sqrt{2}\eta(q'\Lambda)$. The claim on $\mathbb{E}[\|\boldsymbol{y}\|]$ then follows from [21, Lemma 2.11 and Lemma 4.3] and Hölder's inequality. The claim on the probability that $\boldsymbol{y} \notin H$ was initially shown in the preliminary version of [21]; see also [26, Lemma 3.15].

Now assume that $s_i \geq \sqrt{2}\beta_\infty q \cdot \eta(\Lambda) \geq \sqrt{2}\|\boldsymbol{z}\|_\infty \cdot \eta(q'\Lambda)$ for all i. By Theorem 3 the distribution of $\boldsymbol{y}$ is statistically close to $D_{Y/q,s}$ where $Y = \gcd(\boldsymbol{z}) \cdot q'\Lambda + \mathbf{B}(\mathbf{A}\boldsymbol{z})$. Using $\mathbf{A}\boldsymbol{z} = \mathbf{0} \bmod q$ and $\gcd(\boldsymbol{z}) = d$, we get $Y = q\Lambda$. Therefore $\boldsymbol{y}$ has distribution statistically close to $D_{\Lambda,s}$, as claimed. □

Building on Lemma 2, our next lemma shows that for any $q \geq \beta \cdot n^{\Omega(1)}$, an SIS oracle can be used to obtain Gaussian samples of width close to $2\beta\beta_\infty \cdot \eta(\Lambda)$.

Lemma 3. *Let m, n, q, S as in Lemma 2, and also assume $q/\beta \geq n^\delta$ for some constant $\delta > 0$. There is an efficient reduction that, on input any basis $\mathbf{B}$ of an n-dimensional lattice Λ, an upper bound $\eta \geq \eta(\Lambda)$, and given access to an $\mathrm{SIS}(m, n, q, S)$ oracle (which finds a solution $\boldsymbol{z} \in S$ with nonnegligible probability), outputs (with overwhelming probability) a sample $(\boldsymbol{y}, s)$ where $\sqrt{2}\beta_\infty \cdot \eta \leq s \leq 2\beta_\infty\beta \cdot \eta$ and $\boldsymbol{y}$ has distribution statistically close to $D_{\Lambda,s}$.*

Proof. By applying the LLL basis reduction algorithm [14] to the basis $\mathbf{B}$, we can assume without loss of generality that $\|\widetilde{\mathbf{B}}\| \leq 2^n \cdot \eta(\Lambda)$. The main procedure, described below, produces samples having parameters in the range $[1, q] \cdot \sqrt{2}\beta_\infty \cdot \eta$. On these samples we run the procedure from Lemma 1 (with $R = \sqrt{2}\beta_\infty q \cdot \eta$) to obtain samples having parameters in the range $[\sqrt{2}, 2] \cdot \beta_\infty q \cdot \eta$. Finally, we invoke the reduction from Lemma 2 on those samples to obtain a sample satisfying the conditions in the Lemma statement.

The main procedure works in a sequence of phases $i = 0, 1, 2, \ldots$. In phase i, the input is a basis $\mathbf{B}_i$ of Λ, where initially $\mathbf{B}_0 = \mathbf{B}$. The basis $\mathbf{B}_i$ is used in the discrete Gaussian sampling algorithm of [13, Theorem 4.1] to produce samples $(\boldsymbol{y}, s_i)$, where $s_i = \max\{\|\widetilde{\mathbf{B}}_i\| \cdot \omega_n, \sqrt{2}\beta_\infty\eta\} \geq \sqrt{2}\beta_\infty\eta$ and $\boldsymbol{y}_i$ has distribution statistically close to D_{Λ,s_i}. Phase i either manages to produce a sample $(\boldsymbol{y}, s)$ with s in the desired range $[1, q] \cdot \sqrt{2}\beta_\infty\eta$, or it produces a new basis $\mathbf{B}_{i+1}$ for which $\|\widetilde{\mathbf{B}}_{i+1}\| \leq \|\widetilde{\mathbf{B}}_i\|/2$, which is the input to the next phase. The number of phases before termination is polynomial in n, by hypothesis on $\mathbf{B}$.

If $\|\widetilde{\mathbf{B}}_i\| \cdot \omega_n \leq \sqrt{2}q\beta_\infty\eta$, then this already gives samples with $s_i \in [1, q]\sqrt{2}\beta_\infty\eta$ in the desired range, and we can terminate the main phase. So, we may assume that $s_i = \|\widetilde{\mathbf{B}}_i\| \cdot \omega_n \geq \sqrt{2}q\beta_\infty\eta$. Each phase i proceeds in some constant $c \geq 1/\delta$ number of sub-phases $j = 1, 2, \ldots, c$, where the inputs to the first sub-phase are the samples $(\boldsymbol{y}, s_i)$ generated as described above. We recall that these samples satisfy $s_i \geq \sqrt{2}q\beta_\infty\eta$. The same will be true for the samples passed as input to all other subsequent subphases. So, each subphase receives as input samples $(\boldsymbol{y}, s)$ satisfying all the hypotheses of Lemma 2, and we can run the reduction from that lemma to generate new samples $(\boldsymbol{y}', s')$ having parameters s' bounded from above by $s_i \cdot (\beta/q)^j$, and from below by $\sqrt{2}\beta_\infty\eta$. If any of the produces samples satisfies $s' \leq q\sqrt{2}\beta_\infty\eta$, then we can terminate the main procedure with $(\boldsymbol{y}', s')$ as output. Otherwise, all samples produced during the subphase satisfy $s' > q\sqrt{2}\beta_\infty\eta$, and they can be passed as input to the next sub-phase. Notice that the total runtime of all the sub-phases is $\mathrm{poly}(n)^c$, because each invocation of the reduction from Lemma 2 relies on $\mathrm{poly}(n)$ invocations of the reduction in the previous sub-phase; this is why we need to limit the number of sub-phases to a constant c.

If phase i ends up running all its sub-phases without ever finding a sample with $s' \in [1, q]\sqrt{2}\beta_\infty\eta$, then it has produced samples whose parameters are bounded by $(\beta/q)^c \leq s_i \leq s_i/\sqrt{n}$. It uses n^2 of these samples, which with overwhelming probability have lengths all bounded by $s_i/\sqrt{n}$, and include n linearly independent vectors. It transforms those vectors into a basis $\mathbf{B}_{i+1}$ with $\|\widetilde{\mathbf{B}}_{i+1}\| \leq s_i/\sqrt{n} \leq \|\widetilde{B}\|_i\omega_n/\sqrt{n} \leq \|\widetilde{\mathbf{B}}_i\|/2$, as input to the next phase. □

We can now prove our main theorem, reducing worst-case lattice problems with $\max\{1, \beta\beta_\infty/q\} \cdot \tilde{O}(\beta\sqrt{n})$ approximation factors to SIS, when $q \geq \beta \cdot n^{\Omega(1)}$.

Theorem 4. *Let m, n be integers, $S = \{\boldsymbol{z} \in \mathbb{Z}^m \setminus \{\mathbf{0}\} \mid \|\boldsymbol{z}\| \leq \beta \wedge \|\boldsymbol{z}\|_\infty \leq \beta_\infty\}$ for some real $\beta \geq \beta_\infty > 0$, and $q \geq \beta \cdot n^{\Omega(1)}$ be an integer modulus with at most* $\mathrm{poly}(n)$ *integer divisors less than β_∞. For some $\gamma = \max\{1, \beta\beta_\infty/q\} \cdot O(\beta\sqrt{n})$, there is an efficient reduction from* $\mathrm{SIVP}_\gamma^\eta$ *(and hence also from standard* $\mathrm{SIVP}_{\gamma \cdot \omega_n}$*) on n-dimensional lattices to solving the collision-finding problem* $\mathrm{SIS}(m, n, q, S)$ *with non-negligible advantage.*

Proof. Given an input basis $\mathbf{B}$ of a lattice Λ, we can apply the LLL algorithm to obtain a 2^n-approximation to $\eta(\Lambda)$, and by scaling we can assume that $\eta(\Lambda) \in [1, 2^n]$. For $i = 1, \ldots, n$, we run the procedure described below for each candidate upper bound $\eta_i = 2^i$ on $\eta(\Lambda)$. Each call to the procedure either fails, or returns a set of linearly independent vectors in Λ whose lengths are all bounded by $(\gamma/2) \cdot \eta_i$. We return the first such obtained set (i.e., for the minimal value of i). As we show below, as long as $\eta_i \geq \eta(\Lambda)$ the procedure returns a set of vectors with overwhelming probability. Since exactly one $\eta_i \in [1, 2) \cdot \eta(\Lambda)$, our reduction solves $\mathrm{SIVP}_\gamma^\eta$ with overwhelming probability, as claimed.

The procedure invokes the reduction from Lemma 3 with $\eta = \eta_i$ to obtain samples with parameters in the range $[\sqrt{2}\beta_\infty, \sqrt{2}\beta\beta_\infty] \cdot \eta$. On these samples we run the procedure from Lemma 1 with $R = \max\{\sqrt{2}q, \sqrt{2}\beta\beta_\infty\}$ to obtain samples having parameters in the range $[R, \sqrt{2}R] \cdot \eta$. On such samples we repeatedly run (using independent samples each time) the reduction from Lemma 2. After enough runs, we obtain with overwhelming probability a set of linearly independent lattice vectors all having lengths at most $(\gamma/2) \cdot \eta$, as required. □

3 Hardness of LWE with Small Uniform Errors

In this section we prove the hardness of inverting the LWE function even when the error vectors have very small entries, provided the number of samples is sufficiently small. We proceed similarly to [25,4], by using the LWE assumption (for discrete Gaussian error) to construct a lossy family of functions with respect to a uniform distribution over small inputs. However, the parameterization we obtain is different from those in [25,4], allowing us to obtain *pseudorandomness* of LWE under *very small* (e.g., binary) inputs, for a number of LWE samples that exceeds the LWE dimension.

Our results and proofs are more naturally formulated using the SIS function family. So, we will first study the problem in terms of SIS, and then reformulate the results in terms of LWE by lattice duality [17,18]. All statements in this section are proved by relatively simple counting arguments, and the reader is referred to the full version [20] for details. We recall that the main difference between this section and Section 2, is that here we consider parameters for which the resulting functions are essentially injective, or more formally, statistically second-preimage resistant. The following lemma gives sufficient conditions that ensure this property.

Lemma 4. *For any integers m, k, q, s and set $X \subseteq [s]^m$, the function family* $\mathrm{SIS}(m, k, q)$ *is (statistically) ϵ-second preimage resistant with respect to the uniform input distribution $\mathcal{U}(X)$ for $\epsilon = |X| \cdot (s'/q)^k$, where s' is the largest factor of q smaller than s.*

Proof. Let $\boldsymbol{x} \leftarrow \mathcal{U}(X)$ and $\mathbf{A} \leftarrow \mathrm{SIS}(m, k, q)$ be chosen at random. We want to evaluate the probability that there exists an $\boldsymbol{x}' \in X \setminus \{\boldsymbol{x}\}$ such that $\mathbf{A}\boldsymbol{x} = \mathbf{A}\boldsymbol{x}'$ (mod q), or, equivalently, $\mathbf{A}(\boldsymbol{x} - \boldsymbol{x}') = \mathbf{0}$ (mod q). Fix any two distinct vectors $\boldsymbol{x}, \boldsymbol{x}' \in X$ and let $\boldsymbol{z} = \boldsymbol{x} - \boldsymbol{x}'$. The vector $\mathbf{A}\boldsymbol{z}$ (mod q) is distributed uniformly at random in $(d\mathbb{Z}/q\mathbb{Z})^k$, where $d = \gcd(q, z_1, \ldots, z_m)$. All coordinates of $\boldsymbol{z}$ are in the range $z_i \in \{-(s-1), \ldots, (s-1)\}$, and at least one of them is nonzero. Therefore, d is at most s' and $|d\mathbb{Z}_q^k| = (q/d)^k \geq (q/s')^k$. By union bound (over $\boldsymbol{x}' \in X \setminus \{\boldsymbol{x}\}$) for any $\boldsymbol{x}$, the probability that there is a second preimage $\boldsymbol{x}'$ is at most $(|X| - 1)(s'/q)^k$. □

We remark that, as shown in Section 2, even for parameter settings that do not fall within the range specified in Lemma 4, $\mathrm{SIS}(m, k, q)$ is collision resistant, and therefore also (computationally) second-preimage-resistant. This is all that is needed in the rest of this section. However, when $\mathrm{SIS}(m, k, q)$ is not statistically second-preimage resistant, the one-wayness proof that follows (see Theorem 5) is not very interesting: typically, in such settings, $\mathrm{SIS}(m, k, q)$ is also statistically uninvertible, and the one-wayness of $\mathrm{SIS}(m, k, q)$ trivially follows. So, below we focus on parameter settings covered by Lemma 4. We prove the one-wayness of $\mathcal{F} = \mathrm{SIS}(m, k, q, X)$ with respect to the uniform input distribution $\mathcal{X} = \mathcal{U}(X)$ by building a lossy function family $(\mathcal{L}, \mathcal{F}, \mathcal{X})$ where $\mathcal{L}$ is an auxiliary function family that we will prove to be uninvertible and computationally indistinguishable from $\mathcal{F}$. Due to space limitations, we only provide a sketch of the construction and analysis of the lossy function family, followed by formal statement of our main result, and description of some notable instantiations. The reader is referred to the full version [20] for more details.

Outline. The idea behind our construction and proof is easily explained. The functions in our family $\mathcal{F}$ are defined by uniformly chosen random matrices $\mathbf{F} \in \mathbb{Z}_q^{k \times m}$. We define the auxiliary function family $\mathcal{L}$ by choosing the first $\ell < m$ colums of $\mathbf{L} = [\mathbf{A} \mid \ldots] \in \mathbb{Z}_q^{k \times m}$ uniformly at random, and setting the remaining

columns to $\mathbf{AY}$ where $\mathbf{Y} \in \mathbb{Z}_q^{\ell \times m}$ is a random matrix with discrete gaussian entries of small width $\sigma > \sqrt{n}$. It follows from previous worst-case to average-case reductions [26,23,7], search-to-decision reductions and standard lattice duality results (e.g., see [18],) that the matrix $\mathbf{L} = [\mathbf{A} \mid \mathbf{AY}]$ is pseudorandom. So, $\mathcal{F}$ and $\mathcal{L}$ are computationally indistinguishable. We already know from Lemma 4 that $\mathcal{F}$ is second preimage resistant. In order to conclude that $(\mathcal{F}, \mathcal{L}, \mathcal{X})$ is a lossy function family (and therefore, one-way, see Section 1.3,) it only remains to prove that $\mathcal{L}$ is uninvertible with respect to input distribution $\mathcal{X}$. To this end, express $\mathbf{L}$ as a product $\mathbf{L} = \mathbf{A}[\mathbf{I} \mid \mathbf{Y}]$. Notice that $\mathbf{L}$ is the composition of (linear) functions $[\mathbf{I} \mid \mathbf{Y}]$ and $\mathbf{A}$. The first function $[\mathbf{I} \mid \mathbf{Y}]$ is uninvertible with respect to input distribution $\mathcal{X}$ because its matrix has small entries and necessarily maps small input vectors X to a small set of short output vectors. The composition of an uninvertible function with an arbitrary function remains uninvertible. Therefore also $\mathbf{L} = [\mathbf{A} \mid \mathbf{AY}]$ is uninvertible. The rest of the proof is standard, and follows the general lossy function approach of [25]: the function $\mathbf{F}$ is uninvertible because any efficient inverter for $\mathbf{F}$ would allow to distinguish $\mathbf{F}$ from the uninvertible function $\mathbf{L}$, and contradict the pseudorandomness of $\mathbf{L}$. Since $\mathbf{F}$ is both uninvertible and second-preimage resistant, it is also one-way. (See Section 1.3 and full version [20].) Finally, using the search-to-decision reduction of [18] we conclude that $\mathcal{F}$ is not only one-way, but pseudorandom.

Main result. We begin by defining our uninvertible function family consisting of matrices with small entries.

Definition 1. *For any probability distribution $\mathcal{Y}$ over $\mathbb{Z}^\ell$ and integer $m \geq \ell$, let $\mathcal{I}(m, \ell, \mathcal{Y})$ be the probability distribution over linear functions $[\mathbf{I} \mid \mathbf{Y}] \colon \mathbb{Z}^m \to \mathbb{Z}^\ell$ where $\mathbf{I}$ is the $\ell \times \ell$ identity matrix, and $\mathbf{Y} \in \mathbb{Z}^{\ell \times (m-\ell)}$ is obtained choosing each column of $\mathbf{Y}$ independently at random from $\mathcal{Y}$.*

The next lemma shows that, for appropriate parameter values, this is indeed an uninvertible function family.

Lemma 5. *Let $\mathcal{Y} = D_{\mathbb{Z},\sigma}^\ell$ be the discrete Gaussian distribution with parameter $\sigma > 0$, and let $X \subseteq \{-s, \ldots, s\}^m$. Then $\mathcal{I}(m, \ell, \mathcal{Y})$ is ϵ-uninvertible with respect to $\mathcal{U}(X)$, for $\epsilon = O(\sigma m s / \sqrt{\ell})^\ell / |X| + 2^{-\Omega(m)}$.*

Proof. It is enough to bound the expected size of $f(X)$ when $f \leftarrow \mathcal{I}(m, \ell, \mathcal{Y})$ is chosen at random. Remember that $f = [\mathbf{I} \mid \mathbf{Y}]$ where $\mathbf{Y} \leftarrow D_{\mathbb{Z},\sigma}^{\ell \times (m-\ell)}$. Since the entries of $\mathbf{Y} \in \mathbb{R}^{\ell \times (m-\ell)}$ are independent zero-mean subgaussians with parameter σ, by a standard bound from the theory of random matrices, the largest singular value $s_1(\mathbf{Y}) = \max_{\mathbf{0} \neq \boldsymbol{x} \in \mathbb{R}^m} \|\mathbf{Y}\boldsymbol{x}\| / \|\boldsymbol{x}\|$ of $\mathbf{Y}$ is at most $\sigma \cdot O(\sqrt{\ell} + \sqrt{m - \ell}) = \sigma \cdot O(\sqrt{m})$, except with probability $2^{-\Omega(m)}$. We now bound the ℓ_2 norm of all vectors in the image $f(X)$. Let $\boldsymbol{u} = (\boldsymbol{u}_1, \boldsymbol{u}_2) \in X$, with $\boldsymbol{u}_1 \in \mathbb{Z}^\ell$ and $\boldsymbol{u}_2 \in \mathbb{Z}^{m-\ell}$. Then $\|f(\boldsymbol{u})\| \leq \|\boldsymbol{u}_1 + \mathbf{Y}\boldsymbol{u}_2\| \leq \|\boldsymbol{u}_1\| + \|\mathbf{Y}\boldsymbol{u}_2\| \leq \left(\sqrt{\ell} + s_1(\mathbf{Y})\sqrt{m - \ell}\right) s \leq \left(\sqrt{\ell} + \sigma \cdot O(\sqrt{m})\sqrt{m - \ell}\right) s = O(\sigma m s)$. The number of integer points in the

ℓ-dimensional zero-centered ball of radius $R = O(\sigma m s)$ can be bounded by a simple volume argument, as $|f(X)| \leq (R + \sqrt{\ell}/2)^n V_\ell = O(\sigma m s/\sqrt{\ell})^\ell$, where $V_\ell = \pi^{\ell/2}/(\ell/2)!$ is the volume of the ℓ-dimensional unit ball. Dividing by the size of X and accounting for the rare event that $s_1(\mathbf{Y})$ is not bounded as above, we get that $\mathcal{I}(m, \ell, \mathcal{Y})$ is ϵ-uninvertible for $\epsilon = O(\sigma m s/\sqrt{\ell})^\ell/|X| + 2^{-\Omega(m)}$. □

We can now prove the one-wayness of the SIS function family by defining and analyzing an appropriate lossy function family. The parameters below are set up to expose the connection with LWE, via lattice duality: $\mathrm{SIS}(m, m-n, q)$ corresponds to LWE in n dimensions (given m samples), whose one-wayness we are proving, while $\mathrm{SIS}(\ell = m - n + k, m - n, q)$ corresponds to LWE in $k \leq n$ dimensions, whose pseudorandomness we are assuming.

Theorem 5. *Let q be a modulus and let $\mathcal{X}, \mathcal{Y}$ be two distributions over $\mathbb{Z}^m$ and $\mathbb{Z}^\ell$ respectively, where $\ell = m - n + k$ for some $0 < k \leq n \leq m$, such that*

1. *$\mathcal{I}(m, \ell, \mathcal{Y})$ is uninvertible with respect to input distribution $\mathcal{X}$,*
2. *$\mathrm{SIS}(\ell, m-n, q)$ is pseudorandom with respect to input distribution $\mathcal{Y}$, and*
3. *$\mathrm{SIS}(m, m-n, q)$ is second-preimage resistant with respect to input distribution $\mathcal{X}$.*

Then $\mathcal{F} = \mathrm{SIS}(m, m-n, q)$ is one-way with respect to input distribution $\mathcal{X}$. In particular, if $\mathrm{SIS}(\ell, m-n, q)$ is pseudorandom with respect to the discrete Gaussian distribution $\mathcal{Y} = D^\ell_{\mathbb{Z},\sigma}$, then $\mathrm{SIS}(m, m-n, q)$ is $(2\epsilon + 2^{-\Omega(m)})$-one-way with respect to the uniform input distribution $\mathcal{X} = \mathcal{U}(X)$ over any set $X \subseteq \{-s, \ldots, s\}^m$ satisfying $(C'\sigma m s/\sqrt{\ell})^\ell/\epsilon \leq |X| \leq \epsilon \cdot (q/s')^{m-n}$, where s' is the largest divisor of q that is smaller than or equal to $2s$, and C' is the universal constant hidden by the $O(\cdot)$ notation from Lemma 5.

In order to conclude that the LWE function is pseudorandom (under worst-case lattice assumptions) for uniformly random small errors, we combine Theorem 5 with previous hardness results [26,23,7] and search-to-decision reductions [18,19]. For simplicity, we focus on the important case of a prime modulus q. Nearly identical results for composite moduli (e.g., those divisible by only small primes) are also easily obtained.

Theorem 6. *Let $0 < k \leq n \leq m - \omega(\log k) \leq k^{O(1)}$, $\ell = m - n + k$, $s \geq (Cm)^{\ell/(n-k)}$ for a large enough universal constant C, and q be a prime such that $\max\{3\sqrt{k}, (4s)^{m/(m-n)}\} \leq q \leq k^{O(1)}$. For any set $X \subseteq \{-s, \ldots, s\}^m$ of size $|X| \geq s^m$, the $\mathrm{SIS}(m, m-n, q)$ (equivalently, $\mathrm{LWE}(m, n, q)$) function family is one-way (and pseudorandom) with respect to the uniform input distribution $\mathcal{X} = \mathcal{U}(X)$, under the assumption that SIVP_γ and GapSVP_γ are (quantum) hard to approximate, in the worst case, on k-dimensional lattices to within a factor $\gamma = \tilde{O}(\sqrt{k} \cdot q)$.*

Example parameters. We conclude the section with a few notable instantiations of the last theorem. We remark that the condition $|X| \geq s^m$ in Theorem 6 is

not essential, and the same proof yields similar results also with weaker lower bounds on $|X|$. To obtain pseudorandomness for binary errors, we need $s = 2$ and $X = \{0,1\}^m$. For this value of s, the condition $s \geq (Cm)^{\ell/(n-k)}$ can be equivalently be rewritten as $m \leq (n-k) \cdot (1 + 1/\log_2(Cm))$, which can be satisfied by taking $k = n/(C' \log_2 n)$ and $m = n(1 + 1/(c \log_2 n))$ for any desired $c > 1$ and a sufficiently large constant $C' > 1/(1 - 1/c)$. For these values, the modulus should satisfy $q \geq 8^{m/(m-n)} = 8n^{3c} = k^{O(1)}$, and can be set to any sufficiently large prime $p = k^{O(1)}$.[2]

Notice that for binary errors, both the worst-case lattice dimension k and the number $m - n$ of "extra" LWE samples (i.e., the number of samples beyond the LWE dimension n) are sublinear in the LWE dimension n: we have $k = \Theta(n/\log n)$ and $m - n = O(n/\log n)$. This corresponds to both a stronger worst-case security assumption, and a less useful LWE problem. By using larger errors, say, bounded by $s = n^\epsilon$ for some constant $\epsilon > 0$, it is possible to make both the worst-case lattice dimension k and number of extra samples $m-n$ into (small) linear functions of the LWE dimension n, which may be sufficient for some cryptographic applications of LWE. Specifically, for any constant $\epsilon < 1$, one may set $k = (\epsilon/3)n$ and $m = (1+\epsilon/3)n$, which are easily verified to satisfy all the hypotheses of Theorem 6 when $q = k^{O(1)}$ is sufficiently large. These parameters correspond to $(\epsilon/3)n = \Omega(n)$ extra samples (beyond the LWE dimension n), and to the worst-case hardness of lattice problems in dimension $(\epsilon/3)n = \Omega(n)$. Notice that for $\epsilon < 1/2$, this version of LWE has much smaller errors than allowed by previous LWE hardness proofs, and it would be subject to subexponential-time attacks [2] if the number of samples were not restricted. Our result shows that if the number of samples is limited to $(1 + \epsilon/3)n$, then LWE maintains its provable security properties and conjectured exponential-time hardness in the dimension n.

One last instantiation allows for a linear number of samples $m = c \cdot n$ for any desired constant $c \geq 1$, which is enough for most applications of LWE in lattice cryptography. In this case we can choose (say) $k = n/2$, and it suffices to set the other parameters so that $s \geq (Cm)^{2c-1}$ and $q \geq (4s)^{c/(c-1)} \geq 4^{c/(c-1)} \cdot (Ccn)^{2c+1+1/(c-1)} = k^{O(1)}$. (We can also obtain better lower bounds on s and q by letting k be a smaller constant fraction of n.) This proves the hardness of LWE with uniform noise of polynomial magnitude $s = n^{O(1)}$, and any linear number of samples $m = O(n)$. Note that for $m = cn$, any instantiation of the parameters requires the magnitude s of the errors to be at least n^{c-1}. For $c > 3/2$, this is more noise than is typically used in the standard LWE problem, which allows errors of magnitude as small as $O(\sqrt{n})$, but requires them to be independent and follow a Gaussian-like distribution. The novelty in this last instantiation of Theorem 6 is that it allows for a much wider class of error distributions, including the uniform distribution, and distributions where different components of the error vector are correlated.

[2] Here we have not tried to optimize the value of q, and smaller values of the modulus are certainly possible: a close inspection of the proof of Theorem 6 reveals that for binary errors, the condition $q \geq 8n^{3c}$ can be replaced by $q \geq n^{c'}$ for any constant $c' > c$.

References

1. Ajtai, M.: Generating hard instances of lattice problems. Quaderni di Matematica 13, 1–32 (1996); Preliminary version in STOC 1996
2. Arora, S., Ge, R.: New algorithms for learning in presence of errors. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part I. LNCS, vol. 6755, pp. 403–415. Springer, Heidelberg (2011)
3. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 719–737. Springer, Heidelberg (2012)
4. Bellare, M., Kiltz, E., Peikert, C., Waters, B.: Identity-based (lossy) trapdoor functions and applications. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 228–245. Springer, Heidelberg (2012)
5. Blum, A., Kalai, A., Wasserman, H.: Noise-tolerant learning, the parity problem, and the statistical query model. J. ACM 50(4), 506–519 (2003)
6. Boneh, D., Freeman, D.M.: Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 1–16. Springer, Heidelberg (2011)
7. Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: STOC, pp. 575–584 (2013)
8. Cai, J.-Y., Nerurkar, A.: An improved worst-case to average-case connection for lattice problems. In: FOCS, pp. 468–477 (1997)
9. Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better lattice security estimates. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 1–20. Springer, Heidelberg (2011)
10. Dadush, D., Peikert, C., Vempala, S.: Enumerative lattice algorithms in any norm via M-ellipsoid coverings. In: FOCS, pp. 580–589 (2011)
11. Döttling, N., Müller-Quade, J.: Lossy codes and a new variant of the learning-with-errors problem. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 18–34. Springer, Heidelberg (2013)
12. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008)
13. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC, pp. 197–206 (2008)
14. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261(4), 515–534 (1982)
15. Lindner, R., Peikert, C.: Better key sizes (and attacks) for LWE-based encryption. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 319–339. Springer, Heidelberg (2011)
16. Micciancio, D.: Almost perfect lattices, the covering radius problem, and applications to Ajtai's connection factor. SIAM J. Comput. 34(1), 118–169 (2004)
17. Micciancio, D.: Duality in lattice cryptography. In: PKC (2010) (Invited talk)
18. Micciancio, D., Mol, P.: Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 465–484. Springer, Heidelberg (2011)
19. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012)

20. Micciancio, D., Peikert, C.: Hardness of sis and lwe with small parameters. IACR Cryptology ePrint Archive, 2013, 69 (2013)
21. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on Gaussian measures. SIAM J. Comput. 37(1), 267–302 (2004); Prelim. version in FOCS 2004
22. Micciancio, D., Regev, O.: Lattice-based cryptography. In: Post Quantum Cryptography, pp. 147–191. Springer (February 2009)
23. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem. In: STOC, pp. 333–342 (2009)
24. Peikert, C.: An efficient and parallel Gaussian sampler for lattices. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 80–97. Springer, Heidelberg (2010)
25. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: STOC, pp. 187–196 (2008)
26. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM 56(6), 1–40 (2005); Preliminary version in STOC 2005
27. Wagner, D.: A generalized birthday problem. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 288–303. Springer, Heidelberg (2002)

Lattice Signatures and Bimodal Gaussians

Léo Ducas[1], Alain Durmus[2,*], Tancrède Lepoint[3], and Vadim Lyubashevsky[4]

[1] ENS Paris, France
[2] ENPC and ENS Cachan, France
[3] CryptoExperts and ENS Paris, France
[4] INRIA and ENS Paris, France
{Leo.Ducas,Alain.Durmus,Tancrede.Lepoint,Vadim.Lyubashevsky}@ens.fr

Abstract. Our main result is a construction of a lattice-based digital signature scheme that represents an improvement, both in theory and in practice, over today's most efficient lattice schemes. The novel scheme is obtained as a result of a modification of the rejection sampling algorithm that is at the heart of Lyubashevsky's signature scheme (Eurocrypt, 2012) and several other lattice primitives. Our new rejection sampling algorithm which samples from a *bimodal* Gaussian distribution, combined with a modified scheme instantiation, ends up reducing the standard deviation of the resulting signatures by a factor that is asymptotically square root in the security parameter. The implementations of our signature scheme for security levels of 128, 160, and 192 bits compare very favorably to existing schemes such as RSA and ECDSA in terms of efficiency. In addition, the new scheme has shorter signature and public key sizes than all previously proposed lattice signature schemes.

As part of our implementation, we also designed several novel algorithms which could be of independent interest. Of particular note, is a new algorithm for efficiently generating discrete Gaussian samples over $\mathbb{Z}^n$. Current algorithms either require many high-precision floating point exponentiations or the storage of very large pre-computed tables, which makes them completely inappropriate for usage in constrained devices. Our sampling algorithm reduces the hard-coded table sizes from linear to logarithmic as compared to the time-optimal implementations, at the cost of being only a small factor slower.

1 Introduction

Lattice cryptography is arguably the most promising replacement for standard cryptography after the eventual coming of quantum computers. The most ubiquitous public-key cryptographic primitives, encryption schemes [18,26] and digital signatures [24,15], already have somewhat practical lattice-based instantiations. In addition, researchers are rapidly discovering new lattice-based primitives, such as fully-homomorphic encryption [10], multi-linear maps [9], and attribute-based encryption [14], that had no previous constructions based on classical number-theoretic techniques. Even though the above primitives are quite varied in their

* This work was done while the author was at ENS Paris, France.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 40–56, 2013.

Table 1. Benchmarking on a desktop computer (Intel Core i7 at 3.4Ghz, 32GB RAM) with `openssl` 1.0.1c

	Security	Signature size	Sign (ms)	Sign/s	Verify (ms)	Verify/s
BLISS-0	$\leqslant$ 60 bits	3.3 kilobits	0.241	4k	0.017	59k
BLISS-I	128 bits	5.6 kb	0.124	8k	0.030	33k
BLISS-II	128 bits	5 kb	0.480	2k	0.030	33k
BLISS-III	160 bits	6 kb	0.203	5k	0.031	32k
BLISS-IV	192 bits	7 kb	0.375	2.5k	0.032	31k
RSA 1024	72-80 bits	1 kb	0.167	6k	0.004	91k
RSA 2048	103-112 bits	2 kb	1.180	0.8k	0.038	27k
RSA 4096	$\geqslant$ 128 bits	4 kb	8.660	0.1k	0.138	7.5k
ECDSA[1] **160**	80 bits	0.32 kb	0.058	17k	0.205	5k
ECDSA 256	128 bits	0.5 kb	0.106	9.5k	0.384	2.5k
ECDSA 384	192 bits	0.75 kb	0.195	5k	0.853	1k

functionalities, many of them share the same basic building blocks. Thus an improvement in one of these fundamental building blocks, usually results in the simultaneous improvement throughout lattice cryptography. For example, the recent work on the lattice trapdoor generation algorithm [27] resulted in immediate efficiency improvements in lattice-based hash-and-sign signatures, identity-based encryption schemes, group signatures, and functional encryption schemes.

In this work, we propose an improvement of another such building block – the *rejection sampling* procedure that is present in the most efficient constructions of lattice-based digital signatures [24,15], authentication schemes [23], blind signatures [31], and zero-knowledge proofs used in multi-party computation [4]. As a concrete application, we show that with our new algorithm, lattice-based digital signatures become completely practical. We construct and implement a family of digital signature schemes, named BLISS (Bimodal Lattice Signature Scheme) for security levels of 128, 160, and 192 bits. On standard 64-bit processors, our proof-of-concept implementations constitute significant improvements over previous lattice-based signatures and compare very favorably to the `openssl` implementations of RSA and ECDSA signatures schemes (see Table 1).

As part of our implementation, we also designed several novel algorithms that could be of independent interest. Chiefly among them is a new procedure that very efficiently samples from the Gaussian distribution over $\mathbb{Z}^m$ without requiring a very large look-up table. The absence of such an algorithm made researchers avoid using the Gaussian distribution when implementing lattice-based schemes on constrained devices, which resulted in these schemes being less compact than they could have been [15].

1.1 Related Work

Rejection Sampling. Rejection sampling in lattice constructions was first used by Lyubashevsky [22] to construct a three-round identification scheme. A

[1] ECDSA on a prime field $\mathbb{F}_p$: `ecdsap160`, `ecdsap256` and `ecdsap384` in `openssl`.

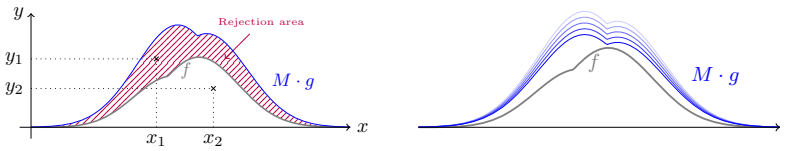
standard identification scheme is a three round sigma protocol that consists of a commit, challenge, and response stages. The main idea underlying their constructions and security proofs from number theoretic assumptions (e.g. Schnorr and GQ schemes [2]) is that the value y committed to in the first stage is used to information-theoretically hide the secret key s in the third stage. This is relatively straight-forward to do in number-theoretic schemes because one can just commit to a random y and then add it to (or multiply it by) some challenge-dependent function of s. Since all operations are performed in a finite ring, y being uniformly random hides s. In lattice constructions, however, we need to hide the secret key with a *small* y. The solution is thus to choose y from a narrow distribution and then perform rejection sampling so that s is not leaked when we add y to it (we describe this idea in much greater detail in Section 1.2). The improvements in lattice-based identification schemes (and therefore signature schemes via the Fiat-Shamir transformation) partly came via picking distributions that were more amenable to rejection sampling.

Lattice Signatures. Early lattice-based signature proposals did not have security reductions [13,19,17], and they were all subsequently broken because it turned out that every signature leaked a part of the secret key [12,29,6]. Among known provably-secure signature schemes, [11,23], [24,27], the most efficient seems to be that of [24] whose most efficient instantiation has both signature and key size of the order of 9kb [15] for approximately 80 bits of security.[2]

1.2 Our Results and Techniques

Rejection Sampling and Signature Construction. To understand our improvement of the rejection sampling procedure, we believe that it is useful to first give an overview of rejection sampling and the most efficient way in which it is currently used in constructing lattice-based signatures [24]. Rejection sampling is a well-known method introduced by von Neumann [33] to sample from an arbitrary target probability distribution f, given a source bound to a different probability distribution g. Conceptually, the method works as follows. A sample x is drawn from g and is accepted with probability $f(x)/(M \cdot g(x))$, where M is some positive real. If it is not accepted, then the process is restarted. It is not hard to prove that if $f(x) \leqslant M \cdot g(x)$ for all x, then the rejection sampling procedure produces exactly the distribution of f. Furthermore, because the expected number of times the procedure will need to be restarted is M, it is crucial to keep M as small as possible, possibly by tailoring the function g so that it resembles the target function f as much as possible. In particular, since rejection sampling can be interpreted as sampling a random point (x_i, y_i) in the area under the distribution $M \cdot g$ (see Figure 1) and accepting if and only if $y_i \leqslant f(x_i)$, reducing the area between the two curves will reduce M.

[2] In [15], a 100-bit security level was claimed, but the cryptanalysis we use in the full version of this paper [5], which combines lattice-reduction attacks with combinatorial meet-in-the-middle techniques [20], estimates the actual security to be around 75-80 bits.



(a) (x_i, y_i) is sampled uniformly in the area under $M \cdot g$, and accepted when $y_i \leqslant f(x_i)$

(b) M can be reduced when g is better adapted to f

Fig. 1. Rejection sampling from the distribution of g to get the distribution of f

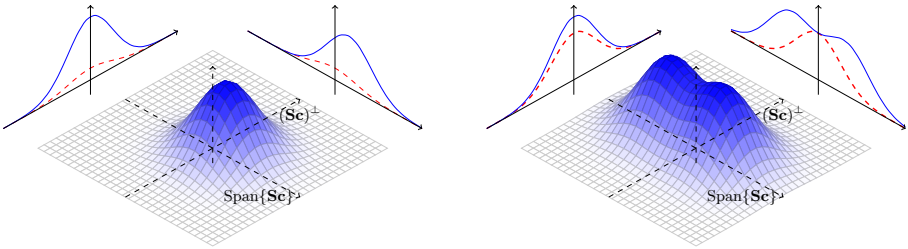
The digital signature from [24] works as follows (for the sake of this discussion, we will present the simplest version based on SIS): the secret key is an $m \times n$ matrix $\mathbf{S}$ with small coefficients, and the public key consists of a random $n \times m$ matrix $\mathbf{A}$ whose entries are uniform in $\mathbb{Z}_q$ and $\mathbf{T} = \mathbf{AS} \bmod q$. There is also a cryptographic hash function H, modeled as a random oracle, which outputs elements in $\mathbb{Z}^n$ with small norms. To sign a message digest μ, the signing algorithm first picks a vector $\mathbf{y}$ according to the distribution D_σ^m, where D_σ^m is the discrete Gaussian distribution over $\mathbb{Z}^m$ with standard deviation σ. The signer then computes $\mathbf{c} = H(\mathbf{Ay} \bmod q, \mu)$ and produces a potential signature $(\mathbf{z}, \mathbf{c})$ where $\mathbf{z} = \mathbf{Sc} + \mathbf{y}$. Notice that the distribution of $\mathbf{z}$ depends on the distribution of $\mathbf{Sc}$, and thus on the distribution of $\mathbf{S}$ – in fact, the distribution of $\mathbf{z}$ is exactly D_σ^m *shifted* by the vector $\mathbf{Sc}$.

To remove the dependence of the signature on $\mathbf{S}$, rejection sampling is used. The target distribution that we want for signatures is D_σ^m, whereas we obtain samples from the distribution D_σ^m shifted by $\mathbf{Sc}$ (call this distribution $D_{\mathbf{Sc},\sigma}^m$). To use rejection sampling, we need to find a positive real M such that for all (or all but a negligible fraction) $\mathbf{x}$ distributed according to D_σ^m we have $D_\sigma^m(\mathbf{x}) \leqslant M \cdot D_{\mathbf{Sc},\sigma}^m(\mathbf{x})$. A simple calculation (see [24, Lemma 4.5]) shows that

$$D_\sigma^m(\mathbf{x}) / D_{\mathbf{Sc},\sigma}^m(\mathbf{x}) = \exp\left(\frac{-2\langle \mathbf{x}, \mathbf{Sc}\rangle + \|\mathbf{Sc}\|^2}{2\sigma^2}\right). \tag{1}$$

The value of $\langle \mathbf{x}, \mathbf{Sc}\rangle$ behaves in many ways as a one-dimensional discrete Gaussian, and it can be thus shown that $|\langle \mathbf{x}, \mathbf{Sc}\rangle| < \tau\sigma\|\mathbf{Sc}\|$ with probability $1 - \exp(-\Omega(\tau^2))$. Asymptotically, the value of τ is proportional to the square root of the security parameter. Concretely, if we would like to have, for example, $1-2^{-100}$ certainty that $|\langle \mathbf{x}, \mathbf{Sc}\rangle| < \tau\sigma\|\mathbf{Sc}\|$, we would set $\tau = 12$. Thus with probability $1 - \exp(-\Omega(\tau^2))$, we have $\exp\left(\frac{-2\langle \mathbf{x},\mathbf{Sc}\rangle + \|\mathbf{Sc}\|^2}{2\sigma^2}\right) \leqslant \exp\left(\frac{2\tau\sigma\|\mathbf{Sc}\| + \|\mathbf{Sc}\|^2}{2\sigma^2}\right)$. So if $\sigma = \tau\|\mathbf{Sc}\|$, we will have $D_\sigma^m(\mathbf{x}) / D_{\mathbf{Sc},\sigma}^m(\mathbf{x}) \leqslant \exp\left(1 + \frac{1}{2\tau^2}\right)$. Therefore if we set $M = \exp\left(1 + \frac{1}{2\tau^2}\right)$, rejection sampling outputs signatures that are distributed according to D_σ^m where $\sigma = \tau\|\mathbf{Sc}\|$ and the expected number of repetitions is $M \approx \exp(1)$.[3]

[3] More precisely $\sigma = \tau \max_{\mathbf{S},\mathbf{c}} \|\mathbf{Sc}\|$, since $\mathbf{Sc}$ is not known in advance.



(a) In the original scheme of [24] (b) In our scheme

Fig. 2. Improvement of Rejection Sampling with Bimodal Gaussian Distributions. In blue is the distribution of $\mathbf{z}$, for fixed $\mathbf{Sc}$ and over the space of all $\mathbf{y}$ in Figure (a) and all $(b, \mathbf{y})$ in Figure (b), before the rejection step and its decomposition as a Cartesian product over $\mathrm{Span}\{\mathbf{Sc}\}$ and $(\mathbf{Sc})^{\perp}$. In dashed red is the target distribution scaled by $1/M$.

Prior to explaining our technique to improve the scheme, we need to state how the verification algorithm in [24] works. Upon receiving the signature $(\mathbf{z}, \mathbf{c})$ of μ, the verifier checks that $\|\mathbf{z}\|$ is "small" (roughly $\sigma\sqrt{m}$) and also that $\mathbf{c} = H(\mathbf{Az} - \mathbf{Tc} \bmod q, \mu)$. It is easy to check that the outputs of the signing procedure satisfy the two requirements. In this work, we show how to remove the factor τ (in fact even more) from the required standard deviation. Above, we described how to perform rejection sampling when we were sampling potential signatures as $\mathbf{z} = \mathbf{Sc} + \mathbf{y}$. Consider now, an alternative procedure, where we first uniformly sample a bit $b \in \{-1, 1\}$ and then choose the potential signature to be $\mathbf{z} = b\mathbf{Sc} + \mathbf{y}$. In particular $\mathbf{z}$ is now sampled from the distribution $\frac{1}{2}D^m_{\mathbf{Sc},\sigma} + \frac{1}{2}D^m_{-\mathbf{Sc},\sigma}$. If our target distribution is still D^m_σ, then, as above, we need to have $D^m_\sigma(\mathbf{x}) / \left(\frac{1}{2}D^m_{\mathbf{Sc},\sigma}(\mathbf{x}) + \frac{1}{2}D^m_{-\mathbf{Sc},\sigma}(\mathbf{x})\right) \leqslant M$. By using Equation (1) and some algebraic manipulations, we obtain that

$$D^m_\sigma(\mathbf{x}) / \left(\frac{1}{2}D^m_{\mathbf{Sc},\sigma}(\mathbf{x}) + \frac{1}{2}D^m_{-\mathbf{Sc},\sigma}(\mathbf{x})\right) = \exp\left(\frac{\|\mathbf{Sc}\|^2}{2\sigma^2}\right) / \cosh\left(\frac{\langle \mathbf{x}, \mathbf{Sc}\rangle}{\sigma^2}\right) \leqslant \exp\left(\frac{\|\mathbf{Sc}\|^2}{2\sigma^2}\right),$$

where the last inequality follows from the fact that $\cosh(y) \geqslant 1$ for all y. Thus for rejection sampling to work with $M = \exp(1)$, as in the previous example, we only require that $\sigma = \|\mathbf{Sc}\|/\sqrt{2}$ rather than $\tau\|\mathbf{Sc}\|$.

Our improvement is depicted on Figure 2. Part 2(a) shows the rejection sampling as done in [24]. There, the distribution D^m_σ (the dashed red line) must be scaled by a somewhat large factor so that all but a negligible fraction of it fits under $D^m_{\mathbf{Sc},\sigma}$. In 2(b), which represents our improved sampling algorithm, the distribution from which we are sampling is bimodal having its two centers at $\mathbf{Sc}$ and $-\mathbf{Sc}$. As can be seen from the figure, the distribution D^m_σ fits much "better" (*i.e.* needs

to be scaled by a much smaller factor) underneath the bimodal distribution and therefore there is a much smaller rejection area between the two curves. As a side note, whereas in (a), a negligible fraction of the scaled D_σ^m is still above $D_{\mathbf{Sc},\sigma}^m$, in (b), all of D_σ^m is underneath the bimodal distribution $\frac{1}{2}D_{\mathbf{Sc},\sigma}^m + \frac{1}{2}D_{-\mathbf{Sc},\sigma}^m$.

While the above sampling procedure potentially produces much shorter signatures since the Gaussian "tail-cut" factor τ is never used, it does not give an improved signature scheme by itself because the verification procedure is no longer guaranteed to work. The verification checks that $\mathbf{c} = H(\mathbf{Az}-\mathbf{Tc} \bmod q, \mu)$ and so will verify correctly if and only if $\mathbf{Ay} = \mathbf{Az} - \mathbf{Tc} = \mathbf{A}(b\mathbf{Sc} + \mathbf{y}) - \mathbf{Tc} = \mathbf{Ay} + b\mathbf{Tc} - \mathbf{Tc}$, which will only happen if $b\mathbf{Tc} = \mathbf{Tc} \bmod q$ for $b \in \{-1, 1\}$. In other words, we will need $\mathbf{Tc} = -\mathbf{Tc} \bmod q$, which will never happen if q is prime unless $\mathbf{T} = \mathbf{0}$. [4] Our solution, therefore, is to work modulo $2q$ and to set $\mathbf{T} = q\mathbf{I}$ where $\mathbf{I}$ is the $n \times n$ identity matrix. In this case $\mathbf{Tc} = -\mathbf{Tc} \bmod 2q$, and so the verification procedure will always work.

Changing the modulus from q to $2q$ and forcing the matrix $\mathbf{T}$ to always be $q\mathbf{I}$ creates several potential problems. In particular, it is no longer clear how to perform key generation, and also the outline for the security proof from [24] no longer holds. But we show that these problems can be overcome. We will now sketch the key generation and the security proof based on the hardness of the SIS problem in which one is given a uniformly random matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$, and is asked to find a short vector $\mathbf{w}$ such that $\mathbf{Bw} = \mathbf{0} \pmod q$. To generate the public and secret keys, we first pick a uniformly random matrix $\mathbf{A}' \in \mathbb{Z}_q^{n \times (m-n)}$ and a random $(m-n) \times n$ matrix $\mathbf{S}'$ consisting of short coefficients. We then compute $\mathbf{A}'' = \mathbf{A}'\mathbf{S}' \bmod q$ and output $\mathbf{A} = [2\mathbf{A}'|2\mathbf{A}'' + q\mathbf{I}]$ as the public key. The secret key is $\mathbf{S} = [\mathbf{S}'| - \mathbf{I}]^T$. Notice that by construction we have $\mathbf{AS} = q\mathbf{I} \pmod{2q}$ and $\mathbf{S}$ consists of small entries. The dimensions m and n are picked so that the distribution of $[\mathbf{A}'|\mathbf{A}'\mathbf{S}' \bmod q]$ can be shown to be uniformly random in $\mathbb{Z}_q^{n \times m}$ by the leftover hash lemma.

In the security proof, we are given a random matrix $\mathbf{B} = [\mathbf{A}'|\mathbf{A}''] \in \mathbb{Z}_q^{n \times m}$ by the challenger and use the adversary that forges a signature to find a short vector $\mathbf{w}$ such that $\mathbf{Bw} = \mathbf{0} \pmod q$. We create the public key $\mathbf{A} = [2\mathbf{A}'|2\mathbf{A}'' + q\mathbf{I}]$ and give it to the adversary. Even though we do not know a secret key $\mathbf{S}$ such that $\mathbf{AS} = q\mathbf{I} \pmod{2q}$, we can still create valid signatures for any messages of the adversary's choosing by picking the $(\mathbf{z}, \mathbf{c})$ according to the correct distributions and then programming the random oracle as is done in [24]. When the adversary forges, we use the forking lemma to create two equations $\mathbf{Az} = q\mathbf{c} \pmod{2q}$ and $\mathbf{Az}' = q\mathbf{c}' \pmod{2q}$. Combining them together, we obtain $\mathbf{A}(\mathbf{z} - \mathbf{z}') = q(\mathbf{c} - \mathbf{c}') \pmod{2q}$. Under some very simple requirements for $\mathbf{z}, \mathbf{z}', \mathbf{c}$, and $\mathbf{c}'$, the previous equation implies that $\mathbf{A}(\mathbf{z} - \mathbf{z}') = \mathbf{0} \pmod q$ and $\mathbf{z} \neq \mathbf{z}'$. This then implies that $2\mathbf{B}(\mathbf{z} - \mathbf{z}') = \mathbf{0} \pmod q$ and since 2 is invertible modulo q, we have found a $\mathbf{w} = (\mathbf{z} - \mathbf{z}')$ such that $\mathbf{Bw} = \mathbf{0} \pmod q$.

[4] One may think that a possible solution could be to output the bit b as part of the signature, but this is not secure. Depending on the sign of $\langle \mathbf{z}, \mathbf{Sc} \rangle$, one of the two values of b is more likely to be output than the other. Therefore outputting the bit b leaks information about $\mathbf{S}$.

The above scheme construction and proof work for SIS and equally well for Ring-SIS, when instantiated with polynomials. As in [24], we can also construct much more efficient schemes based on LWE and Ring-LWE by creating the matrix $\mathbf{A}'' = \mathbf{A}'\mathbf{S}'$ such that $(\mathbf{A}', \mathbf{A}'')$ is not uniformly random, but only computationally. For optimal efficiency, though, we can create the key in yet a different manner related to the way NTRU keys are generated. The formal construction is described in the full version, and we just give the intuition here. We could create two small polynomials $\mathbf{s}_1, \mathbf{s}_2 \in \mathbb{Z}[x]/(x^n + 1)$ and output the public key as $\mathbf{a} = \frac{\mathbf{q}-\mathbf{s}_2}{\mathbf{s}_1} \pmod{2q}$. Notice that this implies that $\mathbf{a}\mathbf{s}_1 + \mathbf{s}_2 = \mathbf{q} \pmod{2q}$, and so we can think of the public key as $\mathbf{A} = [\mathbf{a}, \mathbf{1}]$ and the secret key as $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2]^T$. Assuming that it is a hard problem to find small vectors $\mathbf{w}$ such that $\mathbf{A}\mathbf{w} = \mathbf{0} \pmod{2q}$, the signature scheme instantiated in the above manner will be secure. To those readers familiar with the key generation in the NTRU encryption scheme, the above key generation should look very familiar, except that the modulus is $2q$ rather than q. Since we are not sure what happens when the modulus is $2q$, we show in the full version of this paper [5] how to instantiate our scheme so that it is based on NTRU over modulus q. We then explain how for certain instantiations, this is as hard a problem as Ring-SIS (using the results of Stehlé, Steinfeld [32]) and how for more efficient instantiations, it is a *weaker* assumption than the ones underlying the classic NTRU encryption scheme and the recent construction of fully-homomorphic encryption [21].

Gaussian Sampling. There are two generic methods for sampling according to a discrete Gaussian distribution. The first one uses basic rejection sampling as follows: choose a uniform integer $x \in \{-\tau\sigma, \ldots, \tau\sigma\}$ (where $\tau \approx 12$, as in the preceding discussion) and accept it with probability proportional to $\exp(-x^2/2\sigma^2)$ (and restart otherwise). This involves the computation of the exp function to high precision and requires an average of $2\tau/\sqrt{2\pi} \approx 10$ trials, thus wasting a lot of random bits. The second one involves storing large pre-computed data, namely the cumulative distribution table of the discrete Gaussian from $-\tau\sigma$ to $\tau\sigma$. While the second method is very efficient when given enough memory, neither of the two approaches is appropriate for use in constrained devices.

We solve this issue by modifying the first approach to exploit the properties of discrete Gaussians. We recall that a Bernoulli distribution $\mathcal{B}_c$ assigns 1 (True) with probability $c \in [0, 1]$ and 0 (False) with probability $1 - c$. Overloading the notation for the sake of clarity, we will denote by $\mathcal{B}_c$ both the distribution and a generic random variable that follows that distribution independently of all others (thus we may write, for example, $\mathcal{B}_a \oplus \mathcal{B}_b = \mathcal{B}_{a+b-2ab}$). As a first step, to avoid explicit computation of exp, we use the simple fact that for an integer x in binary form $x = x_1 \cdots x_n$ we have $\mathcal{B}_{\exp(-x/f)} = \bigwedge_{i \text{ s.t. } x_i=1} \mathcal{B}_{\exp(-2^i/f)}$. This allows us to sample according to $\mathcal{B}_{\exp(-x/f)}$ using only logarithmically many precomputed values $\exp(-2^i/f)$. Similarly, we also design another algorithm to sample according to $\mathcal{B}_{1/\cosh(x/f)}$, using a Markov chain that makes less than two calls to $\mathcal{B}_{\exp(-x/f)}$ on average.

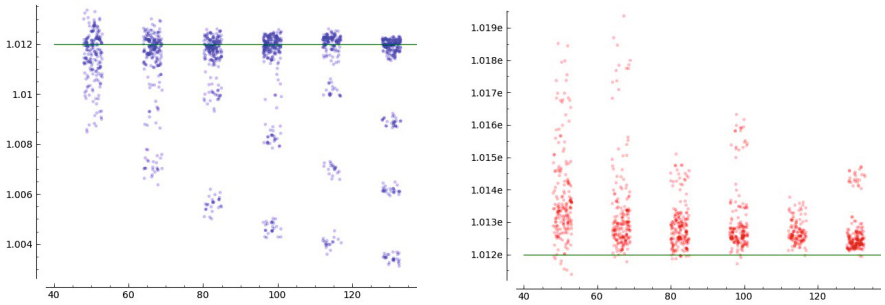
The second step is to replace the uniform distribution from which one chooses an integer by a more adapted one to decrease the rejection rate. It is essential, though, that the rejection rate retains an easily samplable form. To do this, we build on a specific discrete Gaussian of variance $\sigma_2^2 = 1/(2\ln 2)$ for which the distribution $D_{\sigma_2}(x)$ is proportional to 2^{-x^2}. This makes it very easily samplable, and the rejection rate still has the required form $\exp(\cdot/f)$. The final algorithm has bounded repetition rate of 1.5 rather than $2\tau/\sqrt{2\pi} \approx 10$. All the operations are very simple, requiring only small integer arithmetic, and are therefore well-suited for constrained devices.

Cryptanalysis and Experiments on NTRU Lattices. Previous cryptanalytic efforts against schemes based on SIS and LWE mostly involved computing the Hermite factor of the underlying average-case instance, as in the work of Gama and Nguyen [8], and making sure that its value is below the level required for the desired security guarantees. In this work (described in detail in the full version of this paper [5]) we undertake a more careful cryptanalysis by using the results on BKZ 2.0 of Chen and Nguyen [3] in combination with other techniques – namely dual lattice reduction and the combinatorial meet-in-the-middle attack of Howgrave-Graham [20].

For optimal efficiency, the security of our scheme relies on the hardness of a type of NTRU problem that has recently (re-)appeared in the literature [21] and which, we believe, could play a major role in the future of lattice-based cryptography (see Section 2 for the precise definition of the problem). The only cryptanalysis of which we are aware of that studies NTRU lattices deals with instances where the modulus is very close in size to the dimension of the lattice [8,16]. It is thus unclear as to what roles each of the variables plays when looked at independently.

In our work, and also in the previously-mentioned work of [21], the modulus is required to be substantially larger than the dimension. As far as we are aware, no previous cryptanalysis was done for these types of instances. The most complete study of the behavior of BKZ in the presence of unusually short vector(s) is due to Gama and Nguyen [8] who thoroughly analyzed the algorithm's running time in the presence of *one* such vector. Their experiments show that the hardness of finding this vector depends on the ratio λ_2/λ_1, that is, the gap between the second-shortest and the shortest vectors in the m-dimensional lattice. In practice, for BKZ-20, the shortest vector was found when $\lambda_2/\lambda_1 > .48 \cdot 1.01^m$.

We ran similar experiment of BKZ-20 in the case of $2n$-dimensional NTRU lattices where $\lambda_1 = \ldots = \lambda_n$. In NTRU lattices, the gap normally occurs between the n-th and the $n+1$-st successive minima, and one might think that the ratio between these two quantities would somehow determine the hardness of the instance. Our experiments showed that this is not the case, and the shortest vector was found when $\sqrt{qm/2\pi e}/\lambda_1$ was greater than $.40 \cdot 1.012^m$ (see Figure 3). Despite the fact that there is no vector in the lattice having length $\sqrt{qm/2\pi e}$ this is actually consistent with the results of [8]! The reason is that $\sqrt{qm/2\pi e}$ is



(a) Shortest vector not found (b) Shortest vector found

Fig. 3. Results BKZ-20 for $n \in [48, 150]$, $q \in [6000, 25000]$ and binary search on the λ_1-threshold. On horizontal axis is the value of $n + \texttt{random(0,5)}$ and on vertical axis is $\left(\frac{1}{.40}\sqrt{\frac{qm}{2\pi e}}/\lambda_1\right)^{1/2n}$.

the *expected* length of the shortest vector according to the Gaussian heuristic,[5] and we would also expect $\lambda_2 \approx \sqrt{qm/2\pi e}$ in a random q-ary lattice analyzed in [8]. Thus one could say that the hardness of finding a short vector in q-ary lattices depends not on the gap, but rather on the ratio between the Gaussian heuristic and the actual length of the shortest vector.

Similar to the results in [8], when the ratio was smaller than $.40 \cdot 1.012^m$, the resulting shortest vector had length about $\sqrt{q} \cdot 1.012^m$. In other words, BKZ-20 behaved as if the lattice were truly random. Because of our experiments with BKZ-20, it seems reasonable to assume that BKZ behaves analogously for larger block sizes. Thus we can measure its efficacy according to the BKZ 2.0 methodology in [3]. We would like to stress that we have no explanation for the reason why the ratio between the Gaussian heuristic and the actual length of the vector seems to dictate the hardness of finding short vectors in NTRU lattices. We are equally unsure whether this phenomenon implies that these lattices are indeed as hard as the random lattices that have been more exhaustively studied [8,3].

The general dearth of lattice cryptanalysis papers stands in contrast to the vast number of articles proposing theoretical lattice-based constructions. Our belief is that this lack of cryptanalytic effort is in part due to the fact that most of the papers with scheme proposals give no concrete targets to attack. One of the proposed instantiations in the present work is a "toy example" that we estimate has approximately 60 bits of security. Thus if it turns out that NTRU lattices are weaker than believed, it is wholly possible that this example could be broken on a personal computer, and we think this would be of great interest to the practical community. In addition, it could be argued that we do not yet know enough about lattice reduction to be able to propose such "fine-grained" security estimates like 160-bit or 192-bit. But one of the main reasons

[5] The Gaussian heuristic says that for certain types of random lattices $\mathcal{L}$, we will have $\lambda_1(\mathcal{L}) \approx det(\mathcal{L})^{1/m} \cdot \sqrt{\frac{m}{2\pi e}}$ [8].

that we make these proposals is to make it "worthwhile" for cryptanalysts to work on these problems. In short, one of our hopes is that this work spurs on the cryptanalysis that is currently much needed in the field.

Acknowledgments. We thank the CRYPTO 2013 reviewers for their careful reading of the paper and their diligent comments. We also thank Steven Galbraith and Pascal Paillier for useful comments on previous versions of this work.

2 Preliminaries

2.1 Notation

For any integer q, we identify the ring $\mathbb{Z}_q$ with the interval $[-q/2, q/2) \cap \mathbb{Z}$, and in general for a ring $\mathcal{R}$, we define $\mathcal{R}_q$ to be the quotient ring $\mathcal{R}/(q\mathcal{R})$. Whenever working in the quotient ring $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$, we will assume that n is a power of 2 and q is a prime number such that $q = 1 \pmod{2n}$. Vectors, considered as column vectors, will be written in bold lower case letters. Matrices will be written in bold upper case letters. For a positive integer n, we write $\mathbf{I}_n$ to be the identity matrix of dimension n.

We recall that the ℓ_p-norm of a vector $\mathbf{v}$ is defined as $\|\mathbf{v}\|_p = (\sum_i |v_i|^p)^{1/p}$ for $p > 0$, and its ℓ_∞-norm as $\|\mathbf{v}\|_\infty = \max_i |v_i|$. By default, we use $\|\cdot\|$ for the ℓ_2-norm.

We now state a general rejection sampling lemma. The proof of this lemma is quite standard (cf. [24]).

Lemma 2.1 (Rejection Sampling). *Let V be an arbitrary set, and $h\colon V \to \mathbb{R}$ and $f\colon \mathbb{Z}^m \to R$ be probability distributions. If $g_v\colon \mathbb{Z}^m \to R$ is a family of probability distributions indexed by $v \in V$ with the property that there exists a $M \in \mathbb{R}$ such that*

$$\forall v \in V, \forall \mathbf{z} \in \mathbb{Z}^m, M \cdot g_v(\mathbf{z}) \geqslant f(\mathbf{z}) \ ,$$

then, the output distributions of the following two algorithms are identical:

1. *$v \leftarrow h$, $z \leftarrow g_v$, output $(\mathbf{z}, v)$ with probability $f(\mathbf{z})/\big(M \cdot g_v(\mathbf{z})\big)$.*
2. *$v \leftarrow h$, $z \leftarrow f$, output $(\mathbf{z}, v)$ with probability $1/M$.*

2.2 Discrete Gaussian Distribution

Gaussian Distribution. The (un-normalized) Gaussian distribution with standard deviation $\sigma \in \mathbb{R}$ and center $c \in \mathbb{R}$ evaluated at $x \in \mathbb{R}$ is defined by $\rho_{c,\sigma}(x) = \exp\big(\frac{-(x-c)^2}{2\sigma^2}\big)$, and more generally by $\rho_{\mathbf{c},\sigma}(\mathbf{x}) = \exp\big(\frac{-\|\mathbf{x}-\mathbf{c}\|^2}{2\sigma^2}\big)$ for $\mathbf{x}, \mathbf{c} \in \mathbb{R}^n$. When the center $\mathbf{c}$ is $\mathbf{0}$, we generally omit it from the notation and simply write $\rho_\sigma(\mathbf{x})$. The discrete Gaussian distribution over $\mathbb{Z}$ centered at 0 is defined by $D_\sigma(x) = \rho_\sigma(x)/\rho_\sigma(\mathbb{Z})$, and more generally, over $\mathbb{Z}^m$ by $D^m_\sigma(\mathbf{x}) = \rho_\sigma(\mathbf{x})/\rho_\sigma(\mathbb{Z})^m$.

Tailcutting. It is generally useful to ignore large values which are unlikely to appear when drawing according to a Gaussian distribution.

Lemma 2.2 ([28]). *For any dimension m, $\sigma > 0$ and $\tau > 1$, $\rho_\sigma(\mathbb{Z}^m \setminus \tau\sigma\sqrt{m}\mathfrak{B}) < 2C(\tau)^m \cdot \rho_\sigma(\mathbb{Z})^m$, where $C(\tau) = \tau \exp\left(\frac{1-\tau^2}{2}\right) < 1$, and $\mathfrak{B}$ is the centered ℓ_2 unit ball.*

Therefore, to tailcut less than $2^{-\lambda}$ of a 1-dimensional Gaussian, one should choose $\tau \approx \sqrt{\lambda \cdot 2 \ln 2}$, the typical value being $\tau = 12$ for $\lambda = 100$.

2.3 Hardness Assumptions

All the constructions in this paper are based on the hardness of the *generalized SIS* (Short Integer Solution) problem, which we define below.

Definition 2.3 ($\mathcal{R}$-SIS$^{\mathcal{K}}_{q,n,m,\beta}$ problem). *Let $\mathcal{R}$ be some ring and $\mathcal{K}$ be some distribution over $\mathcal{R}_q^{n\times m}$, where $\mathcal{R}_q$ is the quotient ring $\mathcal{R}/(q\mathcal{R})$. Given a random $\mathbf{A} \in \mathcal{R}_q^{n\times m}$ drawn according to the distribution $\mathcal{K}$, find a non-zero $\mathbf{v} \in \mathcal{R}_q^m$ such that $\mathbf{A}\mathbf{v} = \mathbf{0}$ and $\|\mathbf{v}\|_2 \leqslant \beta$.*

If we let $\mathcal{R} = \mathbb{Z}$ and $\mathcal{K}$ be the uniform distribution, then the resulting problem is the classical SIS problem first defined by Ajtai [1] in his seminal paper showing connections between worst-case lattice problems and the average-case SIS problem. By the pigeonhole principle, if $\beta \geqslant \sqrt{m}q^{n/m}$ then the SIS instances are guaranteed to have a solution. Using Gaussian techniques, Micciancio and Regev [28] improved Ajtai's result to show that, for a large enough q as a function of n and β, the SIS$_{q,n,m,\beta}$ problem is as hard (on the average) as the $\tilde{\mathcal{O}}(\sqrt{n}\beta)$-SIVP problem for *all* lattices of dimension n.

In 2006, a ring variant of SIS was introduced independently by Peikert and Rosen [30] and Lyubashevsky and Micciancio [25]. In [25] it was shown that if $\mathcal{R} = \mathbb{Z}[x]/(x^n + 1)$, where n is a power of 2, then the $\mathcal{R}$-SIS$^{\mathcal{K}}_{q,1,m,\beta}$ problem is as hard as the $\tilde{\mathcal{O}}(\sqrt{n}\beta)$-SVP problem in all lattices that are ideals in $\mathcal{R}$ (where $\mathcal{K}$ is again the uniform distribution over $\mathcal{R}_q^{1\times m}$).

NTRU Lattices. In the NTRU cryptosystem over the ring $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$ [18], the key generation procedure picks two short secret keys $\mathbf{f}, \mathbf{g} \in \mathcal{R}_q$ (according to some distribution) and computes the public key as $\mathbf{a} = \mathbf{g}/\mathbf{f}$.[6] When the norm of $\mathbf{f}, \mathbf{g}$ is large enough, it can be shown that $\mathbf{a}$ is actually uniformly random in $\mathcal{R}_q$ [32], but even when the secret keys do not have enough entropy, their quotient still appears to be pseudorandom, although no proof of this fact is known [21]. In the NTRU cryptosystem (or its more secure modification of [32] which is based on the Ring-LWE problem), one encrypts a message μ, represented as a polynomial in $\mathcal{R}_q$ with $\{0, 1\}$ coefficients, by picking two short vectors

[6] In the original NTRU scheme, the ring was $\mathbb{Z}_q[x]/(x^n - 1)$, but lately researchers have also used $\mathbb{Z}_q[x]/(x^n + 1)$ when n is a power of 2. Indeed, the latter choice seems at least as secure.

$\mathbf{r}, \mathbf{e} \in \mathcal{R}_q$ and outputting $\mathbf{z} = 2(\mathbf{ar} + \mathbf{e}) + \mu$. The security of the scheme relies on the fact that the distribution of $(\mathbf{a}, \mathbf{z})$ is pseudo-random in $\mathcal{R}_q^2$.

One can define an NTRU version of the SIS problem that is at least as hard as breaking the NTRU cryptosystem.[7] In particular, given an NTRU public key $\mathbf{a}$, find two polynomials $\mathbf{v}_1, \mathbf{v}_2 \in \mathcal{R}_q$ such that $\|(\mathbf{v}_1|\mathbf{v}_2)\| \leqslant \beta$ and $\mathbf{av}_1 + \mathbf{v}_2 = 0$ in $\mathcal{R}_q$. Notice that $(\mathbf{f}, -\mathbf{g})$ is a solution to this problem, but in fact, finding larger solutions can also be useful in breaking the NTRU cryptosystem. In particular, notice that for any solution $(\mathbf{v}_1|\mathbf{v}_2)$, one can compute $\mathbf{zv}_1 = 2(-\mathbf{rv}_2 + \mathbf{ev}_1) + \mu\mathbf{v}_1$. If β is sufficiently small with respect to $\|(\mathbf{r}|\mathbf{e})\|$, then $\mathbf{z} \cdot \mathbf{v}_1 \bmod 2 = \mu\mathbf{v}_1$, and μ can be recovered. Thus, for certain parameters, the NTRU version of the SIS problem is at least as hard as breaking the NTRU cryptosystem. As a side-note, we would like to point out that the NTRU encryption scheme remains hard even after 15 years of cryptanalysis. The weakness in the NTRU signature scheme, which uses the same key generation procedure, is due to the fact that signatures slowly leak the secret key [29,6], but this is provably (*i.e.* information-theoretically) avoided in our scheme.

In the full version of this paper [5], we propose a practical instantiation of our signature scheme inspired by the NTRU key-generation, and analyze the hardness of the NTRU version of the SIS problem using combinations of lattice [3] and hybrid attacks [20]. We provide concrete parameters, and the resulting signature scheme was implemented as a proof-of-concept on a desktop computer (and yielded the timings of Table 1).

3 BLISS: A Lattice Signature Scheme Using Bimodal Gaussians

In this section, we present our new signature scheme along with the proof of correctness. The security of the signature scheme is based on the hardness of the $\mathcal{R}\text{-SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem. We mention that this is the "simple" version of our algorithm, and its more optimized implementation that uses numerous enhancements is presented in the full version of this paper [5]. For simplicity, we present our algorithm for $\mathcal{R} = \mathbb{Z}$, but it works in exactly the same way for rings $\mathcal{R} = \mathbb{Z}[x]/(x^n + 1)$.

3.1 New Signature and Verification Algorithms

Key pairs. The secret key is a (short) matrix $\mathbf{S} \in \mathbb{Z}_{2q}^{m \times n}$ and the public key is given by the matrix $\mathbf{A} \in \mathbb{Z}_{2q}^{n \times m}$ such that $\mathbf{AS} = q\mathbf{I}_n \pmod{2q}$. A crucial property, for our new rejection sampling algorithm, satisfied by the key pair, is that $\mathbf{AS} = \mathbf{A}(-\mathbf{S}) = q\mathbf{I}_n \pmod{2q}$. Obtaining such a key pair is easy and can be done efficiently. In the full version of this paper [5], we explain the key-generation procedure which results in a scheme whose security is based on the

[7] A way to state the NTRU SIS problem in terms of the $\mathcal{R}\text{-SIS}^{\mathcal{K}}_{q,1,2,\beta}$ problem is to set $\mathcal{R} = \mathbb{Z}[x]/(x^n + 1)$ and let $\mathcal{K}$ be the distribution that picks small $\mathbf{f}, \mathbf{g}$ and outputs the public key $\mathbf{A} = (\mathbf{a}, \mathbf{1}) \in \mathcal{R}_q^{1 \times 2}$ for $\mathbf{a} = \mathbf{g}/\mathbf{f}$.

classic $\mathrm{SIS}_{q,n,m,\beta}$ problem and we present an "NTRU-like" variant of the key generation which yields a more efficient instantiation of the signature scheme.

Random Oracle Domain. We model the hash function H as a random oracle that has uniform output in $\mathbb{B}_\kappa^n$, the set of binary vectors of length n and weight κ. Such a mapping can be found in [7] but its complexity is quadratic in n; in the full version of this paper, we provide an efficient construction.

Algorithm 1. Signature Algorithm

Input: Message μ, public key $\mathbf{A} \in \mathbb{Z}_{2q}^{n \times m}$, secret key $\mathbf{S} \in \mathbb{Z}_{2q}^{m \times n}$, stand. dev. $\sigma \in \mathbb{R}$
Output: A signature $(\mathbf{z}, \mathbf{c})$ of the message μ

1: $\mathbf{y} \leftarrow D_\sigma^m$
2: $\mathbf{c} \leftarrow H(\mathbf{A}\mathbf{y} \bmod 2q, \mu)$
3: Choose a random bit $b \in \{0, 1\}$
4: $\mathbf{z} \leftarrow \mathbf{y} + (-1)^b \mathbf{S}\mathbf{c}$
5: **Output**$(\mathbf{z}, \mathbf{c})$ with probability $1 \Big/ \left(M \exp\left(-\frac{\|\mathbf{S}\mathbf{c}\|^2}{2\sigma^2}\right) \cosh\left(\frac{\langle \mathbf{z}, \mathbf{S}\mathbf{c}\rangle}{\sigma^2}\right)\right)$ otherwise **restart**

Algorithm 2. Verification Algorithm

Input: Message μ, public Key $\mathbf{A} \in \mathbb{Z}_{2q}^n$, signature $(\mathbf{z}, \mathbf{c})$
Output: Accept or Reject the signature

1: **if** $\|\mathbf{z}\| \quad > B_2$ **then** Reject
2: **if** $\|\mathbf{z}\|_\infty \geqslant q/4$ **then** Reject
3: Accept iff $\mathbf{c} = H(\mathbf{A}\mathbf{z} + q\mathbf{c} \bmod 2q, \mu)$

The Signature Algorithm. The signer, who is given a message digest μ, first samples a vector $\mathbf{y}$ from the m-dimensional discrete Gaussian distribution D_σ^m and then computes $\mathbf{c} \leftarrow H(\mathbf{A}\mathbf{y} \bmod 2q, \mu)$. He then samples a bit b in $\{0, 1\}$ and computes the potential output $\mathbf{z} \leftarrow \mathbf{y} + (-1)^b \mathbf{S}\mathbf{c}$. Notice that $\mathbf{z}$ is distributed according to the bimodal discrete Gaussian distribution $\frac{1}{2} D_{\mathbf{S}\mathbf{c},\sigma}^m + \frac{1}{2} D_{-\mathbf{S}\mathbf{c},\sigma}^m$. At this point we perform rejection sampling and output the signature $(\mathbf{z}, \mathbf{c})$ with probability $1 \Big/ \left(M \exp\left(-\frac{\|\mathbf{S}\mathbf{c}\|^2}{2\sigma^2}\right) \cosh\left(\frac{\langle \mathbf{z}, \mathbf{S}\mathbf{c}\rangle}{\sigma^2}\right)\right)$, where M is some fixed positive real that is set large enough to ensure that the preceding probability is always at most 1. We explain how to set M in accordance with the standard deviation σ in the next section. If the signing algorithm did not output the signature, then it is restarted and repeated until something is output. The expected number of iterations of the signing algorithm is M.

The Verification Algorithm. The verification algorithm will accept $(\mathbf{z}, \mathbf{c})$ as the signature for μ if the following three conditions hold:

1. $\|\mathbf{z}\| \quad \leqslant B_2$
2. $\|\mathbf{z}\|_\infty < q/4$
3. $\mathbf{c} = H(\mathbf{A}\mathbf{z} + q\mathbf{c} \bmod 2q, \mu)$

The signer outputs signatures of the form $(\mathbf{z}, \mathbf{c})$ where $\mathbf{z}$ is distributed according to D_σ^m, thus the acceptance bound B_2 should be set a little bit higher than $\sqrt{m}\sigma$, which is the expected value around which the output of D_σ^m is tightly concentrated; denoting $B_2 = \eta\sqrt{m}\sigma$, one can set η so that $\|\mathbf{z}\| \leqslant B_2$ is verified with probability $1-2^{-\lambda}$ [24, Lemma 4.4] for the security parameter λ (in practice, $\eta \in [1.1, 1.4]$). For technical reasons in the security proof, we also need that $\|\mathbf{z}\|_\infty < q/4$, but this condition is usually verified whenever the first one is and does not restrict the manner in which we choose the parameters for the scheme. Condition 3 will also hold for valid signatures because

$$\begin{aligned}\mathbf{Az} + q\mathbf{c} &= \mathbf{A}(\mathbf{y} + (-1)^b\mathbf{Sc}) + q\mathbf{c} = \mathbf{Ay} + \big((-1)^b\mathbf{AS}\big)\mathbf{c} + q\mathbf{c} = \mathbf{Ay} + (q\mathbf{I}_n)\mathbf{c} + q\mathbf{c} \\ &= \mathbf{Ay} \bmod 2q.\end{aligned}$$

3.2 Rejection Sampling: Correctness and Efficiency

We now explain how to pick the standard deviation σ and positive real M so that the signing algorithm in the preceding section produces vectors $\mathbf{z}$ according to the distribution D_σ^m. Because $\mathbf{y}$ is distributed according to D_σ^m, it is easy to see that in Step 4 of the signing algorithm, $\mathbf{z}$ is distributed according to $g_{\mathbf{Sc}} = \frac{1}{2}D^m_{\mathbf{Sc},\sigma} + \frac{1}{2}D^m_{-\mathbf{Sc},\sigma}$ for fixed $\mathbf{Sc}$ and over the space of all $(b, \mathbf{y})$. Thus for any $\mathbf{z}^* \in \mathbb{R}^m$, we have

$$\begin{aligned}\Pr[\mathbf{z} = \mathbf{z}^*] &= \frac{1}{2}D^m_{\mathbf{Sc},\sigma}(\mathbf{z}^*) + \frac{1}{2}D^m_{-\mathbf{Sc},\sigma}(\mathbf{z}^*) \\ &= \frac{1}{2\rho_\sigma(\mathbb{Z}^m)}\exp\left(-\frac{\|\mathbf{z}^* - \mathbf{Sc}\|^2}{2\sigma^2}\right) + \frac{1}{2\rho_\sigma(\mathbb{Z}^m)}\exp\left(-\frac{\|\mathbf{z}^* + \mathbf{Sc}\|^2}{2\sigma^2}\right) \\ &= \frac{1}{2\rho_\sigma(\mathbb{Z}^m)}\exp\left(-\frac{\|\mathbf{z}^*\|^2}{2\sigma^2}\right)\exp\left(-\frac{\|\mathbf{Sc}\|^2}{2\sigma^2}\right)\left(e^{-\frac{\langle \mathbf{z}^*, \mathbf{Sc}\rangle}{\sigma^2}} + e^{\frac{\langle \mathbf{z}^*, \mathbf{Sc}\rangle}{\sigma^2}}\right) \\ &= \frac{1}{\rho_\sigma(\mathbb{Z}^m)}\exp\left(-\frac{\|\mathbf{z}^*\|^2}{2\sigma^2}\right)\exp\left(-\frac{\|\mathbf{Sc}\|^2}{2\sigma^2}\right)\cosh\left(\frac{\langle \mathbf{z}^*, \mathbf{Sc}\rangle}{\sigma^2}\right) .\end{aligned}$$

The desired output distribution is the centered Gaussian distribution $f(\mathbf{z}^*) = \rho_\sigma(\mathbf{z}^*)/\rho_\sigma(\mathbb{Z}^m)$. Thus, by Lemma 2.1, one should accept the sample $\mathbf{z}^*$ with probability:

$$p_{\mathbf{z}*} = \frac{f(\mathbf{z}^*)}{Mg_{\mathbf{Sc}}(\mathbf{z}^*)} = 1\Big/\left(M\exp\left(-\frac{\|\mathbf{Sc}\|^2}{2\sigma^2}\right)\cosh\left(\frac{\langle \mathbf{z}^*, \mathbf{Sc}\rangle}{\sigma^2}\right)\right),$$

where M is chosen large enough so that $p_{\mathbf{z}*} \leqslant 1$. Note that $\cosh(x) \geqslant 1$ for any x, so it suffices that

$$M = e^{\frac{1}{2\alpha^2}} \tag{2}$$

where α is such that $\sigma \geqslant \alpha \cdot \|\mathbf{Sc}\|$.

Bound on $\|\mathbf{Sc}\|$. Notice that if we fix the repetition rate M, then the standard deviation of the signature $\mathbf{z}$, and therefore also its size, only depend on the maximum possible norm of the vector $\mathbf{Sc}$. For this reason, it is important to obtain a bound as tight as possible on this product. Several upper bounds on $\|\mathbf{Sc}\|$ can be used such as $\|\mathbf{Sc}\| \leqslant \|\mathbf{c}\|_1 \cdot \|\mathbf{S}\| = \kappa \|\mathbf{S}\|$ (as in [24]) or $\|\mathbf{Sc}\| \leqslant s_1(\mathbf{S}) \cdot \|\mathbf{c}\| = s_1(\mathbf{S}) \cdot \sqrt{\kappa}$ where $s_1(\mathbf{S})$ is the singular norm of $\mathbf{S}$. Here we introduce a new measure of $\mathbf{S}$, adapted to the form of $\mathbf{c}$, which helps us achieve a tighter bound than with all previous methods. We believe that this norm and the technique for bounding it could be of independent interest.

Definition 3.1. *For any integer* κ*, we define* $N_\kappa \colon \mathbb{R}^{m \times n} \to \mathbb{R}$ *as:*

$$N_\kappa(\mathbf{X}) = \max_{\substack{I \subset \{1,\dots,n\} \\ \#I = \kappa}} \sum_{i \in I} \left(\max_{\substack{J \subset \{1,\dots,n\} \\ \#J = \kappa}} \sum_{j \in J} T_{i,j} \right) \quad \text{where } \mathbf{T} = \mathbf{X}^t \cdot \mathbf{X} \in \mathbb{R}^{n \times n} .$$

The following proposition states that $\sqrt{N_\kappa(\mathbf{S})}$ is also an upper bound for $\|\mathbf{Sc}\|$.

Proposition 3.2. *Let* $\mathbf{S} \in \mathbb{R}^{m \times n}$ *be a real matrix. For any* $\mathbf{c} \in \mathbb{B}^n_\kappa$*, we have* $\|\mathbf{Sc}\|^2 \leqslant N_\kappa(\mathbf{S})$.

In practice, we will use this upper bound to bound $\|\mathbf{Sc}\|$ and derive the parameters. Some secret keys $\mathbf{S}$ will be rejected according to the value of $N_\kappa(\mathbf{S})$, which is easily computable. In addition to the gain from the use of bimodal Gaussians, this new upper bound lowers the standard deviation σ by a factor $\approx \sqrt{\kappa}/2$ compared to [24].

3.3 Security of BLISS

Any existential forger against our signature scheme can solve the $\mathcal{R}\text{-SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem for $\beta = 2B_2$ where $\mathcal{K}$ is the distribution induced by the public-key generation algorithm.

Theorem 3.3. *Suppose there is a polynomial-time algorithm* $\mathcal{F}$ *which makes at most* s *queries to the signing oracle and* h *queries to the random oracle* H*, and succeeds in forging with non negligible probability* δ*. Then there exists a polynomial-time algorithm which can solve the* $\mathcal{R}\text{-SIS}^{\mathcal{K}}_{q,n,m,\beta}$ *problem for* $\beta = 2B_2$ *with probability* $\approx \frac{\delta^2}{2(h+s)}$*. Moreover the signing algorithm produces a signature with probability* $\approx 1/M$ *and the verifying algorithm accepts a signature produced by an honest signer with probability at least* $1 - 2^m$*.*

The proof of the theorem follows from standard arguments, and is simpler and tighter than the proof of [24]. In a nutshell, the fact that the distribution of the signatures in the scheme does not depend on the secret key means that the simulator can "sign" arbitrary messages without having the secret key by programming the random oracle. Then when the adversary produces a forgery, the simulator can extract a solution to the SIS problem. The proof is provided in the full version of this paper [5].

References

1. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: 28th Annual ACM Symposium on Theory of Computing, Philadephia, Pennsylvania, USA, May 22–24, pp. 99–108. ACM Press (1996)
2. Bellare, M., Palacio, A.: GQ and Schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 162–177. Springer, Heidelberg (2002)
3. Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better lattice security estimates. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 1–20. Springer, Heidelberg (2011)
4. Damgård, I., Pastro, V., Smart, N.P., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 643–662. Springer, Heidelberg (2012)
5. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal gaussians. Cryptology ePrint Archive (2013)
6. Ducas, L., Nguyen, P.Q.: Learning a zonotope and more: Cryptanalysis of ntrusign countermeasures. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 433–450. Springer, Heidelberg (2012)
7. Fischer, J.-B., Stern, J.: An efficient pseudo-random generator provably as secure as syndrome decoding. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 245–255. Springer, Heidelberg (1996)
8. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008)
9. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 1–17. Springer, Heidelberg (2013)
10. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Mitzenmacher, M. (ed.) 41st Annual ACM Symposium on Theory of Computing, Bethesda, Maryland, USA, May 31-June 2, pp. 169–178. ACM Press (2009)
11. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: In, R.E., Ladner, C. (eds.) 40th Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, pp. 197–206. ACM Press (2008)
12. Gentry, C., Szydlo, M.: Cryptanalysis of the revised NTRU signature scheme. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 299–320. Springer, Heidelberg (2002)
13. Goldreich, O., Goldwasser, S., Halevi, S.: Public-key cryptosystems from lattice reduction problems. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 112–131. Springer, Heidelberg (1997)
14. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: STOC, pp. 545–554 (2013)
15. Güneysu, T., Lyubashevsky, V., Pöppelmann, T.: Practical lattice-based cryptography: A signature scheme for embedded systems. In: Prouff, E., Schaumont, P. (eds.) CHES 2012. LNCS, vol. 7428, pp. 530–547. Springer, Heidelberg (2012)
16. Hoffstein, J., Howgrave-Graham, N., Pipher, J., Whyte, W.: Practical Lattice-Based Cryptography: NTRUEncrypt and NTRUSign. In: The LLL Algorithm: Survey and Applications. Information Security and Cryptography. Springer (2009)
17. Hoffstein, J., Pipher, J., Howgrave-Graham, N., Silverman, J.H., Whyte, W.: NTRUSIGN: Digital signatures using the NTRU lattice. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 122–140. Springer, Heidelberg (2003)

18. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: Buhler, J. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 267–288. Springer, Heidelberg (1998)
19. Hoffstein, J., Pipher, J., Silverman, J.H.: NSS: An NTRU lattice-based signature scheme. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 211–228. Springer, Heidelberg (2001)
20. Howgrave-Graham, N.: A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 150–169. Springer, Heidelberg (2007)
21. López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Karloff, H.J., Pitassi, T. (eds.) 44th Annual ACM Symposium on Theory of Computing, New York, NY, USA, May 19-22, pp. 1219–1234. ACM Press (2012)
22. Lyubashevsky, V.: Lattice-based identification schemes secure under active attacks. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 162–179. Springer, Heidelberg (2008)
23. Lyubashevsky, V.: Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 598–616. Springer, Heidelberg (2009)
24. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Heidelberg (2012)
25. Lyubashevsky, V., Micciancio, D.: Generalized compact Knapsacks are collision resistant. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. Part II, LNCS, vol. 4052, pp. 144–155. Springer, Heidelberg (2006)
26. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010)
27. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012)
28. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. SIAM J. Comput. 37(1), 267–302 (2007)
29. Nguyen, P.Q., Regev, O.: Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. Journal of Cryptology 22(2), 139–160 (2009)
30. Peikert, C., Rosen, A.: Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 145–166. Springer, Heidelberg (2006)
31. Rückert, M.: Lattice-based blind signatures. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 413–430. Springer, Heidelberg (2010)
32. Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 27–47. Springer, Heidelberg (2011)
33. von Neumann, J.: Various techniques used in connection with random digits. J. Research Nat. Bur. Stand., Appl. Math. Series 12, 36–38 (1951)

Learning with Rounding, Revisited

New Reduction, Properties and Applications*

Joël Alwen[1], Stephan Krenn[2], Krzysztof Pietrzak[3], and Daniel Wichs[4]

[1] ETH Zurich
alwenj@inf.ethz.ch
[2] IBM Research – Zurich
skr@zurich.ibm.com
[3] Institute of Science and Technology Austria
pietrzak@ist.ac.at
[4] Northeastern University
wichs@ccs.neu.edu

Abstract. The learning with rounding (LWR) problem, introduced by Banerjee, Peikert and Rosen at EUROCRYPT '12, is a variant of learning with errors (LWE), where one replaces random errors with deterministic rounding. The LWR problem was shown to be as hard as LWE for a setting of parameters where the modulus and modulus-to-error ratio are super-polynomial. In this work we resolve the main open problem and give a new reduction that works for a larger range of parameters, allowing for a polynomial modulus and modulus-to-error ratio. In particular, a smaller modulus gives us greater efficiency, and a smaller modulus-to-error ratio gives us greater security, which now follows from the worst-case hardness of GapSVP with polynomial (rather than super-polynomial) approximation factors.

As a tool in the reduction, we show that there is a "lossy mode" for the LWR problem, in which LWR samples only reveal partial information about the secret. This property gives us several interesting new applications, including a proof that LWR remains secure with weakly random secrets of sufficient min-entropy, and very simple constructions of deterministic encryption, lossy trapdoor functions and reusable extractors.

Our approach is inspired by a technique of Goldwasser et al. from ICS '10, which implicitly showed the existence of a "lossy mode" for LWE. By refining this technique, we also improve on the parameters of that work to only requiring a polynomial (instead of super-polynomial) modulus and modulus-to-error ratio.

Keywords: Learning with Errors, Learning with Rounding, Lossy Trapdoor Functions, Deterministic Encryption.

* This work was partly funded by the European Research Council under ERC Starting Grant 259668-PSPC and ERC Advanced Grant 321310-PERCY. Parts of this work were done while the second authors was at IST Austria, and the last author was at IBM Research, T.J. Watson. A full version of this paper is available online [1].

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 57–74, 2013.

1 Introduction

Learning With Errors. The Learning with Errors (LWE) assumption states that "noisy" inner products of a secret vector with random public vectors, look pseudorandom. In the last years many cryptosystems have been proven secure under LWE, including (identity-based, leakage-resilient, fully homomorphic, functional) encryption [2–9], pseudorandom functions [10], (blind) signature schemes [3, 11–13], hash functions [14, 15], oblivious transfer [16], etc..

The LWE assumption with parameters $n, m, q \in \mathbb{N}$ and a "small" error distribution χ over $\mathbb{Z}$ states that for uniformly random $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$ and an error vector $\mathbf{e} \leftarrow \chi^m$

$$(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e}) \text{ is computationally indistinguishable from } (\mathbf{A}, \mathbf{u}).$$

Sometimes it will be convenient to think of this distribution as consisting of m "LWE samples" of the form $(\mathbf{a}_i, \langle \mathbf{a}_i, s \rangle + e_i) \in \mathbb{Z}_q^{n+1}$. One of the main advantages of the LWE problem is that, for some settings of parameters, we can prove its security under certain worst-case hardness assumptions over lattices, cf. [2, 17]. One important parameter is the "size" of the error terms $e \xleftarrow{\$} \chi$ which we denote by β.[1] As long as β exceeds some minimum threshold $\approx \sqrt{n}$, the concrete hardness of the LWE problem mainly depends on the dimension n and on the ratio of the modulus q to the error-size β. Therefore, we will often be unspecific about the exact distribution χ, and only focus on the error-size β.

Learning With Rounding. The Learning with Rounding (LWR) problem was introduced in [10]. Instead of adding a small random error to a sample $\langle \mathbf{a}, \mathbf{s} \rangle \in \mathbb{Z}_q$ to hide its exact value, we release a *deterministically rounded* version of $\langle \mathbf{a}, \mathbf{s} \rangle$. That is, for some $p < q$, we divide up the elements of $\mathbb{Z}_q$ into p contiguous intervals of roughly q/p elements each and define the *rounding function* $\lfloor \cdot \rfloor_p : \mathbb{Z}_q \to \mathbb{Z}_p$ that maps $x \in \mathbb{Z}_q$ into the index of the interval that x belongs to. For example if q, p are both powers of 2, than this could correspond to outputting the $\log(p)$ most significant bits of x. We can extend the rounding function to vectors by applying it component-wise. The LWR assumption states that:

$$(\mathbf{A}, \lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p) \text{ is computationally indistinguishable from } (\mathbf{A}, \lfloor \mathbf{u} \rfloor_p).$$

Note that if p divides q, then $\lfloor \mathbf{u} \rfloor_p$ is itself uniform over $\mathbb{Z}_p^m$.

The main advantage of LWR is that one does not need to sample any additional "errors", therefore requiring fewer random bits. The assumption has been used to construct simple and efficient pseudorandom generators and functions in [10], and deterministic encryption in [18].

Banerjee et al. [10] show a beautifully simple reduction proving the hardness of the LWR problem under the LWE assumption for some range of parameters. They observe that if the error size β is sufficiently small and the ratio q/p is sufficiently big, then $\lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rfloor_p = \lfloor \langle \mathbf{a}, \mathbf{s} \rangle + e \rfloor_p$ with overwhelming probability over

[1] We will be informal for now; we can think of β as the the standard deviation or the expected/largest absolute value of the errors.

random $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q$ and $e \xleftarrow{\$} \chi$. In particular, the only way that the two values differ is if $\langle \mathbf{a}, \mathbf{s} \rangle$ ends up within a distance of $|e|$ from a boundary between two different intervals; but since the intervals are of size q/p and the ball around the boundary is only of size $2|e|$ this is unlikely to happen when q/p is super-polynomially bigger than $2|e|$. Therefore, one can show that:

$$(\mathbf{A}, \lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p) \overset{\text{stat}}{\approx} (\mathbf{A}, \lfloor \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \rfloor_p) \overset{\text{comp}}{\approx} (\mathbf{A}, \lfloor \mathbf{u} \rfloor_p)$$

where the first modification is statistically close and the second follows immediately from the hardness of LWE.

Unfortunately, the argument only goes through, when (q/p) is bigger than the error size β by a super-polynomial factor. In fact, if we want statistical distance $2^{-\lambda}$ we would need to set $q \geq 2^{\lambda} \beta p$, where λ is a security parameter. This has three important consequences: (1) the modulus q has to be super-polynomial, which makes all of the computations less efficient, (2) the modulus-to-error ratio q/β is super-polynomial which makes the LWE problem easier and only gives us a reduction if we assume the hardness of the lattice problem GapSVP with super-polynomial approximation factors (a stronger assumption), (3) the ratio of the input-to-output modulus q/p is super-polynomial, meaning that we must "throw away" a lot of information when rounding and therefore get fewer bits of output per LWR sample. The work of [10] conjectured that the LWR problem should be hard even for a polynomial modulus q, but left it as the main open problem to give a reduction. The conjecture is especially interesting in light of the recent results of [19] which give the first *classical* reduction from LWE with small parameters to GapSVP.

1.1 The New Reduction and Properties of LWR

LWR with Polynomial Modulus. In this work, we resolve the open problem of [10] and give a new reduction showing the hardness of LWR from that of LWE for a more general setting of parameters, including when the modulus q is only polynomial. In particular, instead of requiring $q \geq 2^{\lambda} \beta p$, where λ is a security parameter as in [10], we only require $q \geq nm\beta p$, where we recall that n is the dimension of the secret, and m is the number of LWR samples that we output, β is the size of the LWE errors, and p is the new modulus we round to. In particular, as long as the number of LWR samples m is fixed a-priori by some polynomial, we can allow the modulus q (and therefore also the modulus-to-error ratio q/β, and the input-to-output ratio q/p) to all be polynomial. As mentioned, this setting provides greater efficiency (computation with smaller q) and greater security (smaller ratio q/β) allowing for a reduction from the worst-case hardness of the lattice problem GapSVP with polynomial approximation factors. In particular, the above efficiency and security improvements for LWR directly translate into improvements of the PRG and PRF constructions of [10].

To be even more precise, our reduction shows the hardness of LWR with parameters n, m, q, p assuming the hardness of LWE with parameters n', m, q, β (note: different dimension n' vs. n) as long as:

$$n \geq \frac{\log(q)}{\log(2\gamma)} \cdot n' \qquad \text{and} \qquad q \geq \gamma(nm\beta p) \tag{1}$$

for some flexible parameter $\gamma \geq 1$. For example, setting $\gamma = 1$ allows for the smallest modulus $q \approx nm\beta p$, but requires a larger dimension $n \approx n' \log(q)$ in the LWR problem than the dimension n' of the underlying LWE assumption. On the other hand, setting $\gamma = q^{\delta}$ for some constant $\delta \in (0, 1)$ gives a bigger polynomial modulus $q \approx (nm\beta p)^{1/(1-\delta)}$ but allow us to set the LWR dimension $n \approx (1/\delta)n' = O(n')$ to be closer to that of the underlying LWE assumption.

It remains as an open problem to improve the reduction further, and especially to remove the dependence between the modulus q and the number of LWR samples m that we give out.

LWR with Weak and Leaky Secrets. Another advantage of our reduction is that we prove the security of the LWR problem even when the secret $\mathbf{s}$ is not necessarily uniform over $\mathbb{Z}_q^n$. Indeed, our proof also works when $\mathbf{s}$ is uniform over a smaller integer interval $\mathbf{s} \xleftarrow{\$} \{-\gamma, \ldots, \gamma\}^n \subseteq \mathbb{Z}_q^n$, where the relation of $\gamma \geq 1$ to the other parameters is given by equation (1). Moreover, our reduction works when the secret $\mathbf{s}$ is not even truly uniform over this interval (say, because the attacker observed some leakage on $\mathbf{s}$, or $\mathbf{s}$ was sampled using a weak random source) as long as $\mathbf{s}$ retains some sufficiently high amount of *min-entropy* $k \approx n' \log(q)$, where n' is the dimension of the underlying LWE assumption. Notice that, no matter how small the entropy k is, we can still prove some level of security under an LWE assumption with correspondingly smaller dimension n'.

The work of Goldwasser et al. [20] shows similar results for the hardness of LWE with a weak and leaky secret, at least as long as the modulus q and the modulus-to-error ratio q/β are super-polynomial. Indeed, we will use a refinement of the technique from their work as the basis of our LWE to LWR reduction. Our refinement will also allow us to improve the parameters of [20], and show the hardness of LWE with a weak and leaky secret when the modulus q and the ratio q/β are polynomial.

The Reduction. As discussed above, the original reduction of [10] required us to choose parameters so that rounding samples with and without error is almost always identical: $\Pr[\lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rfloor_p \neq \lfloor \langle \mathbf{a}, \mathbf{s} \rangle + e \rfloor_p] \leq \mathsf{negl}$. Therefore LWR outputs do not provide any more information than LWE outputs. In contrast, in our setting of parameters, when q is polynomial, there is a noticeable probability that the two values are different. We therefore need a completely different proof strategy.

Surprisingly, our strategy does *not* directly convert an LWE instance with secret $\mathbf{s}$ into an LWR instance with secret $\mathbf{s}$. Instead, we rely on the LWE problem to change the distribution of the coefficient matrix $\mathbf{A}$. In particular, we show that there is a "lossy" method of sampling a matrix $\tilde{\mathbf{A}} \xleftarrow{\$} \mathsf{Lossy}()$ such that:

(a) Under the LWE assumption, $\tilde{\mathbf{A}} \xleftarrow{\$} \mathsf{Lossy}()$ is computationally indistinguishable from $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$.

(b) When $\tilde{\mathbf{A}} \xleftarrow{\$} \mathsf{Lossy}()$, the values $\tilde{\mathbf{A}}, \lfloor \tilde{\mathbf{A}} \cdot \mathbf{s} \rfloor_p$ do not reveal too much information about $\mathbf{s}$. In particular, $\mathbf{s}$ maintains a large fraction of its statistical entropy given $\tilde{\mathbf{A}}, \lfloor \tilde{\mathbf{A}} \cdot \mathbf{s} \rfloor_p$.

Before we describe how the $\mathsf{Lossy}()$ sampler works in the next paragraph, let us show that the above two properties allow us to prove the hardness of LWR problem. We can do so via a hybrid argument where, given many LWR samples, we replace one sample at a time from being an LWR sample to being uniformly random. In particular, assume we have $m+1$ LWR samples and let the matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$ denote the coefficient vectors of the first m samples, and let $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n$ be the coefficient vector of the last sample. Then we can show:

$$\left(\begin{bmatrix}\mathbf{A}\\ \mathbf{a}\end{bmatrix}, \begin{bmatrix}\lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p \\ \lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rfloor_p\end{bmatrix}\right) \overset{\text{comp}}{\approx} \left(\begin{bmatrix}\tilde{\mathbf{A}}\\ \mathbf{a}\end{bmatrix}, \begin{bmatrix}\lfloor \tilde{\mathbf{A}} \cdot \mathbf{s} \rfloor_p \\ \lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rfloor_p\end{bmatrix}\right) \overset{\text{stat}}{\approx}$$
$$\left(\begin{bmatrix}\tilde{\mathbf{A}}\\ \mathbf{a}\end{bmatrix}, \begin{bmatrix}\lfloor \tilde{\mathbf{A}} \cdot \mathbf{s} \rfloor_p \\ \lfloor u \rfloor_p\end{bmatrix}\right) \overset{\text{comp}}{\approx} \left(\begin{bmatrix}\mathbf{A}\\ \mathbf{a}\end{bmatrix}, \begin{bmatrix}\lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p \\ \lfloor u \rfloor_p\end{bmatrix}\right)$$

In the first step, we use the LWE assumption to replace a uniformly random $\mathbf{A}$ by a lossy matrix $\tilde{\mathbf{A}} \xleftarrow{\$} \mathsf{Lossy}()$, but still choose the last row $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n$ at random. In the second step, we use the fact that *inner product* is a strong extractor, where we think of the secret $\mathbf{s}$ as the source and the vector $\mathbf{a}$ as a seed. In particular, by the properties of the lossy sampler, we know that $\mathbf{s}$ maintains entropy conditioned on seeing $\tilde{\mathbf{A}}, \lfloor \tilde{\mathbf{A}} \cdot \mathbf{s} \rfloor_p$ and therefore the "extracted value" $\langle \mathbf{a}, \mathbf{s} \rangle$ is statistically close to a uniformly random and independent $u \xleftarrow{\$} \mathbb{Z}_q$. In the last step, we simply replace the lossy matrix $\tilde{\mathbf{A}} \xleftarrow{\$} \mathsf{Lossy}()$ back by a uniformly random $\mathbf{A}$. This shows that, given the first m LWR samples the last one looks uniform and independent. We can then repeat the above steps m more times to replace each of the remaining LWR samples (rows) by uniform, one-by-one.

The Lossy Sampler. The basic idea of our Lossy sampler is taken from the work of Goldwasser et al. [20]. We sample the lossy matrix $\tilde{\mathbf{A}} \in \mathbb{Z}_q^{m \times n}$ as

$$\tilde{\mathbf{A}} \overset{\text{def}}{=} \mathbf{B}\mathbf{C} + \mathbf{F} \qquad \text{where } \mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{m \times n'} \;\;,\;\; \mathbf{C} \xleftarrow{\$} \mathbb{Z}_q^{n' \times n} \;\;,\;\; \mathbf{F} \xleftarrow{\$} \chi^{m \times n}$$

where $n' < n$ is some parameter and χ is a "small" LWE error distribution. We now need to show that this satisfies the properties (a) and (b) described above.

It is easy to see that $\tilde{\mathbf{A}}$ is computationally indistinguishable from a uniformly random matrix under the LWE assumption with parameters n', m, q, χ. In particular, each column i of the matrix $\tilde{\mathbf{A}}$ can be thought of as an LWE distribution $\mathbf{B} \cdot \mathbf{c}_i + \mathbf{f}_i$ with coefficient matrix $\mathbf{B}$, secret $\mathbf{c}_i$ which is the ith column of the matrix $\mathbf{C}$, and error vector $\mathbf{f}_i$ which is the ith column of $\mathbf{F}$. Therefore, using n hybrid arguments, we can replace each column i of $\tilde{\mathbf{A}}$ by a uniformly random and independent one. This part of the argument is the same as in [20].

Next, we need to show that the secret $\mathbf{s}$ retains entropy even conditioned on seeing $\tilde{\mathbf{A}}, \lfloor \tilde{\mathbf{A}} \cdot \mathbf{s} \rfloor_p$. Let us first prove this property in the case when $\mathbf{s} \xleftarrow{\$} \{-1, 0, 1\}^n$

is itself a random "short" vector.[2] All of the information that we give out about $\mathbf{s}$ can be reconstructed from:

- The matrices $\mathbf{B}, \mathbf{C}, \mathbf{F}$ which define $\tilde{\mathbf{A}}$ and are independent of $\mathbf{s}$ on their own.
- The value $\mathbf{C} \cdot \mathbf{s}$ whose bit-length is $n' \log(q)$.
- A set Z consisting of all pairs $(i, v_i) \in [m] \times \mathbb{Z}_p$ such that $\lfloor (\mathbf{BC} \cdot \mathbf{s})_i \rfloor_p \neq \lfloor (\tilde{\mathbf{A}} \cdot \mathbf{s})_i \rfloor_p$ along with the value $v_i = \lfloor (\tilde{\mathbf{A}} \cdot \mathbf{s})_i \rfloor_p$. The subscript i denotes the i^{th} component of a vector.

Given the three pieces of information above, we can reconstruct $\tilde{\mathbf{A}}, \lfloor \tilde{\mathbf{A}} \cdot \mathbf{s} \rfloor_p$ by setting $\lfloor (\tilde{\mathbf{A}} \cdot \mathbf{s})_i \rfloor_p := \lfloor (\mathbf{BC} \cdot \mathbf{s})_i \rfloor_p$ for every index i not contained in Z, and setting $\lfloor (\tilde{\mathbf{A}} \cdot \mathbf{s})_i \rfloor_p := v_i$ for every i which is in Z. Therefore, we just need to show that the three pieces of information above do not reveal too much about $\mathbf{s}$. First, we show that the set Z is small with overwhelming probability. In particular, an index i is contained in Z if and only if

$$\lfloor (\mathbf{BC} \cdot \mathbf{s})_i \rfloor_p \neq \lfloor (\mathbf{BC} \cdot \mathbf{s})_i + (\mathbf{F} \cdot \mathbf{s})_i \rfloor_p. \tag{2}$$

Assume that the entries of the error matrix $\mathbf{F}$ are all bounded by β in absolute value with overwhelming probability, and therefore $(\mathbf{F} \cdot \mathbf{s})_i$ is bounded by $n\beta$ in absolute value.[3] Then the event (2) can only occur if the value $(\mathbf{BC} \cdot \mathbf{s})_i$ falls within distance $n\beta$ of a boundary between two different intervals. Since each interval is of size $\approx q/p$ and the ball around each boundary is of size $2n\beta$, this happens with (noticeable but small) probability $\leq 2n\beta p/q \leq 1/m$, when $q \geq 2nm\beta p$ (which gives us the bound of (1)). Therefore, the probability of any index i being in Z is at most $1/m$, the expected size of Z is at most 1, and because these probabilities are independent, we can use Chernoff to bound $|Z| \leq n'$ with overwhelming probability $1 - 2^{-n'}$. So in total, Z can be described by $|Z|(\log m + \log p) \leq n' \log q$ bits with overwhelming probability. Therefore, together, $Z, \mathbf{Cs}$ reveal only $O(n' \log q)$ bits of information about $\mathbf{s}$, even given $\mathbf{B}, \mathbf{C}, \mathbf{F}$. We can summarize the above as:

$$\begin{aligned} H_\infty(\mathbf{s} | \tilde{\mathbf{A}}, \lfloor \tilde{\mathbf{A}}\mathbf{s} \rfloor_p) &\geq H_\infty(\mathbf{s} | \mathbf{B}, \mathbf{C}, \mathbf{F}, \mathbf{C} \cdot \mathbf{s}, Z) \\ &\geq H_\infty(\mathbf{s} | \mathbf{B}, \mathbf{C}, \mathbf{F}) - O(n' \log q) \geq n - O(n' \log q). \end{aligned}$$

Hence, if n is sufficiently larger than some $O(n' \log q)$, the LWR secret maintains a large amount of entropy given the LWR samples with a lossy $\tilde{\mathbf{A}}$. The above analysis also extends to the case where $\mathbf{s}$ is not uniformly random, but only has a sufficient amount of entropy.

We can also extend the above analysis to the case where $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ is uniformly random over the entire space (and not short), by thinking of $\mathbf{s} = \mathbf{s}_1 + \mathbf{s}_2$ where $\mathbf{s}_1 \xleftarrow{\$} \mathbb{Z}_q^n$ is uniformly random and $\mathbf{s}_2 \xleftarrow{\$} \{-1, 0, 1\}^n$ is random and short. Using the same argument as above, we can show that, even given $\mathbf{s}_1$, $\tilde{\mathbf{A}}$ and $\lfloor \tilde{\mathbf{A}} \cdot \mathbf{s} \rfloor_p$, the value $\mathbf{s}_2$ (and therefore also $\mathbf{s}$) maintains entropy.

[2] This proof generalizes to larger intervals $\{-\gamma, \ldots, \gamma\}$ and corresponds to the parameter γ in equation (1). Here we set $\gamma = 1$.

[3] Our actual proof is more refined and only requires us to bound the *expected* absolute value of the entries.

Our analysis of lossiness as described above is inspired by [20] but differs from it significantly. In particular that work considered LWE (not LWR) samples with the matrix $\tilde{\mathbf{A}}$, did not explicitly analyze lossiness, and required super-polynomial modulus and modulus-to-error ratio. Indeed, in the full version [1] we use the ideas from the above analysis to also improve the parameters of that work, showing the robustness of the LWE problem to weak and leaky secrets for a polynomial modulus and modulus-to-error ratio.

1.2 Applications

Reusable Computational Extractors. By the leftover-hash lemma, the function $\mathsf{Ext}(\mathbf{s}; \mathbf{a}) := \langle \mathbf{s}, \mathbf{a} \rangle$ is a good randomness extractor taking a secret source $\mathbf{s} \in \mathbb{Z}_q^n$ of min-entropy $k \geq \log(q) + 2\log(1/\varepsilon)$ and a random public seed $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n$, and its output will be ε-close to the uniform over $\mathbb{Z}_q$. But assume we want to extract many different mutually (pseudo-)random values from the source $\mathbf{s}$ without keeping any long term state: each time we want to extract a new output we choose a fresh seed and apply the extractor. It is easy to see that the above inner-product extractor is completely insecure after at most n applications, and each successive output is easy to predict from the previous ones. The work of [21] introduced the notion of a *reusable computational extractor* that remains secure even after m applications, where m can be an arbitrary polynomial, and gave a construction under a non-standard "learning-subspaces with noise" assumption. Our results immediately give us a new simple construction of reusable extractors defined by $\mathsf{Ext}(\mathbf{s}; \mathbf{a}) := \lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rfloor_p$. That is, we just round the output of the standard inner product extractor! We show that, as long as the LWE assumption holds with some parameters n', m, q, β, the source $\mathbf{s}$ is distributed over $\{0,1\}^n$ and has entropy $k \geq O(n' \log(q))$, and the modulus satisfies $q \geq 2\beta nmp$, the above extractor is secure for m uses. In particular, we can have $m \gg n \gg k$.

Lossy Trapdoor Functions. Lossy trapdoor functions (LTDFs) [22,23] are a family of functions $f_{pk}(\cdot)$ keyed by some public key pk, which can be sampled in one of two indistinguishable modes: `injective` and `lossy`. In the `injective` mode the function $f_{pk}(\cdot)$ is an injective function and we can even sample pk along with a secret trapdoor key sk that allows us to invert it efficiently. In the `lossy` mode, the function $f_{pk}(\cdot)$ is "many-to-one" and $f_{pk}(\mathbf{s})$ statistically loses information about the input $\mathbf{s}$. LTDFs have many amazing applications in cryptography, such as allowing us to output many hardcore bits, construct CCA-2 public-key encryption [23,24], and deterministic encryption [25]. We construct very simple and efficient LTDFs using the LWR problem: the public key is a matrix $pk = \mathbf{A}$ and the function is defined as $f_{\mathbf{A}}(\mathbf{s}) = \lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p$. We can sample an injective $\mathbf{A}$ with a trapdoor using the techniques of Ajtai [26] or subsequent improvements [27,28], and one can sample a lossy $\mathbf{A}$ using our lossy sampler. Although prior constructions of LTDFs based on LWE are known [23,29], our construction is extremely simple and has the advantage that our lossy mode loses "almost all" of the information contained in $\mathbf{s}$.

Deterministic Encryption. Deterministic public-key encryption [25,30–33] is intended to guarantee security as long as the messages have sufficient entropy. Although there are black-box constructions of deterministic encryption using LTDFs [32], we get a very simple direct construction from the LWR problem: the public key is a matrix $pk = \mathbf{A} \in \mathbb{Z}_q^{m \times n}$, and to encrypt a message $\mathbf{s} \in \{0,1\}^n$, we simply output $\lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p$. We can sample $\mathbf{A}$ with a decryption trapdoor using the standard techniques [26–28] mentioned previously. Our analysis here is essentially the same as for our reusable extractor – we simply note that whenever $\mathbf{s}$ has sufficient entropy, the output $\lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p$ is pseudorandom. We note that the same construction was proposed by Xie et al. [18], but because the analysis there was similar to [10,20], they required a super-polynomial modulus and modulus-to-error ratio. The main advantage of this scheme over other deterministic encryption schemes is that we do not need any fixed threshold on the entropy of the message $\mathbf{s}$: no matter how low it is we can still prove security under an LWE assumption with correspondingly degraded parameters.

1.3 Recent Concurrent Work

The work of [34] studies the security of LWE in the case where the error distribution is uniformly random over a small interval. In appendix B of the full version [1], we derive a very similar result. As a tool, both works rely on studying a "lossy mode" of LWE, but the construction and analysis are somewhat different. The work of [35] also studies LWE in a setting with extremely small errors uniform over $\{0,1\}$ also crucially using the notion of lossiness.

2 Preliminaries

Notation. Throughout, we let λ denote the *security parameter*. We use bold lower-case letters (e.g., $\mathbf{s}, \mathbf{e}$) to denote vectors, and bold upper-case letters (e.g., $\mathbf{A}, \mathbf{B}$) to denote matrices. If X is a distribution or a random variable, we write $x \xleftarrow{\$} X$ to denote the process of sampling x according to X. If X is a set, we write $x \xleftarrow{\$} X$ to denote the process of sampling x *uniformly* at random over X. For two distribution ensembles $X = \{X_\lambda\}, Y = \{Y_\lambda\}$, we write $X \overset{\text{comp}}{\approx} Y$ if for all probabilistic polynomial time (PPT) distinguishers D there is a negligible function $\mathsf{negl}(\cdot)$ such that: $|\Pr[D(1^\lambda, X_\lambda) = 1] - \Pr[D(1^\lambda, Y_\lambda)] = 1| \leq \mathsf{negl}(\lambda)$.

Bounded Distribution. A distribution χ over $\mathbb{R}$ is called β-*bounded* if $\mathbb{E}[|\chi|] \leq \beta$.

Probabilistic Notions. We assume that the reader is familiar with some basic notions from probability, such as statistical distance Δ, (conditional) min entropy, and the Chernoff bound. We will further rely on the following less standard definition of *smooth min-entropy*, which was first introduced by Renner and Wolf [36]. Intuitively, a random variable has high smooth min-entropy, if it is statistically close to a random variable with high min-entropy.

Definition 2.1 (Smooth Entropy). *We say that a random variable X has ε-*smooth min-entropy at least k*, denoted by $H^{\varepsilon}_{\infty}(X) \geq k$, if there exists some variable X' such that $\Delta(X, X') \leq \varepsilon$ and $H_{\infty}(X') \geq k$. Similarly, we say that the ε-*smooth conditional min-entropy of X given Y is at least k*, denoted $H^{\varepsilon}_{\infty}(X|Y) \geq k$ if there exist some variables (X', Y') such that $\Delta((X, Y), (X', Y')) \leq \varepsilon$ and $H_{\infty}(X'|Y') \geq k$.*

We will write $H^{\mathsf{smooth}}_{\infty}(\cdot)$ to denote $H^{\varepsilon}_{\infty}(\cdot)$ for some (unspecified) negligible ε.

2.1 Learning with Errors and Learning with Rounding

Learning With Errors. The *decisional learning with errors* (*LWE*) problem was first introduced by Regev [2]. Informally, the problem asks to distinguish slightly perturbed random linear equations from truly random ones.

Definition 2.2 (LWE Assumption [2]). *Let λ be the security parameter, $n = n(\lambda), m = m(\lambda), q = q(\lambda)$ be integers and let $\chi = \chi(\lambda)$ be a distribution over $\mathbb{Z}_q$. The* $\mathrm{LWE}_{n,m,q,\chi}$ *assumption says that for $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$, $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$ the following distributions are computationally indistinguishable:*

$$(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e}) \overset{\text{comp}}{\approx} (\mathbf{A}, \mathbf{u}).$$

It has been shown that the LWE-assumption holds for certain error distributions χ, assuming the worst-case hardness of certain lattice problems. In particular, this is the case if χ is a discrete Gaussian distribution with appropriate variance, see, e.g., [2, 17, 35] for precise statements.

Learning With Rounding. The *learning with rounding* (*LWR*) problem was introduced by Banerjee et al. [10]. It can, in some sense, be seen as a derandomized version of the LWE-problem. The idea is to compute the error terms deterministically: instead of perturbing the answer by adding a small error, we simply round the answer – in both cases we are intuitively hiding the low order bits.

More formally, the LWR-problem is defined via the following *rounding function* for integers $q \geq p \geq 2$:

$$\lfloor \cdot \rfloor_p \ : \ \mathbb{Z}_q \to \mathbb{Z}_p \ : \ x \mapsto \lfloor (p/q) \cdot x \rfloor,$$

where we naturally identify elements of $\mathbb{Z}_k$ with the integers in the interval $\{0, \ldots, k-1\}$.[4] More intuitively, $\lfloor . \rfloor_p$ partitions $\mathbb{Z}_q$ into intervals of length $\approx \frac{q}{p}$ which it maps to the same image. We naturally extend the rounding function to vectors over $\mathbb{Z}_q$ by applying it component-wise.

In the presentation of our results we will make use that the probability that a random element in $\mathbb{Z}_q$ is close to a step in the rounding function is small. We therefore define, for any integer $\tau > 0$:

$$\mathsf{border}_{p,q}(\tau) \overset{\text{def}}{=} \{x \in \mathbb{Z}_q \ : \ \exists y \in \mathbb{Z}, |y| \leq \tau, \lfloor x \rfloor_p \neq \lfloor x + y \rfloor_p\}.$$

[4] The choice of the floor function rather than ceiling or nearest integer is arbitrary and unimportant.

We can easily bound the probability of a random element being on the border. As for the rest of this document, we omit a proof and refer to the full version [1].

Lemma 2.3. *For every p, q, τ it holds that* $\Pr_{x \xleftarrow{\$} \mathbb{Z}_q}[x \in \mathsf{border}_{p,q}(\tau)] \leq \frac{2\tau p}{q}$.

The learning with rounding problem is now defined as follows:

Definition 2.4 (LWR [10]). *Let λ be the security parameter, $n = n(\lambda), m = m(\lambda), q = q(\lambda), p = p(\lambda)$ be integers. The* $\mathrm{LWR}_{n,m,q,p}$ *problem states that for $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$ the following distributions are computationally indistinguishable:* $(\mathbf{A}, \lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p) \overset{\text{comp}}{\approx} (\mathbf{A}, \lfloor \mathbf{u} \rfloor_p)$.

Notice that when p divides q, the distribution $\lfloor u \rfloor_p : u \xleftarrow{\$} \mathbb{Z}_q$ is just the uniform over $\mathbb{Z}_p$. Otherwise, the distribution is slightly skewed with some values in $\mathbb{Z}_p$ having probability $\frac{\lfloor q/p \rfloor}{q}$ and others $\frac{\lceil q/p \rceil}{q}$. However, it is easy to deterministically extract random bits from such independent samples with an asymptotic rate of $O(\log(p))$ bits per sample. Therefore, independent samples from the skewed distribution are often "good enough" in practice.

We also define a variant of the LWR assumption where the secret $\mathbf{s}$ can come from some *weak source of entropy* and the attacker may observe some *partial leakage* about $\mathbf{s}$.

Definition 2.5 (LWR with Weak and Leaky Secrets). *Let λ be the security parameter and n, m, q, p be integer parameters as in Definition 2.4. Let $\gamma = \gamma(\lambda) \in (0, q/2)$ be an integer and $k = k(\lambda)$ be a real. The* $\mathrm{LWR}^{\mathsf{WL}(\gamma,k)}_{n,m,q,p}$ *problem says that for any efficiently samplable correlated random variables $(\mathbf{s}, \mathsf{aux})$, where the support of $\mathbf{s}$ is the integer interval $[-\gamma, \gamma]^n$ and $H_\infty(\mathbf{s}|\mathsf{aux}) \geq k$, the following distributions are computationally indistinguishable:*

$$(\mathsf{aux}, \mathbf{A}, \lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p) \overset{\text{comp}}{\approx} (\mathsf{aux}, \mathbf{A}, \lfloor \mathbf{u} \rfloor_p)$$

where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$, $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$ are chosen randomly and independently of $\mathbf{s}, \mathsf{aux}$.

3 Lossy Mode for LWR

We now show that, under the LWE assumption, the LWR problem has a '*lossy mode*': we can sample a matrix $\tilde{\mathbf{A}}$ which is computationally indistinguishable from a uniformly random $\mathbf{A}$ such that the tuple $(\tilde{\mathbf{A}}, \lfloor \tilde{\mathbf{A}}\mathbf{s} \rfloor_p)$ does not reveal too much information about the secret $\mathbf{s}$.

Definition 3.1 (Lossy Sampler). *Let $\chi = \chi(\lambda)$ be an efficiently samplable distribution over $\mathbb{Z}_q$. The efficient* lossy sampler $\mathsf{Lossy}()$ *is given by:*

$\mathsf{Lossy}(1^n, 1^m, 1^\ell, q, \chi)$**:** *Sample $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{m \times \ell}, \mathbf{C} \xleftarrow{\$} \mathbb{Z}_q^{\ell \times n}, \mathbf{F} \xleftarrow{\$} \chi^{m \times n}$ and output $\tilde{\mathbf{A}} = \mathbf{B} \cdot \mathbf{C} + \mathbf{F}$.*

Although the matrix $\tilde{\mathbf{A}}$ computed by the Lossy algorithm is *statistically* far from a uniformly random matrix, it is easy to show that it is computationally indistinguishable from one under the $\mathrm{LWE}_{\ell,m,q,\chi}$ assumption, where the dimension of the secret is now ℓ instead of n. In particular, we can think of each column of $\mathbf{C}$ as an LWE secrets, the matrix $\mathbf{B}$ as the coefficients, and each column of $\tilde{\mathbf{A}}$ as the corresponding LWE output. Therefore, the following lemma from [20] follows by a simple hybrid argument.

Lemma 3.2 ([20]). *Let* $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$*, and let* $\tilde{\mathbf{A}} \xleftarrow{\$} \mathsf{Lossy}(1^n, 1^m, 1^\ell, q, \chi)$*. Then, under the* $\mathrm{LWE}_{\ell,m,q,\chi}$ *assumption, the following two distributions are computationally indistinguishable:* $\mathbf{A} \overset{\text{comp}}{\approx} \tilde{\mathbf{A}}$.

The following lemma now states that for appropriate parameters, the secret $\mathbf{s}$ maintains high *smooth min-entropy* (see Definition 2.1) given $\tilde{\mathbf{A}}$ and $\lfloor \tilde{\mathbf{A}} \cdot \mathbf{s} \rfloor_p$.

Lemma 3.3. *Let* n, m, ℓ, p, γ *be positive integers,* χ *be some* β*-bounded distribution (i.e.,* $\mathbb{E}[|\chi|] \le \beta$*), and* $q \ge 2\beta\gamma nmp$ *be a prime. Then the following holds:*

(i) (Uniform Secret) For $\tilde{\mathbf{A}} \xleftarrow{\$} \mathsf{Lossy}(1^n, 1^m, 1^\ell, q, \chi)$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ *we have, for* $\varepsilon = 2^{-\lambda} + q^{-\ell}$*:*

$$H_\infty^\varepsilon(\mathbf{s}|\tilde{\mathbf{A}}, \lfloor \tilde{\mathbf{A}}\mathbf{s} \rfloor_p) \ge n\log(2\gamma) - (\ell + \lambda)\log(q).$$

(ii) (High-Entropy Secret) Let $(\mathbf{s}, \mathsf{aux})$ *be correlated random variables with* $\mathbf{s} \in [-\gamma, \gamma]^n \subseteq \mathbb{Z}^n$*, and let* $\tilde{\mathbf{A}} \xleftarrow{\$} \mathsf{Lossy}(1^n, 1^m, 1^\ell, q, \chi)$ *be chosen independently. Then, for* $\varepsilon = 2^{-\lambda} + q^{-\ell}$ *and any* $\varepsilon' > 0$ *we have:*

$$H_\infty^{\varepsilon'+\varepsilon}(\mathbf{s}|\tilde{\mathbf{A}}, \lfloor \tilde{\mathbf{A}}\mathbf{s} \rfloor_p, \mathsf{aux}) \ge H_\infty^{\varepsilon'}(\mathbf{s}|\mathsf{aux}) - (\ell + \lambda)\log(q).$$

Both parts above also holds when q *is* not *prime, as long as the largest prime divisor of* q*, denoted* p_{max}*, satisfies* $GCD(q, q/p_{max}) = 1$*,* $p_{max} \ge 2\beta\gamma nmp$*. In this case we get* $\varepsilon = (2^{-\lambda} + (p_{max})^{-\ell} + \Pr[\mathbf{s} = 0^n \mod p_{max}])$.

The proof is sketched in Section 1.1, and a full proof is given in [1].

4 New "LWR from LWE" Reduction

In the following section we present the main result of this paper, namely sufficient conditions under which the LWR-assumption holds. As discussed earlier: on the positive side, we show that the LWR-assumption also holds if one drops a *small* fraction of the bits in the rounding function. On the negative side, the size of the modulus depends on the number of LWR-samples one needs to output, i.e., on the dimension of the matrix $\mathbf{A}$, and thus this number must be known in advance. However, as we will show in the subsequent sections, this is not a restriction for many interesting applications.

Theorem 4.1. *Let* k, ℓ, n, m, p, γ *be positive integers and* q *be a prime. Further, let* χ *be a* β*-bounded distribution for some* $\beta \in \mathbb{R}$ *(all parameters are functions of* λ*) such that* $q \ge 2\beta\gamma nmp$*. Under the* $\mathrm{LWE}_{\ell,m,q,\chi}$ *assumption we then get:*

(i) If $n \geq (\ell + \lambda + 1)\frac{\log(q)}{\log(2\gamma)} + 2\lambda$*, then the* $\mathrm{LWR}_{n,m,q,p}$*-assumption holds.*

(ii) If $k \geq (\ell + \lambda + 1)\log(q) + 2\lambda$*, then the weak and leaky* $\mathrm{LWR}^{\mathsf{WL}(\gamma,k)}_{n,m,q,p}$*-assumption holds.*

For exact security, if the above LWE assumption is (t, ε)*-secure and* $\ell \geq \lambda$*, then in both cases the corresponding* LWR*-problem is* (t', ε')*-secure, where* $t' = t - \mathsf{poly}(\lambda)$*,* $\varepsilon' = m(2 \cdot n\varepsilon + 3 \cdot 2^{-\lambda}) = \mathsf{poly}(\lambda)(\varepsilon + 2^{-\lambda})$*. Both parts of the above theorem also hold if* q *is not prime as long as the largest prime divisor of* q*, denoted* p_{max}*, satisfies* $GCD(q, q/p_{max}) = 1$*,* $p_{max} \geq 2\beta\gamma nmp$*. In this case we still get* $t' = t - \mathsf{poly}(\lambda)$*,* $\varepsilon' = \mathsf{poly}(\lambda)(\varepsilon + 2^{-\lambda})$*.*

The proof is sketched in Section 1.1, and a full proof can be found in [1].

Remark on β-bounded Distributions. In the theorem, we require that the distribution χ is β-bounded meaning that $\mathbb{E}[|\chi|] \leq \beta$. A different definition, which also would have been sufficient for us, would be to require that $\Pr_{x \xleftarrow{\$} \chi}[|x| > \beta] \leq \mathsf{negl}(\lambda)$. The latter notion of boundedness is used in the work of Banerjee et al. [10]. Although the two notions are technically incomparable (one does not imply the other) for natural distributions, such as the discrete Gaussian, it is easier to satisfy out notion. In particular, the discrete Gaussian distribution Ψ_σ with standard deviation σ satisfies $\mathbb{E}[|\Psi_\sigma|] \leq \sigma$ but we can only get the weaker bound $\Pr_{x \xleftarrow{\$} \Psi_\sigma}[|x| > \sqrt{\omega(\log(\lambda))}\sigma] \leq \mathsf{negl}(\lambda)$. Therefore, we find it advantageous to work with our definition.

Remark on Parameters. Notice that in the above theorem, the parameter γ offers a tradeoff between the size of the modulus q and the secret vector length n: for a bigger γ we need a bigger modulus q but can allow smaller secret length n. The following corollary summarizes two extreme cases of small and large γ.

Corollary 4.2. *Let* Ψ_σ *denote a discrete Gaussian distribution over* $\mathbb{Z}_q$ *with standard deviation* σ*, and assume that the* $\mathrm{LWE}_{\ell,m,q,\Psi_\sigma}$*-assumption holds. Then the* $\mathrm{LWR}_{n,m,q,p}$*-assumption holds in either of the following cases:*

- *(Minimize Modulus/Error Ratio.) If* $q \geq 2\sigma nmp$ *is a prime, and* $n \geq (\ell + \lambda + 1)\log(q) + 2\lambda$*. By setting* $p = O(1)$*, we can get a modulus-to-error ratio as small as* $q/\sigma = O(m \cdot n)$*.*
- *(Maximize Efficiency.) If* $q \geq (2\sigma nm)^3$ *is a prime,* $p = \sqrt[3]{q}$ *and* $n \geq 3\ell + 5\lambda + 3$*. The efficiency of LWR is now similar to the LWE assumption with* $n = O(\ell)$ *and* $\log(p) = O(\log q)$*.*

5 Reusable Extractors

The notion of a 'computational reusable extractor' was defined by Dodis et al. [21]. Intuitively, this is a tool that allows us to take some weak secret $\mathbf{s}$ that has a sufficient amount of entropy, and to use it to repeatedly extract fresh pseudorandomness $\mathsf{Ext}(\mathbf{s}; \mathbf{a}_i)$ using multiple public random seeds $\mathbf{a}_i$. Each extracted

output should look random and independent.[5] The work of [21] constructed such reusable extractors under a new assumption called "Learning Subspaces with Noise (LSN)". Reusable extractors were also implicitly constructed based on the DDH assumption in the work of Naor and Segev [37].[6] Here we give a new construction based on the LWR problem, with a security reduction from the LWE assumption.

Definition 5.1 (Reusable Extractor). *Let $\mathcal{S}, \mathcal{D}, \mathcal{U}$ be some domains, parametrized by the security parameter λ. A function $\mathsf{Ext} : \mathcal{S} \times \mathcal{D} \to \mathcal{U}$ is a (k, m)-reusable-extractor if for any efficiently samplable correlated random variables $\mathbf{s}, \mathsf{aux}$ such that the support of $\mathbf{s}$ is $\mathcal{S}$ and $H_\infty(\mathbf{s}|\mathsf{aux}) \geq k$, we have:*

$$(\mathsf{aux}, \mathbf{a}_1, \ldots, \mathbf{a}_m, \mathsf{Ext}(\mathbf{s}; \mathbf{a}_1), \ldots, \mathsf{Ext}(\mathbf{s}; \mathbf{a}_m)) \overset{\text{comp}}{\approx} (\mathsf{aux}, \mathbf{a}_1, \ldots, \mathbf{a}_m, u_1, \ldots, u_m)$$

where the values $\{\mathbf{a}_j \xleftarrow{\$} \mathcal{D}\}, \{u_j \xleftarrow{\$} \mathcal{U}\}$ are sampled independently.

Theorem 5.2. *Let n, p, γ be integers, p' be a prime, and define $q = p \cdot p'$. Then, assuming that the $\mathrm{LWE}_{\ell,m,q,\chi}$ assumption holds for some β-bounded distribution χ such that $p' > 2\beta\gamma nmp$ and $k \geq (\ell + \lambda + 1)\log(q) + 2\lambda$, the function*

$$\mathsf{Ext} : [-\gamma, \gamma]^n \times \mathbb{Z}_q^n \to \mathbb{Z}_p \quad \textit{defined by} \quad \mathsf{Ext}(\mathbf{s}; \mathbf{a}) \overset{\text{def}}{=} \lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rfloor_p$$

is a (k, m)-reusable extractor.

Notice that one nice property of the above reusable extractor is that it has a *graceful degradation of security* as the min-entropy k of the source drops. In particular, there is no hard threshold on the entropy k determined by the parameters that define the scheme: γ, n, q, p. Instead, as the entropy k drops we can still reduce security from a correspondingly less secure LWE assumption with smaller secret size ℓ. In other words, the scheme designer does not need to know the actual entropy k of the secret - but the scheme gets gradually less/more secure as the entropy of the secret shrinks/grows. A similar notion of graceful security degradation was noted in the work of Goldwasser et al. [20].

6 Lossy Trapdoor Functions

Lossy trapdoor functions (LTDFs) [22,23], are a family of functions $f_{pk}(\cdot)$ keyed by some public key pk, which can be sampled in one of two indistinguishable modes: `injective` and `lossy`. In the `injective` mode the function $f_{pk}(\cdot)$ is injective and we can even sample pk along with a secret trapdoor key sk that allows us to invert it efficiently. In the `lossy` mode, the function $f_{pk}(\cdot)$ is "many-to-one" and $f_{pk}(\mathbf{s})$ statistically loses information about the input $\mathbf{s}$. LTDFs have many interesting applications in cryptography, such as allowing us to output

[5] Equivalently, we can think of a reusable extractor as a weak PRF $f_{\mathbf{s}}(\cdot)$ for which security holds for a bounded number of inputs even using a high entropy key $\mathbf{s}$.

[6] The function $\mathsf{Ext}(\mathbf{s}; \mathbf{a}) = \prod \mathbf{a}_i^{\mathbf{s}_i}$ is a reusable extractor if $\mathbf{s} \in \mathbb{Z}_q^n$, and the $\mathbf{a} \in \mathbb{G}^n$ for some DDH group of prime order q.

many hardcore bits, construct CCA-2 public-key encryption [23, 24], and deterministic encryption [25]. In this section, we construct very simple and efficient LTDFs using the LWR problem, with security based on standard LWE. Our LTDF function is unusually simple: the public key is a matrix $pk = \mathbf{A}$ and the function is defined as $f_{\mathbf{A}}(\mathbf{s}) = \lfloor \mathbf{A} \cdot \mathbf{s} \rceil_p$. As we will describe, one can sample an injective $\mathbf{A}$ with a trapdoor using the techniques of Ajtai [26] or subsequent improvements [27,28], and one can sample a lossy $\mathbf{A}$ using the techniques we developed in Section 3. Although prior constructions of LTDFs from LWE are known [23, 29], our construction here has several advantages. Firstly, our scheme is extremely simple to describe and implement. Secondly, in contrast to both [22, 29], our lossy mode loses "almost all" of the information contained in $\mathbf{s}$. In fact, the amount of "lossiness" in our LTDF construction is flexible and not determined by the parameters of the scheme itself. Even after we fix the parameters that allow us to sample the injective mode, we have an additional free parameter that allows us to make the lossy mode progressively more lossy under under a progressively stronger variant of the LWE assumption.

6.1 Entropic LTDFs

Our notion differs somewhat from that of [23] in how we define the "lossy" property. Instead of requiring that, for a lossy pk, the range of $f_{pk}(\cdot)$ is small, we require that very little *entropy* is lost from observing $f_{pk}(\cdot)$. To the best of our knowledge, our version can be used interchangeably in all of the applications of LTDFs to date. To avoid confusion, we call our notion *entropic* LTDF (eLTDF).

Definition 6.1 (eLTDF). *A family of* $l(\lambda)$-entropic lossy trapdoor functions (eLTDF) *with security parameter* λ *and domain* $\mathcal{D}_\lambda$ *consists of a* PPT *sampling algorithms* Gen *and two deterministic* PPT *algorithms* F, F^{-1} *such that:*

Injective Functions: *For any* (pk, sk) *in the support of* $\mathsf{Gen}(1^\lambda, \mathtt{injective})$*, any* $\mathbf{s} \in \mathcal{D}_\lambda$ *we require that* $F^{-1}(sk, F(pk, \mathbf{s})) = \mathbf{s}$.

Lossy Functions: *When* $pk \xleftarrow{\$} \mathsf{Gen}(1^\lambda, \mathtt{lossy})$*, the function* $F(pk, \cdot)$ *is lossy. In particular, for any mutually correlated random variables* $(\mathbf{s}, \mathsf{aux})$ *where the domain of* $\mathbf{s}$ *is* $\mathcal{D}_\lambda$ *and for an independently sampled* $pk \xleftarrow{\$} \mathsf{Gen}(1^\lambda, \mathtt{lossy})$*, we have:* $H_\infty^{\mathsf{smooth}}(\mathbf{s}|pk, F(pk, \mathbf{s}), \mathsf{aux}) \geq H_\infty^{\mathsf{smooth}}(\mathbf{s}|\mathsf{aux}) - l(\lambda)$*. We call the parameter* $l = l(\lambda)$ *the* residual leakage *of the LTDF.*

Indistinguishability: *The distributions of* pk *as sampled by* $\mathsf{Gen}(1^\lambda, \mathtt{lossy})$ *and* $\mathsf{Gen}(1^\lambda, \mathtt{injective})$ *are computationally indistinguishable.*

We now show how to construct eLTDFs from LWR (and so also from LWE).

Tools. As a tool in our construction, we will rely on the fact that we can sample a random LWE matrix $\mathbf{A}$ along with an *inversion* trapdoor that allows us to recover $\mathbf{s}, \mathbf{e}$ given an LWE sample $\mathbf{As} + \mathbf{e}$ where the error $\mathbf{e}$ is "sufficiently" short. The first example of such algorithms was given by Ajtai in [26], and was subsequently improved in [27]. More recently [28] significantly improved the efficiency of these results, by using a "qualitatively" different type of trapdoor.

We describe the properties that we need abstractly, and can use any of the above algorithms in a black-box manner. In particular we need the following PPT algorithms for some range of parameters (m, n, q, β):

$\mathsf{GenTrap}(1^n, 1^m, q)$**:** An algorithm which on input positive integers n, q and sufficiently large m samples a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ and trapdoor T such that $\mathbf{A}$ is statistically close to uniform (in $n \log q$).

$\mathsf{Invert}(T, \mathbf{A}, \mathbf{c})$**:** An algorithm which receives as input $(\mathbf{A}, T)$ in the support of $\mathsf{GenTrap}(1^n, 1^m, q)$ and some value $\mathbf{c} \in \mathbb{Z}_q^m$ such that $\mathbf{c} = \mathbf{As} + \mathbf{e}$ for some $\mathbf{s} \in \mathbb{Z}_q^n$ and some error satisfying $||\mathbf{e}||_2 \leq \beta$. The algorithm outputs $\mathbf{s}$.

$\mathsf{LWRInvert}(T, \mathbf{A}, \mathbf{c})$ Takes as input $(\mathbf{A}, T)$ in the support of $\mathsf{GenTrap}(1^n, 1^m, q)$ and some value $\mathbf{c} \in \mathbb{Z}_p^m$ such that $\mathbf{c} = \lfloor \mathbf{As} \rfloor_p$ for some $\mathbf{s} \in \mathbb{Z}_q^n$ and outputs $\mathbf{s}$.

For example [28] shows that there are algorithms $(\mathsf{GenTrap}, \mathsf{Invert})$ which work for $n \geq 1$, $q \geq 2$, sufficiently large $m = O(n \log q)$ and sufficiently small $\beta < q/O(\sqrt{n \log q})$. Since we can convert LWR samples $\lfloor \mathbf{As} \rfloor_p$ into samples $\mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ for some short error $||\mathbf{e}||_2 \leq \sqrt{m}q/p$, this also implies the following.

Lemma 6.2 (Trapdoors for LWR). *For $n \geq 1$, $q \geq 2$, sufficiently large $m \geq O(n \log q)$ and $p \geq O(\sqrt{mn \log q})$, there exist* $(\mathsf{GenTrap}, \mathsf{LWRInvert})$ *as above.*

The Construction. We will rely on the algorithms $\mathsf{GenTrap}$ and $\mathsf{LWRInvert}$ described above. We also rely on the lossy sampling algorithm Lossy and its properties developed in Section 3. The construction is parametrized by integers n, m, q, p (all functions of the security parameter λ). Furthermore, there will be two additional parameters ℓ and χ which are only needed by the lossy sampler.

$\mathsf{Gen}(1^\lambda, \mathtt{injective})$**:** Sample $(\mathbf{A}, T) \xleftarrow{\$} \mathsf{GenTrap}(1^n, 1^m, q)$. Output $pk = \mathbf{A}$ and trapdoor $sk = (\mathbf{A}, T)$.

$\mathsf{Gen}(1^\lambda, \mathtt{lossy})$**:** Sample $\mathbf{A} \xleftarrow{\$} \mathsf{Lossy}(1^n, 1^m, 1^\ell, q, \chi)$. Output $pk = \mathbf{A}$.

$F(pk, \mathbf{s})$**:** On input $\mathbf{s} \in \{0,1\}^n$ and matrix $pk = \mathbf{A} \in \mathbb{Z}_q^{m \times n}$ output $\lfloor \mathbf{As} \rfloor_p$.

$F^{-1}(sk, \mathbf{c})$**:** On input $\mathbf{c} \in \mathbb{Z}_p^m$ and $sk = (\mathbf{A}, T)$ output $\mathsf{LWRInvert}(T, \mathbf{A}, \mathbf{c})$.

The following theorem summarizes the properties of this construction.

Theorem 6.3. *Let χ be an efficiently samplable β-bounded distribution and λ be the security parameter. For any positive integers $n \geq \lambda$, sufficiently large $m \geq O(n \log q)$, $p \geq O(\sqrt{mn \log q})$ and a prime $q \geq 2\beta nmp$, if the* $\mathrm{LWE}_{\ell,m,q,\chi}$ *assumption holds then the above construction is an l-LTDF with $l = (\ell + \lambda) \log q$.*

We refer the interested reader to the full version [1], where we additionally show how to construct efficient *all-but-one trapdoor functions*, and how to obtain CCA-2 secure encryption schemes therefrom.

7 Deterministic Encryption

Deterministic public-key encryption [25,30–33] is intended to guarantee security as long as the messages have sufficient entropy. Although there are black-box

constructions of deterministic encryption using LTDFs [32], here we present a very simple direct construction from the LWR problem. There are several definitions of deterministic encryption which can be proven equivalent; see [31, 32]. Here, we will use one such simple definition based on indistinguishability of encrypting messages from two different distributions.

Definition 7.1 (Deterministic Encryption). *A triple of* PPT *algorithms* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$*, where* $\mathsf{Enc}, \mathsf{Dec}$ *are deterministic, is a* deterministic encryption scheme *with message length* $n = n(\lambda)$*, if it satisfies the following properties. First, it is* correct*, i.e., for all* $(pk, sk) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$ *and all messages* $\mathbf{s} \in \{0,1\}^n$*, we have* $\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(\mathbf{s})) = \mathbf{s}$*. We further say that the scheme is* secure *for all* $k(\lambda)$*-sources if for any two distribution ensembles* $\{S_\lambda^{(0)}\}_{\lambda\in\mathbb{N}}, \{S_\lambda^{(1)}\}_{\lambda\in\mathbb{N}}$ *over* $\{0,1\}^{n(\lambda)}$ *which are efficiently samplable in* $\mathsf{poly}(\lambda)$*-times and have sufficient entropy* $H_\infty(S_\lambda^0) \geq k$*,* $H_\infty(S_\lambda^1) \geq k$*, we have* $(pk, \mathsf{Enc}_{pk}(\mathbf{s}_0)) \overset{\text{comp}}{\approx} (pk, \mathsf{Enc}_{pk}(\mathbf{s}_1))$*, where* $\mathbf{s}_0 \xleftarrow{\$} S_\lambda^{(0)}$ *and* $\mathbf{s}_1 \xleftarrow{\$} S_\lambda^{(1)}$ *and* $(pk, sk) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$*.*

Construction. We give a very simple construction of deterministic encryption based on the LWR assumption. This construction is the same as one given by Xie et al. [18], except for the setting of parameters. Whereas they required a super-polynomial modulus and modulus to error ratio by relying on variants of the analysis of [10,20] we use our improved analysis from Section 4. We will rely on the LWR trapdoor generation and inversion algorithms $\mathsf{GenTrap}, \mathsf{LWRInvert}$ described in Section 6.1 and Lemma 6.2. Our scheme is parametrized by some n, m, q, p, all functions of the security parameter λ, and has message length n.

$\mathsf{Gen}(1^\lambda)$**:** Choose $(\mathbf{A}, T) \xleftarrow{\$} \mathsf{GenTrap}(1^n, 1^m, q)$. Output $pk = \mathbf{A}$, $sk = T$.
$\mathsf{Enc}_{pk}(\mathbf{s})$**:** For a message $\mathbf{s} \in \{0,1\}^n$, output $\lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p$.
$\mathsf{Dec}_{sk}(\mathbf{c})$**:** For a ciphertext $\mathbf{c} \in \mathbb{Z}_p^m$, output $\mathsf{LWRInvert}(T, \mathbf{A}, \mathbf{c})$.

Theorem 7.2. *Let* λ *be the security parameter,* $n \geq \lambda, \ell, m, p$ *be an integers,* q *be a prime, and* χ *be an efficiently samplable* β*-bounded distribution (all parameters are functions of* λ*) such that* $m \geq O(n \log q)$*,* $p \geq O(\sqrt{mn \log q})$ *are sufficiently large and* $q \geq 2\beta nmp$*. If the* $\mathrm{LWE}_{\ell,m,q,\chi}$ *assumption holds then the above construction with parameters* n, m, q, p *is a deterministic encryptions secure for all* k *sources where* $k \geq (\ell + \Omega(\lambda)) \log(q)$*.*

One big advantage of our scheme is that the parameters n, m, q, p do not determine the minimal entropy k. Instead for any k, we can prove security under a corresponding LWE assumption with dimension $\ell < k$.

8 Open Problems

We conclude with two interesting open problems. Firstly, is it possible to improve the reduction and remove the dependence between the modulus q and the number of samples m? And secondly, is there a related reduction for *Ring LWR* from *Ring LWE*? This does not seem to follow in a straight-forward manner.

References

1. Alwen, J., Krenn, S., Pietrzak, K., Wichs, D.: Learning with Rounding, Revisited: New Reduction, Properties and Applications. Cryptology ePrint Archive, Report 2013/098 (2013)
2. Regev, O.: On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. In: Gabow, H.N., Fagin, R. (eds.) STOC, pp. 84–93. ACM (2005)
3. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for Hard Lattices and New Cryptographic Constructions. In: Dwork, C. (ed.) STOC, pp. 197–206. ACM (2008)
4. Akavia, A., Goldwasser, S., Vaikuntanathan, V.: Simultaneous Hardcore Bits and Cryptography against Memory Attacks. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 474–495. Springer, Heidelberg (2009)
5. Lyubashevsky, V., Peikert, C., Regev, O.: On Ideal Lattices and Learning with Errors over Rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010)
6. Agrawal, S., Freeman, D.M., Vaikuntanathan, V.: Functional Encryption for Inner Product Predicates from Learning with Errors. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 21–40. Springer, Heidelberg (2011)
7. Brakerski, Z., Vaikuntanathan, V.: Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 505–524. Springer, Heidelberg (2011)
8. Lindner, R., Peikert, C.: Better Key Sizes (and Attacks) for LWE-Based Encryption. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 319–339. Springer, Heidelberg (2011)
9. Goldwasser, S., Kalai, Y., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: Succinct Functional Encryption and Applications: Reusable Garbled Circuits and Beyond. Cryptology ePrint Archive, Report 2012/733 (2012)
10. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom Functions and Lattices. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 719–737. Springer, Heidelberg (2012)
11. Lyubashevsky, V.: Fiat-Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 598–616. Springer, Heidelberg (2009)
12. Rückert, M.: Lattice-Based Blind Signatures. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 413–430. Springer, Heidelberg (2010)
13. Lyubashevsky, V.: Lattice Signatures without Trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Heidelberg (2012)
14. Katz, J., Vaikuntanathan, V.: Smooth Projective Hashing and Password-Based Authenticated Key Exchange from Lattices. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 636–652. Springer, Heidelberg (2009)
15. Peikert, C., Rosen, A.: Efficient Collision-Resistant Hashing from Worst-Case Assumptions on Cyclic Lattices. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 145–166. Springer, Heidelberg (2006)
16. Peikert, C., Vaikuntanathan, V., Waters, B.: A Framework for Efficient and Composable Oblivious Transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (2008)
17. Peikert, C.: Public-Key Cryptosystems from the Worst-Case Shortest Vector Problem: Extended Abstract. In: Mitzenmacher, M. (ed.) STOC, pp. 333–342. ACM (2009)
18. Xie, X., Xue, R., Zhang, R.: Deterministic Public Key Encryption and Identity-Based Encryption from Lattices in the Auxiliary-Input Setting. In: Visconti, I., De Prisco, R. (eds.) SCN 2012. LNCS, vol. 7485, pp. 1–18. Springer, Heidelberg (2012)

19. Brakersi, Z., Langlois, A., Peikert, C., Regev, O., Stehl, D.: Classical Hardness of Learning with Errors. In: STOC (2013)
20. Goldwasser, S., Kalai, Y.T., Peikert, C., Vaikuntanathan, V.: Robustness of the Learning with Errors Assumption. In: Yao, A.C.C. (ed.) ICS, pp. 230–240. Tsinghua University Press (2010)
21. Dodis, Y., Kalai, Y.T., Lovett, S.: On Cryptography with Auxiliary Input. In: Mitzenmacher, M. (ed.) STOC, pp. 621–630. ACM (2009)
22. Peikert, C., Waters, B.: Lossy Trapdoor Functions and Their Applications. In: Dwork, C. (ed.) STOC, pp. 187–196. ACM (2008)
23. Peikert, C., Waters, B.: Lossy Trapdoor Functions and Their Applications. SIAM J. Comput. 40(6), 1803–1844 (2011)
24. Mol, P., Yilek, S.: Chosen-Ciphertext Security from Slightly Lossy Trapdoor Functions. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 296–311. Springer, Heidelberg (2010)
25. Fuller, B., O'Neill, A., Reyzin, L.: A Unified Approach to Deterministic Encryption: New Constructions and a Connection to Computational Entropy. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 582–599. Springer, Heidelberg (2012)
26. Ajtai, M.: Generating Hard Instances of the Short Basis Problem. In: Wiedermann, J., Van Emde Boas, P., Nielsen, M. (eds.) ICALP 1999. LNCS, vol. 1644, pp. 1–9. Springer, Heidelberg (1999)
27. Alwen, J., Peikert, C.: Generating Shorter Bases for Hard Random Lattices. Theory Comput. Syst. 48(3), 535–553 (2011)
28. Micciancio, D., Peikert, C.: Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012)
29. Bellare, M., Kiltz, E., Peikert, C., Waters, B.: Identity-Based (Lossy) Trapdoor Functions and Applications. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 228–245. Springer, Heidelberg (2012)
30. Bellare, M., Boldyreva, A., O'Neill, A.: Deterministic and Efficiently Searchable Encryption. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 535–552. Springer, Heidelberg (2007)
31. Bellare, M., Fischlin, M., O'Neill, A., Ristenpart, T.: Deterministic Encryption: Definitional Equivalences and Constructions without Random Oracles. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 360–378. Springer, Heidelberg (2008)
32. Boldyreva, A., Fehr, S., O'Neill, A.: On Notions of Security for Deterministic Encryption, and Efficient Constructions without Random Oracles. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 335–359. Springer, Heidelberg (2008)
33. Brakerski, Z., Segev, G.: Better Security for Deterministic Public-Key Encryption: The Auxiliary-Input Setting. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 543–560. Springer, Heidelberg (2011)
34. Döttling, N., Müller-Quade, J.: Lossy Codes and a New Variant of the Learning-With-Errors Problem. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 18–34. Springer, Heidelberg (2013)
35. Micciancio, D., Peikert, C.: Hardness of SIS and LWE with Small Parameters. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 21–39. Springer, Heidelberg (2013)
36. Renner, R., Wolf, S.: Smooth Rényi Entropy and Applications. In: ISIT, vol. 4, p. 233 (2004)
37. Naor, M., Segev, G.: Public-Key Cryptosystems Resilient to Key Leakage. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 18–35. Springer, Heidelberg (2009)

Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based

Craig Gentry[1,⋆], Amit Sahai[2,⋆⋆], and Brent Waters[3,⋆⋆⋆]

[1] IBM Research
cbgentry@us.ibm.com
[2] UCLA
sahai@cs.ucla.edu
[3] UT Austin
bwaters@cs.utexas.edu

Abstract. We describe a comparatively simple fully homomorphic encryption (FHE) scheme based on the learning with errors (LWE) problem. In previous LWE-based FHE schemes, multiplication is a complicated and expensive step involving "relinearization". In this work, we propose a new technique for building FHE schemes that we call the *approximate eigenvector* method. In our scheme, for the most part, homomorphic addition and multiplication are just matrix addition and multiplication. This makes our scheme both asymptotically faster and (we believe) easier to understand.

In previous schemes, the homomorphic evaluator needs to obtain the user's "evaluation key", which consists of a chain of encrypted secret

⋆ This work was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center (DoI/NBC) contract number D11PC20202. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

⋆⋆ Research supported in part from a DARPA/ONR PROCEED award, NSF grants 1228984, 1136174, 1118096, 1065276, 0916574 and 0830803, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, and an Okawa Foundation Research Grant. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0389. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.

⋆⋆⋆ Supported by NSF CNS-0915361 and CNS-0952692, CNS-1228599 DARPA via Office of Naval Research under Contract N00014-11-1-0382, DARPA N11AP20006, the Alfred P. Sloan Fellowship, and Microsoft Faculty Fellowship, and Packard Foundation Fellowship. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Department of Defense or the U.S. Government.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 75–92, 2013.

keys. Our scheme has no evaluation key. The evaluator can do homomorphic operations without knowing the user's public key at all, except for some basic parameters. This fact helps us construct the first identity-based FHE scheme. Using similar techniques, we show how to compile a recent attribute-based encryption scheme for circuits by Gorbunov et al. into an attribute-based FHE scheme that permits data encrypted under the same index to be processed homomorphically.

1 Introduction

Fully homomorphic encryption (FHE) schemes [RAD78, Gen09, Gen10, vDGHV10] [SV10, GH11b, CMNT11, BV11a, BV11b, GH11a, BGV12, CNT12, GHS12a, GHS12b] [LATV12, Bra12] "have been simplified enough so that their description can fit, well, in a blog post" [BB12b, BB12a]. In this paper, we try to make FHE even simpler.

1.1 Previous FHE Schemes Based on Learning with Errors

Currently, perhaps the simplest leveled[1] FHE scheme based on the learning with errors (LWE) assumption [Reg05] is by Brakerski [Bra12]. In fact, Barak and Brakerski do give a remarkably clear exposition of this scheme in a blog post [BB12a]. However, while the scheme's key generation, encryption, decryption, and homomorphic addition procedures are easy to describe, they note that "multiplication is more tricky".

In Brakerski's scheme, similar to previous FHE schemes based on LWE [BV11b, BGV12], the ciphertext $\boldsymbol{c}$ and secret key $\boldsymbol{s}$ are n-dimensional vectors whose dot product $\langle \boldsymbol{c}, \boldsymbol{s} \rangle \approx \mu$ equals the message μ, up to some small "error" that is removed by rounding. Homomorphic multiplication uses an identity regarding dot products of tensor products of vectors: namely, $\langle \boldsymbol{u}_1 \otimes \boldsymbol{u}_2, \boldsymbol{v}_1 \otimes \boldsymbol{v}_2 \rangle = \langle \boldsymbol{u}_1, \boldsymbol{v}_1 \rangle \cdot \langle \boldsymbol{u}_2, \boldsymbol{v}_2 \rangle$. Thus, if ciphertexts $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ satisfy $\langle \boldsymbol{c}_1, \boldsymbol{s} \rangle \approx \mu_1$ and $\langle \boldsymbol{c}_2, \boldsymbol{s} \rangle \approx \mu_2$, then $\langle \boldsymbol{c}_1 \otimes \boldsymbol{c}_2, \boldsymbol{s} \otimes \boldsymbol{s} \rangle \approx \mu_1 \cdot \mu_2$, where $\boldsymbol{c}_1 \otimes \boldsymbol{c}_2$ is interpreted as the new ciphertext and $\boldsymbol{s} \otimes \boldsymbol{s}$ as the new secret key, each having dimension $\Theta(n^2)$. Since multiplying-by-tensoring blows up the ciphertext size, it can only be used for a constant number of steps. For efficiency, the evaluator must *relinearize* [BV11b] the ciphertext after tensoring. Relinearization is a procedure that takes the *long* ciphertext that encrypts $\mu_1 \cdot \mu_2$ under the long key $\boldsymbol{s} \otimes \boldsymbol{s}$, and compresses it into a *normal-sized* n-dimensional ciphertext that encrypts $\mu_1 \cdot \mu_2$ under a normal-sized n-dimensional key $\boldsymbol{s}'$. To relinearize, the evaluator multiplies the long ciphertext vector by a special $n \times \Theta(n^2)$ relinearization matrix.

[1] "Leveled" FHE is a relaxation of "pure" FHE [Gen09]. For fixed parameters, a pure FHE scheme can evaluate arbitrary circuits. In a leveled FHE scheme, the parameters of the scheme may depend on the *depth*, but not the *size*, of the circuits that the scheme can evaluate. We focus on leveled FHE schemes, and typically omit the term "leveled". One can transform our leveled FHE schemes to pure ones by using Gentry's bootstrapping theorem and assuming "circular security" [Gen09].

This relinearization matrix is part of the "evaluation key" that the evaluator must obtain from the public key to perform homomorphic evaluation.

The relinearization step [BV11b] is ingenious and is perhaps the main insight that led to FHE based on LWE. However, relinearization is not particularly natural, nor is it easy to give an intuitive description of how and why it works. Moreover, relinearization is expensive. Each relinearization matrix has size $\Omega(n^3)$, and the public key must contain L of them to evaluate circuits of maximum multiplicative depth L. Computationally, relinearization requires $\Omega(n^3)$ operations, where each operation has cost polynomial in L.

This situation raises the question: Can we construct a LWE-based FHE scheme with a *natural* multiplication procedure? For ciphertexts c_1 and c_2, can we construct a scheme where homomorphic addition and multiplication are just $c_1 + c_2$ and $c_1 \cdot c_2$, where '$+$' and '$\cdot$' are natural algebraic operations over some ring, and where the new ciphertexts have the "same form" as the old ones; for example, $c_1 \cdot c_2$ is not a "long" ciphertext? Can we eliminate the need for an "evaluation key" in general, and the relinearization matrices in particular? If so, LWE-based FHE might become easier to explain. If we can simplify LWE-based FHE while also improving its efficiency and supporting new applications, then even better.

1.2 Our Results

Our main results are:

- **Conceptually simpler FHE based on LWE:** We fully describe our scheme here in the Introduction, and think our new approach will prove valuable pedagogically and theoretically.
- **Asymptotically faster FHE based on LWE:** We eliminate relinearization and the large relinearization matrices, with their $\Omega(n^3)$ complexity. Instead, ciphertexts are matrices that are added and multiplied naturally. In principle, matrix multiplication uses sub-cubic computation: e.g., Strassen and Williams achieved $n^{2.807}$ and $n^{2.3727}$ respectively [Str69, Wil12].
- **Identity-based FHE:** We solve an open problem mentioned in previous works [Nac10, GHV10, Bra12, CHT13] – namely, to construct an identity-based FHE scheme, in which there are no user-specific keys that must be obtained by the encrypter or evaluator. Informally speaking, in an identity-based FHE scheme, a user that has only the public parameters should be able to perform *both* encryption and homomorphism operations. The homomorphism operations should allow a user to take two ciphertexts encrypted to the same target identity, and homomorphically combine them to produce another ciphertext under the same target identity. Previously, only "weak" identity-based FHE schemes were known, where the evaluator needs a user-specific evaluation key, and thus the homomorphism is *not* exploitable by a user that only has the public parameters. Our scheme solves the problem by eliminating evaluation keys entirely.

  We obtain our identity-based FHE scheme by presenting a "compiler" that transforms any LWE-based IBE scheme in the literature that satisfies certain properties, into a fully homomorphic identity-based encryption

scheme. Several LWE-based IBE schemes in the literature satisfy the properties needed for our compiler [GPV08, ABB10a, ABB10b, CHKP10].

– **Attribute-based FHE:** Recently Gorbunov et al. [GVW13] constructed an attribute-based encryption (ABE) for circuits based on LWE. Our compiler for LWE-based IBE also works for their ABE scheme, with relatively minor modifications. We obtain an ABE scheme in which messages encrypted under the same index can be processed homomorphically without any evaluation key in a polynomial depth circuit, and still be decrypted by any party that was entitled to decrypt the original ciphertexts.[2]

Our FHE scheme retains advantages of other LWE-based FHE schemes, such as making bootstrapping optional [BGV12], (with bootstrapping) basing security on LWE for quasi-polynomial factors versus sub-exponential factors [BGV12], eliminating "modulus switching" [Bra12], and basing security directly on the hardness of classical GapSVP [Bra12].

We do not want to oversell our asymptotic result; we now provide some additional context: In general, FHE schemes based on LWE have much worse performance (certainly asymptotically) than schemes based on ring LWE (RLWE) [LPR10, BV11a, GHS12a], and even RLWE-based schemes cannot yet be considered practical [GHS12b]. Moreover, sub-cubic matrix multiplication algorithms may not beat cubic ones by much in practice. Rather, we view our asymptotic result mainly as evidence of how fundamentally new our techniques are. We note that it is straightforward to construct an RLWE-based version of our scheme, but its performance is worse than the best known RLWE-based schemes [BGV12, Bra12, GHS12a, GHS12b] by log factors. On the other hand, while our techniques may not reduce evaluation complexity as much as we would like, they reduce the space complexity significantly (from quasi-cubic to quasi-quadratic), which is a significant issue for LWE-based FHE schemes in practice.

As with all current FHE schemes without bootstrapping, the parameters and per-gate complexity of evaluation depend on the multiplicative depth L of the circuit. "Bootstrapping" [Gen09], together with an assumption of circular security, remains the only known way of making these performance metrics independent of L, and while the overhead of bootstrapping is high, it becomes an attractive option once L passes some threshold. However, our scheme loses some of its advantages once bootstrapping is used. First, to apply bootstrapping, the evaluator needs to obtain the user's secret key encrypted under its public key – in effect, an evaluation key – and therefore we no longer achieve identity-based/attribute-based FHE in this context. Second, this encrypted secret key has quasi-cubic size in our scheme, and while this can be mitigated by public key compression techniques [CNT12], it eliminates the space complexity advantages of our scheme. Essentially, bootstrapping returns us to the realm of "unnatural" operations, with all of its disadvantages. It remains a fascinating open problem to find some

[2] Independently, Garg et al. [GGH+13b] also recently constructed an ABE scheme for circuits using multilinear maps [GGH13a, CLT13], but our techniques do not work as effectively with their scheme.

"natural" alternative to bootstrapping, and (relatedly) to achieve "pure" FHE without an assumption of circular security.

1.3 An Overview of Our FHE Scheme

Our main insight is that we can achieve LWE-based homomorphic encryption where homomorphic addition and multiplication correspond directly to matrix addition and multiplication.

Homomorphic Operations. Let us skip key generation and encryption for the moment, and jump directly to the homomorphic operations (and decryption).

In our scheme, for some modulus q and dimension parameter N to be specified later, a ciphertext C is a $N \times N$ matrix over $\mathbb{Z}_q$, with "small" entries (much smaller than q) and the secret key $\boldsymbol{v}$ is a N-dimensional vector over $\mathbb{Z}_q$ with at least one "big" coefficient v_i. We restrict the message μ to be a "small" integer. We say C encrypts μ when $C \cdot \boldsymbol{v} = \mu \cdot \boldsymbol{v} + \boldsymbol{e}$, where $\boldsymbol{e}$ is a "small" error vector. To decrypt, we extract the i-th row C_i from C, compute $x \leftarrow \langle C_i, \boldsymbol{v} \rangle = \mu \cdot v_i + e_i$, and output $\mu = \lfloor x/v_i \rceil$. In a nutshell, the essence of our scheme is that the secret key $\boldsymbol{v}$ is an *approximate eigenvector* of the ciphertext matrix C, and the message μ is the *eigenvalue*.

Now, let us see why matrix addition and multiplication are correct homomorphic operations. Suppose C_1 and C_2 encrypt μ_1 and μ_2 in that $C_i \cdot \boldsymbol{v} = \mu_i \cdot \boldsymbol{v} + \boldsymbol{e}_i$ for small $\boldsymbol{e}_i$. Let $C^+ = C_1 + C_2$ and $C^\times = C_1 \cdot C_2$. For addition, we have $C^+ \cdot \boldsymbol{v} = (\mu_1 + \mu_2) \cdot \boldsymbol{v} + (\boldsymbol{e}_1 + \boldsymbol{e}_2)$, where the error likely has grown a little, as usual in FHE schemes. But assuming the error is still "small", the sum of the ciphertext matrices encrypts the sum of the messages. For multiplication, we have

$$C^\times \cdot \boldsymbol{v} = C_1 \cdot (\mu_2 \cdot \boldsymbol{v} + \boldsymbol{e}_2) = \mu_2 \cdot (\mu_1 \cdot \boldsymbol{v} + \boldsymbol{e}_1) + C_1 \cdot \boldsymbol{e}_2 = \mu_1 \cdot \mu_2 \cdot \boldsymbol{v} + \mu_2 \cdot \boldsymbol{e}_1 + C_1 \cdot \boldsymbol{e}_2$$
$$= \mu_1 \cdot \mu_2 \cdot \boldsymbol{v} + \mathit{small}$$

where the final error vector is hopefully "small", since μ_2, C_1, $\boldsymbol{e}_1$, and $\boldsymbol{e}_2$ are all small. If so, the product of the ciphertext matrices encrypts the product of the messages. Interestingly, $C_2 \cdot C_1$ is also an encryption of $\mu_1 \cdot \mu_2$, even though matrix multiplication is not commutative.

To simplify further, it might be helpful to imagine an *error-free* version of the scheme, where $C_i \cdot \boldsymbol{v} = \mu_i \cdot \boldsymbol{v}$ *exactly*. In this case, the key $\boldsymbol{v}$ is an (exact) eigenvector of ciphertext matrices, and the message μ_i is the eigenvalue. In general, if matrices C_1 and C_2 have a common eigenvector $\boldsymbol{v}$ with eigenvalues μ_1 and μ_2, then $C_1 \cdot C_2$ and $C_2 \cdot C_1$ have eigenvector $\boldsymbol{v}$ with eigenvalue $\mu_1 \cdot \mu_2$.

Of course, in our scheme, the secret key $\boldsymbol{v}$ is only an *approximate* eigenvector, not an *exact* one. Introducing error is necessary to base the security of our scheme on LWE. The cost of making $\boldsymbol{v}$ only an *approximate* eigenvector is that certain terms in our scheme must be "small" to ensure that homomorphic operations do not disrupt the essential form of the ciphertexts. We call our new approach to LWE-based (homomorphic) encryption the *approximate eigenvector* method.

Bounding the Error and Somewhat Homomorphic Encryption. Although we have not fully specified the scheme, let us go ahead and estimate how homomorphic it is. The scheme above works correctly until the coefficients of the error vector begin to approach q in magnitude. How many homomorphic operations can we perform before that happens?

Suppose C_1 and C_2 are B-bounded ciphertexts, in the sense that μ_i and the coefficients of C_i and $\boldsymbol{e}_i$ all have magnitude at most some bound B. Then, C^+ is $2B$-bounded, and $C^\times$ is $(N+1)B^2$-bounded. In short, the error level grows worse than B^{2^L}, *doubly exponentially with the multiplicative depth* L of the circuit being evaluated. Alternatively, if one wants to consider the *degree* (rather than *depth*) of functions that can be evaluated, if we evaluate a multivariate polynomial $P(x_1, \ldots, x_t)$ of total degree d, on B-bounded ciphertexts as input, the final ciphertext is $|P|(N+1)^{d-1}B^d$-bounded, where $|P|$ is the ℓ_1-norm of P's coefficient vector. Taking q to comfortably exceed this bound, we (roughly) can evaluate polynomials of degree $\log_{NB} q$. Since q/B must be subexponential (at most) in N for security reasons, our scheme-so-far can only evaluate polynomials of (sublinear) polynomial degree in N (only logarithmic depth). In short, our scheme-so-far is a *somewhat homomorphic encryption* (SWHE) scheme [Gen09] that can evaluate log-depth or polynomial degree. Though not yet fully homomorphic, it is by far the most homomorphic LWE-based encryption scheme that uses only "natural" homomorphic operations.

Flattening Ciphertexts and Fully Homomorphic Encryption. To obtain a leveled FHE scheme that can evaluate circuits of polynomial *depth* without bootstrapping or techniques like relinearization, we need to ensure better bounds on the growth of the error. Let us say that a ciphertext C is B-strongly-bounded if its associated μ and the coefficients of C all have magnitude *at most 1*, while the coefficients of its $\boldsymbol{e}$ all have magnitude at most B. If we evaluate a NAND gate on B-strongly-bounded ciphertexts C_1, C_2 to obtain a new ciphertext $C_3 \leftarrow I_N - C_1 \cdot C_2$ (where I_N is the N-dimensional identity matrix), then the message remains in $\{0,1\}$, and the coefficients of C_3's error vector have magnitude at most $(N+1)B$. If we could somehow additionally ensure that C_3's coefficients have magnitude at most 1 so that strong-boundedness is preserved, then we could evaluate a circuit of *depth* L while keeping the error magnitude at most $(N+1)^L B$. Setting q/B to be subexponential in N, we could evaluate a circuit of polynomial *depth* rather than merely polynomial *degree*. In short, we would have a leveled FHE scheme.

Here we describe a operation called ciphertext *flattening* that keeps ciphertexts strongly bounded, so that we obtain leveled FHE.

Flattening uses some simple transformations from [BV11b, BGV12, Bra12] that modify vectors without affecting dot products. Let $\boldsymbol{a}$, $\boldsymbol{b}$ be vectors of some dimension k over $\mathbb{Z}_q$. Let $\ell = \lfloor \log_2 q \rfloor + 1$ and $N = k \cdot \ell$. Let $\mathsf{BitDecomp}(\boldsymbol{a})$ be the N-dimensional vector $(a_{1,0}, \ldots, a_{1,\ell-1}, \ldots, a_{k,0}, \ldots, a_{k,\ell-1})$, where $a_{i,j}$ is the j-th bit in a_i's binary representation, bits ordered least significant to most significant. For $\boldsymbol{a'} = (a_{1,0}, \ldots, a_{1,\ell-1}, \ldots, a_{k,0}, \ldots, a_{k,\ell-1})$, let $\mathsf{BitDecomp}^{-1}(\boldsymbol{a'}) = (\sum 2^j \cdot a_{1,j}, \ldots, \sum 2^j \cdot a_{k,j})$ be the inverse of $\mathsf{BitDecomp}$, but well-defined even

when the input is not a 0/1 vector. For N-dimensional $\boldsymbol{a}'$, let $\mathsf{Flatten}(\boldsymbol{a}') = \mathsf{BitDecomp}(\mathsf{BitDecomp}^{-1}(\boldsymbol{a}'))$, a N-dimensional vector with 0/1 coefficients. When A is a matrix, let $\mathsf{BitDecomp}(A)$, $\mathsf{BitDecomp}^{-1}$, or $\mathsf{Flatten}(A)$ be the matrix formed by applying the operation to each row of A separately. Finally, let $\mathsf{Powersof2}(\boldsymbol{b}) = (b_1, 2b_1, \ldots, 2^{\ell-1}b_1, \ldots, b_k, 2b_k, \ldots, 2^{\ell-1}b_k)$, a N-dimensional vector. Here are some obvious facts:

- $\langle \mathsf{BitDecomp}(\boldsymbol{a}), \mathsf{Powersof2}(\boldsymbol{b}) \rangle = \langle \boldsymbol{a}, \boldsymbol{b} \rangle$.
- For any N-dimensional $\boldsymbol{a}'$, $\langle \boldsymbol{a}', \mathsf{Powersof2}(\boldsymbol{b}) \rangle = \langle \mathsf{BitDecomp}^{-1}(\boldsymbol{a}'), \boldsymbol{b} \rangle = \langle \mathsf{Flatten}(\boldsymbol{a}'), \mathsf{Powersof2}(\boldsymbol{b}) \rangle$.

An interesting feature of $\mathsf{Flatten}$ is that it makes the coefficients of a vector or matrix *small*, without affecting its product with $\mathsf{Powersof2}(\boldsymbol{b})$, and without knowing $\boldsymbol{b}$.

To facilitate ciphertext flattening, we give a special form to our secret key $\boldsymbol{v}$. Specifically, we set $\boldsymbol{v} = \mathsf{Powersof2}(\boldsymbol{s})$ for some secret vector $\boldsymbol{s}$ (to be specified later). This form is consistent with our earlier requirement that $\boldsymbol{v}$ have some big coefficient v_i for decryption; indeed, since $\boldsymbol{v}$'s coefficients go up by $\lfloor \log_2 q \rfloor$ powers of 2, it *must* have a big coefficient suitable to recover $\mu \in \{0, 1\}$.

Now, for any $N \times N$ matrix C, we have $\mathsf{Flatten}(C) \cdot \boldsymbol{v} = C \cdot \boldsymbol{v}$. So, after we compute an initial ciphertext $C_3 \leftarrow I_N - C_1 \cdot C_2$ for the NAND gate, we set $C^{\mathsf{NAND}} = \mathsf{Flatten}(C_3)$ to obtain a ciphertext that has 0/1 coefficients and is strongly bounded. Thus, we obtain leveled FHE without relinearization, under a fixed approximate eigenvector secret key.

Key Generation, Encryption, and Reduction to LWE. Let us finally circle back to key generation and encryption. We want to base security on LWE. So, for key generation, we generate an LWE instance. For suitable parameters $q, n, m = O(n \log q)$, an LWE instance over $\mathbb{Z}_q$ consists of a $m \times (n+1)$ matrix A such that there exists a $(n+1)$-dimensional vector $\boldsymbol{s}$ whose first coefficient is 1 where $\boldsymbol{e} = A \cdot \boldsymbol{s}$ is a "small" error vector. (See Section 2 for a formal definition of LWE.) In our scheme, A is public and $\boldsymbol{s}$ is secret. We set our approximate eigenvector to be $\boldsymbol{v} = \mathsf{Powersof2}(\boldsymbol{s})$, a vector of dimension $N = (n+1) \cdot \ell$ for $\ell = \lfloor \log_2 q \rfloor + 1$.

To encrypt $\mu \in \mathbb{Z}_q$, the encrypter generates a random $N \times m$ matrix R with 0/1 entries, and sets $C = \mathsf{Flatten}(\mu \cdot I_N + \mathsf{BitDecomp}(R \cdot A))$, where I_N is the N-dimensional identity matrix. Since $\mathsf{Flatten}$ does not affect the product with $\boldsymbol{v}$, we have:

$$C \cdot \boldsymbol{v} = \mu \cdot \boldsymbol{v} + \mathsf{BitDecomp}(R \cdot A) \cdot \boldsymbol{v} = \mu \cdot \boldsymbol{v} + R \cdot A \cdot \boldsymbol{s} = \mu \cdot \boldsymbol{v} + \mathit{small}$$

$\mathsf{Flatten}$ ensures that the coefficients of C are small, and therefore that C has the proper form of a ciphertext that permits our homomorphic operations. Decryption works as mentioned previously.

To show that security is based on LWE, it is now enough to show that C is statistically independent of μ when A is a uniformly random $m \times (n+1)$ matrix over $\mathbb{Z}_q$. Let $C' = \mathsf{BitDecomp}^{-1}(C)$. Recall that C is $\mathsf{Flatten}$'d, and so

$C = \mathsf{Flatten}(C) = \mathsf{BitDecomp}(C')$. Therefore, C reveals nothing more than C'. But $C' = \mathsf{BitDecomp}^{-1}(\mu \cdot I_N) + R \cdot A$, and $R \cdot A$ is statistically uniform by the leftover hash lemma when $m = O(n \log q)$ is chosen appropriately.

1.4 Roadmap

After finishing some preliminaries in Section 2, we describe our new FHE construction more formally in Section 3. In Section 4, we provide an overview of our identity-based and attribute-based FHE schemes.

2 Preliminaries

2.1 The Learning With Errors (LWE) Problem and GapSVP

The learning with errors (LWE) problem was introduced by Regev [Reg05].

Definition 1 (LWE). *For security parameter λ, let $n = n(\lambda)$ be an integer dimension, let $q = q(\lambda) \geq 2$ be an integer, and let $\chi = \chi(\lambda)$ be a distribution over $\mathbb{Z}$. The $\mathrm{LWE}_{n,q,\chi}$ problem is to distinguish the following two distributions: In the first distribution, one samples $(\boldsymbol{a}_i, b_i)$ uniformly from $\mathbb{Z}_q^{n+1}$. In the second distribution, one first draws $\boldsymbol{s} \leftarrow \mathbb{Z}_q^n$ uniformly and then samples $(\boldsymbol{a}_i, b_i) \in \mathbb{Z}_q^{n+1}$ by sampling $\boldsymbol{a}_i \leftarrow \mathbb{Z}_q^n$ uniformly, $e_i \leftarrow \chi$, and setting $b_i = \langle \boldsymbol{a}_i, \boldsymbol{s} \rangle + e_i$. The $\mathrm{LWE}_{n,q,\chi}$ assumption is that the $\mathrm{LWE}_{n,q,\chi}$ problem is infeasible.*

Sometimes it is convenient to view the vectors $b_i \| \boldsymbol{a}_i$ as the rows of a matrix A, and to redefine $\boldsymbol{s}$ as $(1, -\boldsymbol{s})$. Then, either A is uniform, or there is a vector $\boldsymbol{s}$ whose first coefficient is 1 such that $A \cdot \boldsymbol{s} = \boldsymbol{e}$, where the coefficients of $\boldsymbol{e}$ come from the distribution χ.

For lattice dimension parameter n and number d, GapSVP_γ is the problem of distinguishing whether a n-dimensional lattice has a vector shorter than d or no vector shorter than $\gamma(n) \cdot d$. The two theorems below capture reductions, quantum and classical, from GapSVP to LWE for certain parameters. We state the result in terms of B-bounded distributions.

Definition 2 (B-bounded distributions). *A distribution ensemble $\{\chi_n\}_{n \in \mathbb{N}}$, supported over the integers, is called B-bounded if $\Pr_{e \leftarrow \chi_n}[|e| > B] = \mathsf{negl}(n)$.*

Theorem 1 ([Reg05, Pei09, MM11, MP12], stated as Corollary 2.1 from [Bra12]). *Let $q = q(n) \in \mathbb{N}$ be either a prime power or a product of small (size $\mathrm{poly}(n)$) distinct primes, and let $B \geq \omega(\log n) \cdot \sqrt{n}$. Then there exists an efficient sampleable B-bounded distribution χ such that if there is an efficient algorithm that solves the average-case LWE problem for parameters n, q, χ, then:*

- *There is an efficient quantum algorithm that solves $GapSVP_{\tilde{O}(nq/B)}$ on any n-dimensional lattice.*
- *If $q \geq \tilde{O}(2^{n/2})$, then there is an efficient classical algorithm for $GapSVP_{\tilde{O}(nq/B)}$ on any n-dimensional lattice.*

In both cases, if one also considers distinguishers with sub-polynomial advantage, then we require $B \geq \tilde{O}(n)$ and the resulting approximation factor is slightly larger than $\tilde{O}(n^{1.5}q/B)$.

Theorem 2 (Informal Theorem 1.1 of [BLP+13]). *Solving n-dimensional LWE with $\mathrm{poly}(n)$ modulus implies an equally efficient solution to a worst-case lattice problem (e.g., GapSVP) in dimension $\sqrt{n}$.*

2.2 Identity-Based and Attribute-Based Homomorphic Encryption

In a homomorphic encryption scheme $\mathsf{HE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$, the message space is some ring, and Eval homomorphically evaluates arithmetic circuits over this ring (with addition and multiplication gates). We omit formal definitions and theorems regarding homomorphic encryption, which can be found in referenced papers.

An identity-based HE scheme $\mathsf{IBHE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ has all of the properties of a normal IBE scheme $\mathsf{IBE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ [Sha84, BF03]. Setup generates master keys $(\mathsf{MSK}, \mathsf{MPK})$, $\mathsf{KeyGen}(\mathsf{MSK}, \mathsf{ID})$ outputs a secret key $\mathsf{sk}_{\mathsf{ID}}$ for identity ID, $\mathsf{Enc}(\mathsf{MPK}, \mathsf{ID}, m)$ outputs an encryption c of m under ID, and $\mathsf{Dec}(\mathsf{sk}_{\mathsf{ID}}, c)$ decrypts c (if it is under ID). Standard security properties apply. For example, an IBE scheme is expected to be *collusion-resistant* – in particular, the adversary can ask for many secret keys, as long as the challenge ciphertext is under an unqueried identity.

For some function family $\mathcal{F}$, IBHE's procedure $c \leftarrow \mathsf{Eval}(\mathsf{MPK}, \mathsf{ID}, f, c_1, \ldots, c_t)$ homomorphically evaluates any $f \in \mathcal{F}$ on ciphertexts $\{c_i \leftarrow \mathsf{Enc}(\mathsf{MPK}, \mathsf{ID}, m_i)\}$ under the same ID. Ultimately, $\mathsf{Dec}(\mathsf{sk}_{\mathsf{ID}}, c) = f(m_1, \ldots, m_t)$. We define identity-based (leveled) *fully* homomorphic encryption (IBFHE) in the expected way.

The definition of IBHE can be extended to a multi-identity setting – specifically, Eval could work over ciphertexts under multiple identities. For security to make sense, Dec would require cooperation of all parties whose identities were used in Eval. In this paper, we restrict our attention to the single-identity setting.

An attribute-based HE scheme $\mathsf{ABHE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ has all of the properties of a normal ABE scheme $\mathsf{ABE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ [SW05, GPSW06]. For some relation R, some function family $\mathcal{F}$ and any $f \in \mathcal{F}$, and any ciphertexts $\{c_i \leftarrow \mathsf{Enc}(\mathsf{MPK}, x, m_i)\}$ encrypted under common index x, the ciphertext $c \leftarrow \mathsf{Eval}(\mathsf{MPK}, x, f, c_1, \ldots, c_t)$ can be decrypted (to $f(m_1, \ldots, m_t)$) using a key sk_y for any y for which $R(x, y) = 1$. In an ABE scheme *for circuits*, R can be a circuit of polynomial depth. We define attribute-based (leveled) *fully* homomorphic encryption (ABFHE) in the expected way.

Similar to IBHE, ABHE can be extended so that Eval operates on ciphertexts under multiple indices $x_1, \ldots, x_k$. Regarding decryption, there are different possibilities. For example, the result can only be decrypted using some sk_y for which $R(x_1, y) = \cdots = R(x_k, y) = 1$. Alternatively, the result can be cooperatively decrypted using $\mathsf{sk}_{y_1}, \ldots, \mathsf{sk}_{y_\ell}$ such that for every x_i there is some j such that $R(x_i, y_j) = 1$. We restrict our attention to the single-index setting.

2.3 Other Preliminaries

For n, q, and $\ell = \lfloor \log q \rfloor + 1$, we define the procedures $\mathsf{BitDecomp}$, $\mathsf{BitDecomp}^{-1}$, $\mathsf{Flatten}$ and $\mathsf{Powersof2}$ as described in the Introduction. I_N denotes the N-dimensional identity matrix.

3 Our LWE-Based FHE Scheme

3.1 Basic Encryption Scheme

Here, we formally describe our basic encryption scheme (without homomorphic operations). This description matches the description outlined in the Introduction. In our description, we split up KeyGen into three parts Setup, $\mathsf{SecretKeyGen}$ and $\mathsf{PublicKeyGen}$. We provide two decryption algorithms Dec and MPDec. Dec is sufficient to recover the message μ when it is in a small space (e.g., $\{0,1\}$). MPDec is an algorithm by Micciancio and Peikert [MP12] that can recover any $\mu \in \mathbb{Z}_q$.

- $\mathsf{Setup}(1^\lambda, 1^L)$: Choose a modulus q of $\kappa = \kappa(\lambda, L)$ bits, lattice dimension parameter $n = n(\lambda, L)$, and error distribution $\chi = \chi(\lambda, L)$ appropriately for LWE that achieves at least 2^λ security against known attacks. Also, choose parameter $m = m(\lambda, L) = O(n \log q)$. Let $params = (n, q, \chi, m)$. Let $\ell = \lfloor \log q \rfloor + 1$ and $N = (n+1) \cdot \ell$.
- $\mathsf{SecretKeyGen}(params)$: Sample $\boldsymbol{t} \leftarrow \mathbb{Z}_q^n$. Output $sk = \boldsymbol{s} \leftarrow (1, -t_1, \ldots, -t_n) \in \mathbb{Z}_q^{n+1}$. Let $\boldsymbol{v} = \mathsf{Powersof2}(\boldsymbol{s})$.
- $\mathsf{PublicKeyGen}(params, sk)$: Generate a matrix $B \leftarrow \mathbb{Z}_q^{m \times n}$ uniformly and a vector $\boldsymbol{e} \leftarrow \chi^m$. Set $\boldsymbol{b} = B \cdot \boldsymbol{t} + \boldsymbol{e}$. Set A to be the $(n+1)$-column matrix consisting of $\boldsymbol{b}$ followed by the n columns of B. Set the public key $pk = A$. (*Remark:* Observe that $A \cdot \boldsymbol{s} = \boldsymbol{e}$.)
- $\mathsf{Enc}(params, pk, \mu)$: To encrypt a message $\mu \in \mathbb{Z}_q$, sample a uniform matrix $R \in \{0,1\}^{N \times m}$ and output the ciphertext C given below.

$$C = \mathsf{Flatten}\big(\mu \cdot I_N + \mathsf{BitDecomp}(R \cdot A)\big) \in \mathbb{Z}_q^{N \times N}.$$

- $\mathsf{Dec}(params, sk, C)$: Observe that the first ℓ coefficients of $\boldsymbol{v}$ are $1, 2, \ldots, 2^{\ell-1}$. Among these coefficients, let $v_i = 2^i$ be in $(q/4, q/2]$. Let C_i be the i-th row of C. Compute $x_i \leftarrow \langle C_i, \boldsymbol{v} \rangle$. Output $\mu' = \lfloor x_i / v_i \rceil$.
- $\mathsf{MPDec}(params, sk, C)$ (for q a power of 2): Observe that $q = 2^{\ell-1}$ and the first $\ell - 1$ coefficients of $\boldsymbol{v}$ are $1, 2, \ldots, 2^{\ell-2}$, and therefore if $C \cdot \boldsymbol{v} = \mu \cdot \boldsymbol{v} + small$, then the first $\ell - 1$ coefficients of $C \cdot \boldsymbol{v}$ are $\mu \cdot \boldsymbol{g} + small$, where $\boldsymbol{g} = (1, 2, \ldots, 2^{\ell-2})$. Recover $\mathrm{LSB}(\mu)$ from $\mu \cdot 2^{\ell-2} + small$, then recover the next-least-significant-bit from $(\mu - \mathrm{LSB}(\mu)) \cdot 2^{\ell-3} + small$, etc. (See [MP12] for the general q case.)

Dec is a $\mathsf{BitDecomp}$'d version of Regev's decryption procedure, applied to one row of the ciphertext, which is a $\mathsf{BitDecomp}$'d Regev ciphertext. (The extra rows

will come into play in the homomorphic operations). If C is properly generated, then by the elementary properties of BitDecomp and Powersof2, we have

$$C \cdot \boldsymbol{v} = \mu \cdot \boldsymbol{v} + R \cdot A \cdot \boldsymbol{s} = \mu \cdot \boldsymbol{v} + R \cdot \boldsymbol{e}.$$

Dec only uses the i-th coefficient of the above expression, which is $x_i = \mu \cdot v_i + \langle R_i, \boldsymbol{e} \rangle$. The error $\langle R_i, \boldsymbol{e} \rangle$ has magnitude at most $\|\boldsymbol{e}\|_1$. In general, if $x_i = \mu \cdot v_i + e'$ for some error e' of magnitude at most $q/8$, and if $v_i \in (q/4, q/2]$, then x_i/v_i differs from μ by at most $(q/8)/v_i < 1/2$, and Dec uses rounding to output the correct value of μ. (In the basic scheme, we set χ to ensure that the error is so bounded with overwhelming probability.)

For the basic scheme (without homomorphic operations), one can take n to be quasi-linear in the security parameter λ and $\kappa = O(\log n)$. When allowing homomorphic operations, L represents the circuit complexity of the functions that the scheme correctly evaluates (roughly, L is the multiplicative depth); we provide a detailed analysis later of how L affects the other parameters.

3.2 Security

Observe that $\mathsf{BitDecomp}^{-1}(C) = \mu \cdot G + R \cdot A$, where $G = \mathsf{BitDecomp}^{-1}(I_N)$ is (the transpose of) the "primitive matrix" used by Micciancio and Peikert [MP12] in their construction of lattice trapdoors, and the rows of $R \cdot A$ are simply Regev [Reg05] encryptions of 0 for dimension n. Assuming $\mathsf{BitDecomp}^{-1}(C)$ hides μ, C does as well, since C can be derived by applying BitDecomp. Thus, the security of our basic encryption scheme follows directly from the following lemma, used to prove the security of Regev's encryption scheme [Reg05].

Lemma 1 (Implicit in [Reg05]). *Let params $= (n, q, \chi, m)$ be such that the* $\mathrm{LWE}_{n,q,\chi}$ *assumption holds. Then, for $m = O(n \log q)$ and A, R as generated above, the joint distribution $(A, R \cdot A)$ is computationally indistinguishable from uniform over $\mathbb{Z}_q^{m \times (n+1)} \times \mathbb{Z}_q^{N \times (n+1)}$.*

Concretely, it suffices to take $m > 2n \log q$ [Reg05].

Like Brakerski [Bra12], we can also base security on GapSVP via a classical reduction from LWE [Pei09, BLP+13]. Specifically, Peikert [Pei09] gives a classical reduction of GapSVP to LWE, with the caveat that q must be exponential in n. Brakerski notes that exponential q was unusable in previous FHE schemes, since the ratio of q to the error level B of "fresh" ciphertexts cannot be exponential in n for security reasons (since LLL [LLL82] could be used to break such a scheme), and since B must be very small to permit many homomorphic operations. As in Brakerski's scheme, we do not have that problem. The error bound B of fresh ciphertexts in our scheme does not need to be small to permit many homomorphic operations; we only require q/B to be sub-exponential, and we can therefore permit q to be exponential. Alternatively, we can use a sub-exponential q and base security on GapSVP via Brakerski et al.'s [BLP+13] recent classical reduction to LWE that works even for polynomial-size moduli, with the caveat that, in their reduction, the dimension of the GapSVP instances may be much smaller than the dimension of the LWE instances.

3.3 Homomorphic Operations

Recall that we already described some basic "homomorphic" operations $\mathsf{BitDecomp}$, $\mathsf{BitDecomp}^{-1}$, $\mathsf{Flatten}$, and $\mathsf{Powersof2}$. These will play an important role in analyzing the homomorphic operations supported by our scheme. We remark that $\mathsf{BitDecomp}$ could alternatively decompose with respect to bases other than 2, or according to the Chinese Remainder Theorem.

We provide additional homomorphic operations $\mathsf{MultConst}$, Add, Mult, NAND as follows.

- $\mathsf{MultConst}(C, \alpha)$: To multiply a ciphertext $C \in \mathbb{Z}_q^{N \times N}$ by known constant $\alpha \in \mathbb{Z}_q$, set $M_\alpha \leftarrow \mathsf{Flatten}(\alpha \cdot I_N)$ and output $\mathsf{Flatten}(M_\alpha \cdot C)$. Observe that:

$$\begin{aligned}\mathsf{MultConst}(C, \alpha) \cdot \boldsymbol{v} = M_\alpha \cdot C \cdot \boldsymbol{v} = M_\alpha \cdot (\mu \cdot \boldsymbol{v} + \boldsymbol{e}) = \mu \cdot (M_\alpha \cdot \boldsymbol{v}) + M_\alpha \cdot \boldsymbol{e} \\ = \alpha \cdot \mu \cdot \boldsymbol{v} + M_\alpha \cdot \boldsymbol{e}\end{aligned}$$

  Thus, the error increases by a factor of at most N, regardless of what element $\alpha \in \mathbb{Z}_q$ is used for multiplication. As in "classical" additively homomorphic encryption schemes, we could alternatively perform multiplication-by-constant α by recursively applying Add. But this multiplies the error size by at least α, whereas $\mathsf{MultConst}$ increases the error by at most a factor of N, regardless of α. An example application of $\mathsf{MultConst}$ is that we can perform homomorphic fast Fourier transformations (FFTs) natively over $\mathbb{Z}_q$ without error growth dependent on q. Previously, the error growth depended on the size of the field underlying the FFT [GHS12a, GHS12b], restricting the choice of field.
- $\mathsf{Add}(C_1, C_2)$: To add ciphertexts $C_1, C_2 \in \mathbb{Z}_q^{N \times N}$, output $\mathsf{Flatten}(C_1 + C_2)$. The correctness of this operation is immediate. Note that the addition of messages is over the full base ring $\mathbb{Z}_q$.
- $\mathsf{Mult}(C_1, C_2)$: To multiply ciphertexts $C_1, C_2 \in \mathbb{Z}_q^{N \times N}$, output $\mathsf{Flatten}(C_1 \cdot C_2)$. Observe that:

$$\begin{aligned}\mathsf{Mult}(C_1, C_2) \cdot \boldsymbol{v} = C_1 \cdot C_2 \cdot \boldsymbol{v} = C_1 \cdot (\mu_2 \cdot \boldsymbol{v} + \boldsymbol{e}_2) + \mu_2 \cdot (\mu_1 \boldsymbol{v} + \boldsymbol{e}_1) + C_1 \cdot \boldsymbol{e}_2 \\ = \mu_1 \cdot \mu_2 \cdot \boldsymbol{v} + \mu_2 \cdot \boldsymbol{e}_1 + C_1 \cdot \boldsymbol{e}_2\end{aligned}$$

  As in Add, the multiplication operator is over the full base field $\mathbb{Z}_q$. In Mult, the new error depends on the old errors, the ciphertext C_1, and the message μ_2. The dependence on the old errors seems unavoidable (and normal for LWE-based HE schemes), and observe that C_1 contributes at most a factor N blowup of error, since all components of C_1 are restricted to $\{0, 1\}$. The error growth based on the message μ_2, however, presents a concern. In general, we must address this concern by using homomorphic operations in a way that restricts the message space to small messages. One way to do this is to consider Boolean circuits using only NAND operations: this would restrict the message space to $\{0, 1\}$. We elaborate below.
- $\mathsf{NAND}(C_1, C_2)$: To NAND ciphertexts $C_1, C_2 \in \mathbb{Z}_q^{N \times N}$ that are known to encrypt messages $\mu_1, \mu_2 \in \{0, 1\}$, output $\mathsf{Flatten}(I_N - C_1 \cdot C_2)$. Observe that:

$$\mathsf{NAND}(C_1, C_2) \cdot \boldsymbol{v} = (I_N - C_1 \cdot C_2) \cdot \boldsymbol{v} = (1 - \mu_1 \cdot \mu_2) \cdot \boldsymbol{v} - \mu_2 \cdot \boldsymbol{e}_1 - C_1 \cdot \boldsymbol{e}_2$$

Note here that the NAND homomorphic operation maintains the invariant that if the input messages are in $\{0,1\}$, then the output ciphertext will also encryption of $\{0,1\}$, thus guaranteeing small messages. Note that since $\mu_2 \in \{0,1\}$, the error is increased by a factor of at most $N+1$.

Circuits. By iteratively applying the homomorphic operations above, different types of (bounded-depth) circuits may be homomorphically computed while maintaining correctness of decryption.

The simplest case to analyze is the case of Boolean circuits computed over encryptions of $\{0,1\}$ values. In this case, the circuit can be converted to use only NAND gates, and through appropriate leveled application of the NAND homomorphic operation, the final ciphertext's error will be bounded by $(N+1)^L \cdot B$, where L is the NAND-depth of the circuit, and B is the original bound on the error of a fresh encryption of $\{0,1\}$.

More generally, with more care, we may consider arithmetic circuits over $\mathbb{Z}_q$ that make use of gates that perform addition, multiplication, or multiplication by a known constant. However, as we have seen in the case of multiplication gates, the error growth may depend on the values being encrypted in intermediate computations. One way to deal with this is to focus on situations where (1) all input values are known to encrypt values bounded by some value T, and (2) the arithmetic circuit is chosen to guarantee that all intermediate values are also bounded by T' whenever the circuit inputs are constrained to values bounded by T. In such a situation, the final ciphertext's error will be bounded by $(N+T')^L \cdot B$, where L is the depth of the arithmetic circuit, and B is the original bound on the error of fresh encryptions of values smaller than T. For example, in this way, we can homomorphically evaluate polynomials of degree d in this large-message-space variant when the initial messages are bounded by roughly $q^{1/d}$, achieving a scheme that is "somewhat homomorphic" [Gen09]. Another example application would be to convert encryptions of a polynomially bounded set of small values to encryptions of binary values, by using an appropriate arithmetic circuit for the conversion. Once converted to encryptions of binary values, a NAND-based Boolean circuit could be used for further computations.

3.4 Parameters, Performance and Optimizations

Suppose that Flatten'd ciphertexts C_1, C_2 encrypt $\mu_1, \mu_2 \in \{0,1\}$ under approximate eigenvector $\boldsymbol{v}$ with B-bounded error – that is, $C_i \cdot \boldsymbol{v} = \mu_i \cdot \boldsymbol{v} + \boldsymbol{e}_i$ where $|\boldsymbol{e}_i|_\infty \leq B$. Then $C^{\mathsf{NAND}} \leftarrow \mathsf{NAND}(C_1, C_2)$ encrypts $\mathsf{NAND}(\mu_1, \mu_2) \in \{0,1\}$ under $\boldsymbol{v}$ with $(N+1)B$-bounded error. As long as $q/B > 8(N+1)^L$, we can evaluate a depth-L circuit of NANDs over B-bounded ciphertexts to obtain a $q/8$-bounded ciphertext, which Dec will decrypt correctly.

As in previous LWE-based FHE schemes, n (hence N) must increase linearly with $\log(q/B)$ to maintain fixed 2^λ security against known attacks, so q/B grows more like $\exp(L \log L)$. We will brush such issues under the rug and view n as a fixed parameter. Choosing χ so that B is not too large, and since in practice there is no reason to have $\kappa = \log q$ grow super-linearly with n, we have $\kappa =$

$O(L \log N) = O(L(\log n + \log \kappa)) = O(L \log n)$, similar to [BGV12,Bra12]. Given that the NAND procedure is dominated by multiplication of two $N \times N$ matrices for $N = O(n\kappa) = \tilde{O}(nL)$, we have the following theorem to characterize the performance of our FHE scheme.

Theorem 3. *For dimension parameter n and depth parameter L, our FHE scheme evaluates depth-L circuits of* NAND *gates with $\tilde{O}((nL)^\omega)$ field operations per gate, where $\omega < 2.3727$ is the matrix multiplication exponent.*

This compares favorably with previous LWE-based FHE schemes, which all have at least $\tilde{O}(n^3 L)$ field operations per gate [BV11b, BGV12, Bra12].

Theorem 3 hides some factors, both good and bad. On the good size, it hides the fact that ciphertext matrices in our scheme have 0/1 entries, and therefore can be multiplied faster than if they were general matrices over $\mathbb{Z}_q$. In previous LWE-based FHE schemes, the field operations involve multiplying a small number with a general number of $\mathbb{Z}_q$, which has complexity $\tilde{O}(\kappa) = \tilde{O}(L)$. So, previous LWE-based FHE schemes have real complexity $\tilde{O}(n^3 L^2)$ whereas ours remains $\tilde{O}((nL)^\omega)$. On the bad side, Theorem 3 hides logarithmic factors in the dimension of the ciphertext matrices, since $N = O(n\kappa) = O(nL \log n)$. We note that typically n will dominate L, since for very deep circuits, one would want to use Gentry's bootstrapping technique [Gen09] to make the per-gate computation *independent* of L.

Since bootstrapping involves homomorphically evaluating the decryption function, and since Dec is essentially Regev decryption [Reg05], bootstrapping works as in previous LWE-based FHE schemes. In particular, we can use techniques from [BV11b] to reduce the dimension and modulus-size of the ciphertext before bootstrapping, so that the complexity of decryption (and hence bootstrapping) is completely independent of the depth L of the circuit that was evaluated to arrive at that ciphertext. Regev decryption can be evaluated in $O(\log n)$ depth. Due to the logarithmic depth, one can take q/B to be quasi-polynomial in n, and base security on LWE for quasi-polynomial factors.

4 Our Identity-Based and Attribute-Based FHE Schemes

Identity-based encryption (IBE) [Sha84, BF03] and attribute-based encryption (ABE) [SW05, GPSW06] are designed to provide more flexible access control of encrypted data than a traditional public key infrastructure. Traditionally, IBE and ABE do not offer any computation over the encrypted data. However, access control of encrypted data remains important even (or especially) when the data is encrypted homomorphically. (See [CHT13] for a nice discussion of applications.)

Unfortunately, while there are some IBE schemes that allow simple homomorphic operations [GHV10, CHT13], it has remained a stubborn open problem [Nac10, GHV10, Bra12, CHT13] to construct an IBE scheme that allows fully or even "somewhat" homomorphic encryption. Previously it was mentioned [Bra12, CHT13]) that instead of building an FHE scheme on Regev's encryption scheme as we do in Section 3, one can alternatively use the "dual-Regev"

system [GPV08], for which it is known how to generate identity-based keys (see also [ABB10a, ABB10b, CHKP10]). However, making the encryption/decryption keys identity-based only solves half of the problem, and yields only a "weak" form of identity-based FHE. In all previous FHE schemes, there is also an "evaluation key" required for homomorphic evaluation. This evaluation key is user-specific and is not "identity-based", in the sense that it cannot be computed non-interactively from the user's identity. But having to *obtain* this evaluation key undermines the main appeal of IBE: its non-interactivity. Thus, identity-based FHE (IBFHE) has remained wide open, and attribute-based FHE (ABFHE) seems even more difficult to construct.

Interestingly, however, our new FHE scheme does not have evaluation keys. To perform evaluation, the evaluator only needs to know some basic parameters of the scheme (like n, m and ℓ).

The absence of evaluation keys allows us to construct the first IBFHE scheme. We describe a simple "compiler" that transforms any LWE-based IBE scheme (that satisfies certain natural properties) into a IBFHE. All LWE-based IBE schemes that we know of (e.g., [GPV08, ABB10a, ABB10b, CHKP10]) can be described so as to have the required properties.

1. **Property 1 (Ciphertext and decryption key vectors):** The decryption key for identity ID, and a ciphertext for ID, are vectors $\boldsymbol{s}_{\mathsf{ID}}, \boldsymbol{c}_{\mathsf{ID}} \in \mathbb{Z}_q^{n'}$ for some n'. The first coefficient of $\boldsymbol{s}_{\mathsf{ID}}$ is 1.
2. **Property 2 (Small Dot Product):** If $\boldsymbol{c}_{\mathsf{ID}}$ encrypts 0, then $\langle \boldsymbol{c}_{\mathsf{ID}}, \boldsymbol{s}_{\mathsf{ID}} \rangle$ is "small".
3. **Property 3 (Security):** Encryptions of 0 are indistinguishable from uniform vectors over $\mathbb{Z}_q$ (under LWE).

Theorem 4. *We can compile an IBE scheme* E *with the above properties into a related IBFHE scheme.*

Proof. The IBFHE uses E's Setup and KeyGen algorithms, supplementing E's MPK with the basic parameters for our FHE scheme (such as m, ℓ). Let $N = (n+1) \cdot \ell$ for $\ell = \lfloor \log q \rfloor + 1$, as usual. To encrypt $\mu \in \{0, 1\}$, the encrypter generates N encryptions of 0 using $\mathsf{E.Enc}$, sets C'_{ID} to be the $N \times (n+1)$ matrix whose rows are these ciphertexts, and outputs $C_{\mathsf{ID}} = \mathsf{Flatten}(\mu \cdot I_N + \mathsf{BitDecomp}(C'_{\mathsf{ID}}))$. Suppose $\boldsymbol{s}_{\mathsf{ID}}$ is the decryption key for ID, as above, and let $\boldsymbol{v}_{\mathsf{ID}} = \mathsf{Powersof2}(\boldsymbol{s}_{\mathsf{ID}})$. The decrypter runs our FHE decryption algorithm $\mathsf{Dec}(\boldsymbol{v}_{\mathsf{ID}}, C_{\mathsf{ID}})$ to recover μ. Homomorphic operations are as in Section 3.3.

Decryption is correct, since $C_{\mathsf{ID}} \cdot \boldsymbol{v}_{\mathsf{ID}} = \mu \cdot \boldsymbol{v}_{\mathsf{ID}} + C'_{\mathsf{ID}} \cdot \boldsymbol{s}_{\mathsf{ID}} = \mu \cdot \boldsymbol{v}_{\mathsf{ID}} + \mathit{small}$, where $C'_{\mathsf{ID}} \cdot \boldsymbol{s}_{\mathsf{ID}}$ is a small vector by Property 2. In this setting Dec recovers $\mu \in \{0, 1\}$. Any adversary that breaks the semantic security of our IBFHE scheme can distinguish C'_{ID} from a uniform matrix over $\mathbb{Z}_q$, and therefore distinguish LWE by Property 3.

For ABFHE, our approach begins by re-interpreting the decryption process in the Gorbunov et al. (GVW) ABE scheme [GVW13]. To decrypt a ABE ciphertext under x with sk_y for which $R(x, y) = 1$, we view the decrypter as deriving a

"sub-key" $\boldsymbol{s}_{x,y}$ associated to x. This sub-key will satisfy something similar to Property 2 above – i.e., if $\boldsymbol{c}_x$ encrypts 0 under x, then $\langle \boldsymbol{c}_x, \boldsymbol{s}_{x,y} \rangle$ is "small". Viewing GVW in this way allows us to apply our compiler above.

We provide more details of our identity-based and attribute-based FHE constructions in the full version of the paper.

Acknowledgments. We gratefully thank Shai Halevi for his collaboration on this work. We also thank Boaz Barak and the anonymous CRYPTO reviewers for their many helpful comments.

References

[ABB10a] Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (h)ibe in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010)

[ABB10b] Agrawal, S., Boneh, D., Boyen, X.: Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 98–115. Springer, Heidelberg (2010)

[BB12a] Barak, B., Brakerski, Z.: Building the swiss army knife. Windows on Theory Blog (2012), `http://windowsontheory.org/2012/05/02/building-the-swiss-army-knife`

[BB12b] Barak, B., Brakerski, Z.: The swiss army knife of cryptography. Windows on Theory Blog (2012), `http://windowsontheory.org/2012/05/01/the-swiss-army-knife-of-cryptography`

[BF03] Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. SIAM J. of Computing 32(3), 586–615 (2003); Extended abstract in Kilian, J. (ed.): CRYPTO 2001. LNCS, vol. 2139, pp. 586–615. Springer, Heidelberg (2001)

[BGV12] Brakerski, Z., Gentry, C., Vaikuntanathan, V.: Fully homomorphic encryption without bootstrapping. In: Innovations in Theoretical Computer Science, ITCS 2012 (2012), `http://eprint.iacr.org/2011/277`

[BLP+13] Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: STOC, pp. 575–584 (2013)

[Bra12] Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical gapSVP. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 868–886. Springer, Heidelberg (2012)

[BV11a] Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-LWE and security for key dependent messages. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 505–524. Springer, Heidelberg (2011)

[BV11b] Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: FOCS, pp. 97–106 (2011), `http://eprint.iacr.org/2011/344`

[CHKP10] Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010)

[CHT13] Clear, M., Hughes, A., Tewari, H.: Homomorphic encryption with access policies: Characterization and new constructions. In: Youssef, A., Nitaj, A., Hassanien, A.E. (eds.) AFRICACRYPT 2013. LNCS, vol. 7918, pp. 61–87. Springer, Heidelberg (2013)

[CLT13] Coron, J.-S., Lepoint, T., Tibouchi, M.: Practical multilinear maps over the integers. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 476–493. Springer, Heidelberg (2013)

[CMNT11] Coron, J.-S., Mandal, A., Naccache, D., Tibouchi, M.: Fully homomorphic encryption over the integers with shorter public keys. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 487–504. Springer, Heidelberg (2011)

[CNT12] Coron, J.-S., Naccache, D., Tibouchi, M.: Public key compression and modulus switching for fully homomorphic encryption over the integers. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 446–464. Springer, Heidelberg (2012)

[Gen09] Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC, pp. 169–178 (2009)

[Gen10] Gentry, C.: Toward basing fully homomorphic encryption on worst-case hardness. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 116–137. Springer, Heidelberg (2010)

[GGH13a] Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 1–17. Springer, Heidelberg (2013)

[GGH+13b] Garg, S., Gentry, C., Halevi, S., Sahai, A., Waters, B.: Attribute-based encryption for circuits from multilinear maps. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 479–499. Springer, Heidelberg (2013)

[GH11a] Gentry, C., Halevi, S.: Fully homomorphic encryption without squashing using depth-3 arithmetic circuits. In: FOCS, pp. 107–109 (2011)

[GH11b] Gentry, C., Halevi, S.: Implementing gentry's fully-homomorphic encryption scheme. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 129–148. Springer, Heidelberg (2011)

[GHS12a] Gentry, C., Halevi, S., Smart, N.P.: Fully homomorphic encryption with polylog overhead. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 465–482. Springer, Heidelberg (2012)

[GHS12b] Gentry, C., Halevi, S., Smart, N.P.: Homomorphic evaluation of the AES circuit. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 850–867. Springer, Heidelberg (2012)

[GHV10] Gentry, C., Halevi, S., Vaikuntanathan, V.: A simple BGN-type cryptosystem from LWE. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 506–522. Springer, Heidelberg (2010)

[GPSW06] Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM CCS, pp. 89–98 (2006)

[GPV08] Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC, pp. 197–206. ACM (2008)

[GVW13] Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: STOC, pp. 545–554 (2013)

[LATV12] López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: STOC, pp. 1219–1234 (2012)

[LLL82] Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261, 515–534 (1982), doi:10.1007/BF01457454

[LPR10] Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010)

[MM11] Micciancio, D., Mol, P.: Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 465–484. Springer, Heidelberg (2011)

[MP12] Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012)

[Nac10] Naccache, D.: Is theoretical cryptography any good in practice? Invited talk at Crypto/CHES 2010 (2010), http://www.iacr.org/workshops/ches/ches2010

[Pei09] Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: STOC, pp. 333–342 (2009)

[RAD78] Rivest, R., Adleman, L., Dertouzos, M.L.: On data banks and privacy homomorphisms. In: Foundations of Secure Computation, pp. 169–180 (1978)

[Reg05] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC, pp. 84–93 (2005)

[Sha84] Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)

[Str69] Strassen, V.: Gaussian elimination is not optimal. Numer. Math. 13, 354–356 (1969)

[SV10] Smart, N.P., Vercauteren, F.: Fully homomorphic encryption with relatively small key and ciphertext sizes. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 420–443. Springer, Heidelberg (2010)

[SW05] Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)

[vDGHV10] van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. EUROCRYPT, pp. 24–43. Springer, Heidelberg (2010)

[Wil12] Williams, V.V.: Multiplying matrices faster than coppersmith-winograd. In: STOC, pp. 887–898 (2012)

A Uniform Min-Max Theorem with Applications in Cryptography*

Salil Vadhan and Colin Jia Zheng

School of Engineering and Applied Sciences
Harvard University
Cambridge, Massachusetts
{salil,colinz}@seas.harvard.edu

Abstract. We present a new, more constructive proof of von Neumann's Min-Max Theorem for two-player zero-sum game — specifically, an algorithm that builds a near-optimal mixed strategy for the second player from several best-responses of the second player to mixed strategies of the first player. The algorithm extends previous work of Freund and Schapire (Games and Economic Behavior '99) with the advantage that the algorithm runs in poly(n) time even when a pure strategy for the first player is a distribution chosen from a set of distributions over $\{0,1\}^n$. This extension enables a number of additional applications in cryptography and complexity theory, often yielding uniform security versions of results that were previously only proved for nonuniform security (due to use of the non-constructive Min-Max Theorem).

We describe several applications, including a more modular and improved uniform version of Impagliazzo's Hardcore Theorem (FOCS '95), showing impossibility of constructing succinct non-interactive arguments (SNARGs) via black-box reductions under uniform hardness assumptions (using techniques from Gentry and Wichs (STOC '11) for the nonuniform setting), and efficiently simulating high entropy distributions within any sufficiently nice convex set (extending a result of Trevisan, Tulsiani and Vadhan (CCC '09)).

1 Introduction

Von Neumann's Min-Max Theorem (or Linear Programming Duality, finite-dimensional Hahn-Banach Theorem) has proved to be an extremely useful tool in theoretical computer science. Consider a zero-sum game between two players where for every mixed strategy V for Player 1 (as a distribution over his strategy space $\mathcal{V}$), Player 2 has a response $W \in \mathcal{W}$ that guarantees $\mathbb{E}[F(V,W)] \geq 0$, where F (payoff) can be an arbitrary function. The Min-Max Theorem says that there must exist a Player 2's mixed strategy W^* (as a distribution over his strategy space $\mathcal{W}$) that guarantees $\mathbb{E}[F(V,W^*)] \geq 0$ for *all* strategies $V \in \mathcal{V}$ of Player 1.

* Supported by NSF grant CCF-1116616 and US-Israel BSF grant 2010196. A full version of this paper [VZ2] to appear on the Cryptology ePrint Archive.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 93–110, 2013.

The Min-Max Theorem gives rise to a number of results in cryptography and complexity theory such as Impagliazzo's Hardcore Theorem [Imp], equivalence of different notions of computational entropy [BSW], the Dense Model Theorem [RTTV], leakage-resilient cryptography [DP,FR], efficient simulation of high entropy distributions [TTV], impossibility of constructing succinct non-interactive arguments (SNARGs) via black-box reductions [GW], and simple construction of pseudorandom generators from one-way functions [VZ1]. In a typical application like these, Player 1 chooses V from a convex set $\mathcal{V}$ of distributions over $\{0,1\}^n$, and Player 2 chooses W from a set $\mathcal{W}$ of (possibly randomized) boolean functions $\{0,1\}^n \to \{0,1\}$ and receives payoff $F(V,W) = \mathbb{E}[W(V)]$ i.e. function W's expected output when input is drawn from the distribution V. For example, $\mathcal{V}$ contains all high entropy distributions over $\{0,1\}^n$ and $\mathcal{W}$ contains all boolean functions of small circuit size.

A limitation of the Min-Max Theorem is that it is highly non-constructive; it only asserts the existence of the optimal strategy W^* but does not say how it can be found (algorithmically). Consequently, applications of the Min-Max Theorem only give rise to results about nonuniform boolean circuits, rather than uniform algorithms (e.g. we set cryptographic protocols based on nonuniform hardness rather than uniform hardness assumptions).

To overcome this, we consider the natural algorithmic task of constructing such an optimal strategy W^* for Player 2, given an efficient algorithm for F. When the sizes of strategy spaces $\mathcal{V}$ and $\mathcal{W}$ are small (e.g. polynomial) this can be done by linear programming, for which efficient algorithms are well-known. However, applications in cryptography and complexity theory such as ones just mentioned involve exponentially large strategy spaces, and an optimal strategy W^* cannot be found in polynomial time in general. Thus we also require that, given any mixed strategy V for Player 1, not only does there exist a strategy $W \in \mathcal{W}$ for Player 2 with $\mathbb{E}[F(V,W)] \geq 0$, but such response W can be obtained efficiently by an oracle (or an efficient uniform algorithm).

Assuming such an oracle, Freund and Schapire [FS] show how to find an approximately optimal W^* for Player 2 in polynomial time and by making $O((\log|\mathcal{V}|)/\epsilon^2)$ adaptive oracle queries, using the idea of multiplicative weight updates. However, their algorithm still falls short in some of aforementioned applications where $\mathcal{V}$ is a set of distributions over $\{0,1\}^n$, and thus $\mathcal{V}$ can have doubly-exponentially many vertices. For example, consider the set of distributions on $\{0,1\}^n$ of min-entropy at least k; the vertices of $\mathcal{V}$ are uniform distributions on a subset of size 2^k, and there are $\binom{2^n}{2^k}$ such subsets.

We present a Uniform Min-Max Theorem that efficiently finds an approximately optimal strategy W^* for Player 2, given an oracle that for any of Player 1's mixed strategy $V \in \mathcal{V}$ returns some Player 2's strategy that guarantees reasonable payoff, even when $\mathcal{V}$ is a (sufficiently nice) set of distributions over $\{0,1\}^n$. Our algorithm is inspired by the proof of Uniform Hardcore Theorem of Barak, Hardt, and Kale [BHK]. Like [BHK], our algorithm uses "relative entropy (KL) projections" together with multiplicative weight updates (a technique originally due to Herbster and Warmuth [HW]). Our contribution is the formulation

of this algorithm as providing a Uniform Min-Max Theorem. An advantage of this formulation is that it is more modular, and not specific to the Hardcore Theorem. Consequently it immediately enables a number of applications, including deriving uniform versions of many of the aforementioned results, where we now deal with algorithms rather than nonuniform boolean circuits. Even for the Hardcore Theorem, where the uniform version was already known [Hol1,BHK], there are several advantages to deducing it using the Uniform Min-Max Theorem.

Uniform Hardcore Theorem. Impagliazzo's Hardcore Theorem ([Imp] and later strengthened in [KS,Hol1,BHK]) is a fundamental result in complexity theory that says if a boolean function f is somewhat hard on average, then there must be a subset of inputs (the hardcore) on which f is extremely hard, and outside of which f is easy. There are two approaches to proving the theorem. One is constructive [Imp,KS,Hol1,BHK] and leads to a *Uniform Hardcore Theorem* where hardness of f is measured against uniform algorithms, rather than nonuniform boolean circuits, and has found several applications in cryptography [KS,Hol1,Hol2,HHR,HRV]. However, the existing proofs turn out to be adhoc and do not achieve all of the optimal parameters simultaneously for a Uniform Hardcore Theorem. Another approach due to Nisan [Imp] (and strengthened in [Hol1]) uses the (non-constructive) Min-Max Theorem and has the advantage of simplicity, but is restricted to the nonuniform measure of hardness.

In Section 4, we show that by replacing the use of Min-Max Theorem in the proof of Nisan [Imp] or Holenstein [Hol1] with our Uniform Min-Max Theorem, we obtain a new proof of the Uniform Hardcore Theorem with the advantages of (i) optimal hardcore density; (ii) optimal complexity blow-up; and (iii) modularity and simplicity.

Construction of Pseudorandom Generators from One-Way Functions. Recently, we [VZ1] obtained a simplified and more efficient construction of pseudorandom generators from arbitrary one-way functions, building on the work of [HRV]. Key to the simplification is a new characterization of a computational analogue of Shannon entropy, whose proof in the nonuniform setting involves the Min-Max Theorem. Using the Uniform Min-Max Theorem instead, we proved our characterization of pseudoentropy in the uniform setting, and hence obtain (simpler) pseudorandom generator from arbitrary one-way functions that are secure against efficient algorithms. We refer to the full version [VZ2] for a more detailed discussion.

Impossibility of Black-Box Construction of Succinct Non-interactive Argument. A result of Gentry and Wichs [GW] shows that there is no black-box construction of succinct non-interactive arguments (SNARGs) from any natural cryptographic assumption. Their result relies on the (mild) assumption that there exist *hard subset membership problems*, which is equivalent to the existence of subexponentially hard one-way functions. One limitation is that they need to assume nonuniformly secure one-way functions, in part due to their use of the non-constructive Min-Max theorem (in [GW] Lemma 3.1).

In Section 5, we show how to obtain the analogous result in the *uniform setting* by using the Uniform Min-Max Theorem. More specifically, assuming that there exist subexponentially hard one-way functions that are secure against uniform algorithms, we show that there is no construction of SNARGs whose security can be reduced in a black-box way to a cryptographic assumption against uniform algorithms (unless the assumption is already false).

Simulating Arbitrary Distributions within a Convex Set. In the full version [VZ2], we apply the Uniform Min-Max Theorem to show a result analogous to the main result of Trevisan, Tulsiani, and Vadhan [TTV], which (informally) says that any high min-entropy distribution X is indistinguishable from some high min-entropy distribution Y of *low complexity*. It is shown in [TTV] that such results can be used to deduce (versions of) the Dense Model Theorem [GT,TZ,RTTV], the Hardcore Theorem [Imp], and the Weak Regularity Lemma [FK], by translating the problem to a simpler one where the unknown distribution X is replaced with the low complexity distribution Y that can be efficiently analyzed and manipulated.

Our result is more general than [TTV] in the sense that we are no longer restricted to distributions of high min-entropy. We show that for any sufficiently "nice" convex set of distributions $\mathcal{V}$, any distribution $X \in \mathcal{V}$ is indistinguishable from some distribution $Y \in \mathcal{V}$ where Y has "low complexity" (for several slightly different definitions of complexity than [TTV]). One application of this result is a slight strengthening of the Weak Regularity Lemma of Frieza and Kannan [FK] that achieves better parameters for graphs that are not dense. Another application is deducing an "efficient" version of a technical lemma of [GW]. (The efficient version has been independently proved by Chung, Lui, and Pass [CLP] and applied in the context of distributional zero-knowledge). We note that our result has an average-case variant, which contains as special case a recent result of Pietrzak and Jetchev [PJ] on leakage-resilient cryptography.

1.1 Paper Organization

Basic notions from information theory including KL projection are defined in Section 2. In Section 3 we state and prove the Uniform Min-Max Theorem, and show that it also implies the standard Min-Max Theorem. In Section 4, 5, we describe two applications of the Uniform Min-Max Theorem (other applications can be found in the full version [VZ2]).

2 Preliminaries

Notations. For a natural number n, $[n]$ denotes the set $\{1, \ldots, n\}$, U_n denotes the uniform distribution on binary strings of length n. For a finite set Σ, U_Σ denotes the uniform distribution on Σ. For a distribution X, $\mathrm{supp}(X)$ denotes the support of X, and $x \leftarrow X$ means x is a random sample drawn from distribution X. We write $\mathrm{Avg}_{a \le i \le b}$ as a shorthand for the average over all $i \in \{a, \ldots, b\}$. $\mathrm{Conv}(\cdot)$ denotes the convex hull.

For more background on entropy and proofs of the lemmas below, see [CT].

Definition 2.1 (Entropy). *For a random variable X, the* (Shannon) entropy *of X is defined to be*

$$\mathrm{H}(X) = \mathop{\mathbb{E}}_{x \leftarrow X}\left[\log \frac{1}{\Pr[X = x]}\right].$$

The min-entropy *of X is defined to be*

$$\mathrm{H}_\infty(X) = \min_{x \in \mathrm{supp}(X)}\left(\log \frac{1}{\Pr[X = x]}\right).$$

The notion of *KL divergence* from random variable A to random variable B is closely related to Shannon entropy; intuitively it measures how dense A is within B, on average (with 0 divergence representing maximum density, i.e. $A = B$, and large divergence meaning that A is concentrated in a small portion of B).

Definition 2.2 (KL divergence). *For random variables A and B, the* KL divergence from A to B *is defined to be*

$$\mathrm{KL}(A \parallel B) = \mathop{\mathbb{E}}_{a \leftarrow A}\left[\log \frac{\Pr[A = a]}{\Pr[B = a]}\right],$$

or conventionally $+\infty$ if $\mathrm{supp}(A) \not\subseteq \mathrm{supp}(B)$.

For random variables (X, A) and (Y, B), the conditional KL divergence from $A|X$ to $B|Y$ *is defined to be*

$$\mathrm{KL}((A|X) \parallel (B|Y)) = \mathop{\mathbb{E}}_{(x,a) \leftarrow (X,A)}\left[\log \frac{\Pr[A = a|X = x]}{\Pr[B = a|Y = x]}\right].$$

Thus, conditional KL divergence captures the expected KL divergence from $A|_{X=x}$ to $B|_{Y=x}$, over $x \leftarrow X$. Like Shannon entropy, it has a chain rule:

Proposition 2.1 (Chain rule for KL divergence). $\mathrm{KL}(X, A \parallel Y, B) = \mathrm{KL}(X \parallel Y) + \mathrm{KL}((A|X) \parallel (B|Y))$.

Note however, that the KL divergence is *not* a metric; it is not symmetric and does not satisfy the triangle inequality.

Definition 2.3 (KL projection). *Let X be a distribution on Σ, and $\mathcal{V}$ be a non-empty closed convex set of distributions on Σ. $Y^* \in \mathcal{V}$ is called a* KL projection of X on $\mathcal{V}$ *if*

$$Y^* = \arg\min_{Y \in \mathcal{V}} \mathrm{KL}(Y \parallel X).$$

A nice property of KL projection is the following geometric structure (see [CT], Chap 11, Section 6):

Theorem 2.1 (Pythagorean theorem). *Let $\mathcal{V}$ be a non-empty closed convex set of distributions on Σ. Let Y^* be a KL projection of X on $\mathcal{V}$. Then for all $Y \in \mathcal{V}$,*

$$\mathrm{KL}(Y \parallel Y^*) + \mathrm{KL}(Y^* \parallel X) \leq \mathrm{KL}(Y \parallel X).$$

In particular,

$$\mathrm{KL}(Y \parallel Y^*) \leq \mathrm{KL}(Y \parallel X).$$

Assuming $\mathrm{KL}(Y^* \parallel X)$ is finite, then Pythagorean theorem implies that the KL projection Y^* is unique: for any $Y \in \mathcal{V}$ which is also a KL projection, the theorem implies $\mathrm{KL}(Y \parallel Y^*) = 0$, which holds only when $Y = Y^*$.

Finding the exact KL projection is often computationally infeasible, so we consider *approximate KL projection*:

Definition 2.4 (Approximate KL projection). *We say Y^* is a* σ-approximate KL projection *of X on $\mathcal{V}$, if $Y^* \in \mathcal{V}$ and for all $Y \in \mathcal{V}$,*

$$\mathrm{KL}(Y \parallel Y^*) \leq \mathrm{KL}(Y \parallel X) + \sigma.$$

3 A Uniform Min-Max Theorem

Consider a zero-sum game between two players, where the space of pure strategies for Player 1 is $\mathcal{V}$, the space of pure strategies for Player 2 is $\mathcal{W}$, and $\mathcal{V}$ is an arbitrary subset of distributions over $[N]$. In this section we present a Uniform Min-Max Theorem that efficiently finds an approximately optimal strategy $W^* \in \mathrm{Conv}(\mathcal{W})$ for Player 2, given an oracle which, when fed any of Player 1's mixed strategies $V \in \mathrm{Conv}(\mathcal{V})$, returns a strategy for Player 2 that guarantees good payoff. Our algorithm is inspired by the proof of Uniform Hardcore Theorem of Barak, Hardt, and Kale [BHK]. Like [BHK], our algorithm uses "relative entropy (KL) projections" together with multiplicative weight updates (a technique originally due to Herbster and Warmuth [HW]).

We first state the theorem and mention how it implies standard Min-Max Theorem.

Theorem 3.1 (A Uniform Min-Max Theorem). *Consider a two-player zero-sum game where the sets of pure strategies for Player 1 and Player 2 are $\mathcal{V} \subseteq \{$distributions over $[N]\}$ and $\mathcal{W}$, and the payoff to Player 2 is defined to be $F(V, W) = \mathbb{E}_V[f(V, W)]$ for some function $f : [N] \times \mathcal{W} \to [-k, k]$. Then for every $0 < \epsilon \leq 1$ and $S \geq \max_{V \in \mathrm{Conv}(\mathcal{V})} \mathrm{KL}(V \parallel V^{(1)})/\epsilon^2$, after S iterations Algorithm 3.1 (Finding Universal Strategy) always outputs a mixed strategy W^* for Player 2 such that*

$$\min_{V \in \mathcal{V}} F(V, W^*) \geq \operatorname*{Avg}_{1 \leq i \leq S} F(V^{(i)}, W^{(i)}) - O(k\epsilon).$$

(This holds regardless of the arbitrary choice of $W^{(i)}$ and $V^{(i+1)}$ in the algorithm.)

In particular, it suffices to take $S \geq (\log N - \min_{V \in \mathcal{V}} \mathrm{H}(V))/\epsilon^2$ if we set $V^{(1)} = U_{[N]} \in \mathrm{Conv}(\mathcal{V})$.

Choose an initial strategy $V^{(1)} \in \mathrm{Conv}(\mathcal{V})$ for Player 1
for $i \leftarrow 1$ **to** S **do**
  Obtain an arbitrary strategy $W^{(i)} \in \mathcal{W}$ for Player 2
  Weight Update:
  Let $V^{(i)'}$ be such that $\Pr[V^{(i)'} = x] \propto e^{-\epsilon \cdot f(x,W^{(i)})/2k} \cdot \Pr[V^{(i)} = x]$
  Projection:
  $V^{(i+1)} \leftarrow$ an arbitrary ϵ^2-approx KL projection of $V^{(i)'}$ on $\mathrm{Conv}(\mathcal{V})$
end
Let W^* be the mixed strategy for Player 2 uniform over $W^{(1)}, \ldots, W^{(S)}$
return W^*

Algorithm 3.1. Finding Universal Strategy

We now describe how Theorem 3.1 implies the original Min-Max Theorem, which says

$$\max_{W \in \mathrm{Conv}(\mathcal{W})} \min_{V \in \mathcal{V}} F(V,W) = \min_{V \in \mathrm{Conv}(\mathcal{V})} \max_{W \in \mathcal{W}} F(V,W).$$

For each i, take $W^{(i)}$ to be Player 2's best response to Player 1's mixed strategy $V^{(i)}$, i.e. $F(V^{(i)}, W^{(i)}) = \max_{W \in \mathcal{W}} F(V^{(i)}, W)$. Theorem 3.1 says for every $\lambda = O(k\epsilon) > 0$, by setting an appropriate $V^{(1)}$ and sufficiently large S, there exists $W^* \in \mathrm{Conv}(\mathcal{W})$ with

$$\begin{aligned}
\min_{V \in \mathcal{V}} F(V,W^*) &\geq \operatorname*{Avg}_{1 \leq i \leq S} F(V^{(i)}, W^{(i)}) - \lambda \\
&= \operatorname*{Avg}_{1 \leq i \leq S} \max_{W \in \mathcal{W}} F\left(V^{(i)}, W\right) - \lambda \\
&\geq \min_{V \in \mathrm{Conv}(\mathcal{V})} \max_{W \in \mathcal{W}} F\left(V, W\right) - \lambda,
\end{aligned}$$

where the last inequality holds because for every i, $\max_{W \in \mathcal{W}} F\left(V^{(i)}, W\right) \geq \min_{V \in \mathrm{Conv}(\mathcal{V})} \max_{W \in \mathcal{W}} F\left(V, W\right)$. Thus, for every $\lambda > 0$,

$$\max_{W \in \mathrm{Conv}(\mathcal{W})} \min_{V \in \mathcal{V}} F(V,W) \geq \min_{V \in \mathrm{Conv}(\mathcal{V})} \max_{W \in \mathcal{W}} F(V,W) - \lambda$$

Taking $\lambda \to 0$ gives the Min-Max Theorem.

Proof (of Theorem 3.1). Consider any $V \in \mathcal{V}$. It follows from Lemma A.1 that

$$\mathrm{KL}(V \parallel V^{(i)'}) \leq \mathrm{KL}(V \parallel V^{(i)}) - (\log e)\epsilon \left(\frac{F(V^{(i)}, W^{(i)}) - F(V, W^{(i)})}{2k} - \epsilon \right).$$

Since $V^{(i+1)}$ is a σ-approximate KL projection of $V^{(i)'}$ on $\mathrm{Conv}(\mathcal{V})$,

$$\mathrm{KL}(V \parallel V^{(i+1)}) \leq \mathrm{KL}(V \parallel V^{(i)'}) + \sigma.$$

Therefore

$$\mathrm{KL}(V \parallel V^{(i)}) - \mathrm{KL}(V \parallel V^{(i+1)}) \geq (\log e)\epsilon \left(\frac{F(V^{(i)}, W^{(i)}) - F(V, W^{(i)})}{2k} - \epsilon \right) - \sigma.$$

Summing over $i = 1, \ldots, S$ and telescoping, we obtain

$$\begin{aligned} &\mathrm{KL}(V \parallel V^{(1)}) - \mathrm{KL}(V \parallel V^{(S+1)}) \\ &\geq (\log e)\epsilon \sum_{i=1}^{S} \left(\frac{F(V^{(i)}, W^{(i)}) - F(V, W^{(i)})}{2k} - \epsilon \right) - S\sigma \\ &= (\log e) S\epsilon \left(\frac{\mathrm{Avg}_{1 \leq i \leq S} F(V^{(i)}, W^{(i)}) - F(V, W^*)}{2k} - \epsilon \right) - S\sigma. \end{aligned}$$

Since $\mathrm{KL}(V \parallel V^{(S+1)}) \geq 0$, rearranging gives

$$\frac{\mathrm{Avg}_{1 \leq i \leq S} F(V^{(i)}, W^{(i)}) - F(V, W^*)}{2k} \leq \frac{\mathrm{KL}(V \parallel V^{(1)}) + S\sigma}{(\log e) S\epsilon} + \epsilon = O(\epsilon)$$

for $\sigma = \epsilon^2$, $S = \mathrm{KL}(V \parallel V^{(1)})/\epsilon^2$.

Next we describe an average case variant where the set $\mathcal{V}$ of strategies for Player 1 is a set of distributions of the form (X, C) where C may vary, but the marginal distribution of X is fixed. This is convenient for a number of applications (e.g. Section 5, and simple construction of pseudorandom generators from one-way functions [VZ1]) that involve distinguishers on such joint distributions (X, C).

Theorem 3.2 (Uniform Min-Max Theorem – Average Case). *Let $\mathcal{V}$ be a subset of distributions over $[N] \times [q]$ of the form (X, C) where C may vary, but the marginal distribution of X is fixed. That is, for every (X, C), $(X', C') \in \mathcal{V}$ and every $x \in [N]$ we have $\sum_c \Pr[(X, C) = (x, c)] = \sum_c \Pr[(X', C') = (x, c)]$.*

Consider a two-player zero-sum game where the sets of pure strategies for Player 1 and Player 2 are $\mathcal{V}$ and $\mathcal{W}$, and the payoff to Player 2 is defined to be $F((X, C), W) = \mathbb{E}_{X,C}[f((X, C), W)]$ for some function $f : [N] \times [q] \times \mathcal{W} \to [-k, k]$. Then for every $0 < \epsilon \leq 1$ and $S \geq \max_{(X,C) \in \mathrm{Conv}(\mathcal{V})} \mathrm{KL}(X, C \parallel X, C^{(1)})/\epsilon^2$, after S iterations Algorithm 3.2 (Finding Universal Strategy – Average Case) always outputs a mixed strategy W^* for Player 2 such that

$$\min_{(X,C) \in \mathcal{V}} F((X, C), W^*) \geq \operatorname*{Avg}_{1 \leq i \leq S} F((X, C^{(i)}), W^{(i)}) - O(k\epsilon).$$

(This holds regardless of the arbitrary choice of $W^{(i)}$ and $C^{(i+1)}$ in the algorithm.)

In particular, it suffices to take $S \geq \left(\log q - \min_{(X,C) \in \mathcal{V}} \mathrm{H}(C|X)\right)/\epsilon^2$ if we set $(X, C^{(1)}) = (X, U_{[q]}) \in \mathrm{Conv}(\mathcal{V})$ (where $U_{[q]}$ is independent of X).

Choose an initial strategy $(X, C^{(1)}) \in \mathrm{Conv}(\mathcal{V})$ for Player 1
for $i \leftarrow 1$ **to** S **do**
  Obtain an arbitrary strategy $W^{(i)} \in \mathcal{W}$ for Player 2
  Weight Update:
  Let $C^{(i)'}$ be such that $\forall x, a$,
    $\Pr[C^{(i)'} = a | X = x] \propto e^{-\epsilon \cdot f(x,a,W^{(i)})/2k} \cdot \Pr[C^{(i)} = a | X = x]$
  Projection:
  $(X, C^{(i+1)})$
    $\leftarrow$ an arbitrary ϵ^2-approx KL projection of $(X, C^{(i)'})$ on $\mathrm{Conv}(\mathcal{V})$
end
Let W^* be the mixed strategy for Player 2 uniform over $W^{(1)}, \ldots, W^{(S)}$
return W^*

Algorithm 3.2. Finding Universal Strategy – Average Case

Proof. Note that Algorithm 3.2 is the same as Algorithm 3.1, except for the difference that here we update $C^{(i)}$ instead of $V^{(i)}$. We show that the combined effect of the update and KL projection steps is identical in the two algorithms. Note that we can write $V^{(i)'}$ as $(X^{(i)'}, g_i(X^{(i)'}))$ for the randomized function g_i where $\Pr[g_i(x) = a] \propto e^{\epsilon \cdot f(x,a,W^{(i)})/2k} \cdot \Pr[C^{(i)} = a | X = x]$ for every x and a. For the same function g_i, we have $(X, g_i(X)) = (X, C^{(i)'})$. Thus, we can apply the following lemma.

Lemma 3.1. *Let X' be a distribution on $[N]$ with $\mathrm{supp}(X') \supseteq \mathrm{supp}(X')$, and let $g : [N] \to [q]$ be a randomized function. Then the KL projection of $(X', g(X'))$ on $\mathrm{Conv}(\mathcal{V})$ equals the KL projection of $(X, g(X))$ on $\mathrm{Conv}(\mathcal{V})$.*

Proof. Consider any $(X, C) \in \mathrm{Conv}(\mathcal{V})$. We have

$$
\begin{aligned}
&\mathrm{KL}(X, C \parallel X', g(X')) \\
&= \mathrm{KL}(X \parallel X') + \mathrm{KL}((C|X) \parallel (g(X')|X')) && \text{(by chain rule for KL divergence)} \\
&= \mathrm{KL}(X \parallel X') + \mathrm{KL}((C|X) \parallel (g(X)|X)) && \text{(by def of conditional KL divergence)} \\
&= \mathrm{KL}(X \parallel X') + \mathrm{KL}(X, C \parallel X, g(X)). && \text{(by chain rule for KL divergence)}
\end{aligned}
$$

Thus the KL projections are the same.

4 Application: Uniform Hardcore Theorem

A fundamental result in complexity theory is Impagliazzo's Hardcore Theorem [Imp], which, in the strengthened version due to Klivans and Servedio [KS] and Holenstein [Hol1], says that every function $f : \{0,1\}^n \to \{0,1\}$ that is δ-hard for poly-sized boolean circuits (that is, every poly-sized circuit fails to compute f on at least δ fraction of inputs) must be *extremely* hard on a subset of inputs of density at least 2δ (the *hardcore set*) (and may be easy elsewhere). In this

section, we provide a simplified proof of a hardcore theorem with optimal parameters, where hardness is defined with respect to *uniform* algorithms rather than boolean circuits. Following [Imp], we will deal with hardcore distributions instead of hardcore sets, which are equivalent up to a negligible additive difference in density, where density of a distribution is defined as follows:

Definition 4.1 (Density of distribution). *Let X and Y be distributions over* some finite set Σ. *We say X is* δ-dense in Y *if* $\Pr[Y = x] \geq \delta \cdot \Pr[X = x]$ *for all* $x \in \Sigma$. We say X is δ-dense if it is δ-dense in U_Σ (equivalently, having min-entropy at least $\log|\Sigma| - \log(1/\delta)$). *We denote by $\mathcal{C}_{m,\delta}$ the set of* all δ-dense *distributions* on $\{0,1\}^m$.

The (nonuniform) hardcore theorem with optimal hardcore density 2δ and optimal complexity blow-up $O(\log(1/\delta)/\epsilon^2)$, is due to [KS] using techniques from boosting, and an idea of iteratively increasing hardcore size due to Wigderson. The theorem can be stated as follows:

Theorem 4.1 (Hardcore Theorem [KS]). *Let (X, B)[1] be a joint distribution over $\{0,1\}^n \times \{0,1\}$ and $\epsilon > 0$. Let B be (t, δ)-hard given X, i.e. for every size t circuit P it holds that* $\Pr[P(X) = B] \leq 1 - \delta$. *Then there is a joint distribution $(\hat{X}, \hat{B})$ that is 2δ-dense in (X, B), such that for every size $t' = t/O(\log(1/\delta)/\epsilon^2)$ circuit A it holds that* $\Pr[A(\hat{X}) = \hat{B}] \leq (1 + \epsilon)/2$.

The original paper of Impagliazzo [Imp] contains both a non-trivial constructive proof, as well as a much simpler, yet non-constructive proof due to Nisan that uses the Min-Max Theorem. Nisan's proof has an appealing simplicity: Assume for contradiction that there is no hardcore distribution of high density. Then, by the Min-Max Theorem there is a *universal* predictor A^* such that for every $(\hat{X}, \hat{B})$ that is dense in (X, B) it holds that $\Pr\left[A^*(\hat{X}) = \hat{B}\right] > (1 + \epsilon)/2$. A^* is a distribution over circuits of size t, and its prediction probability is taken over this distribution as well as $(\hat{X}, \hat{B})$. By subsampling we can assume that A^* is uniform over a multiset of $S = O(\log(1/\epsilon\delta)/\epsilon^2)$ circuits of size t, while changing the advantage ϵ by at most a constant fraction. Given the universal predictor A^*, one can build a good predictor for B, contradicting the hardness of B given X, as formalized in Lemma 4.1:

Lemma 4.1 (From universal circuit to predictor [Imp]). *Let (X, B) be a joint distribution on $\{0,1\}^n \times \{0,1\}$. Let A^* be the uniform distribution over a multiset of S circuits of size t. Suppose for every joint distribution $(\hat{X}, \hat{B})$ that is δ-dense in (X, B) it holds that* $\Pr\left[A^*(\hat{X}) = \hat{B}\right] > (1 + \epsilon)/2$. *Then there is a circuit P of size $O(S \cdot t)$ such that* $\Pr[P(X) = B] > 1 - \delta$.

Specifically, we can let $P(x) = \text{majority}\{A(x) : A \in A^*\}$. Equivalently, $P(x)$ outputs 1 with probability

$$\frac{1}{2}\left(1 + \text{sign}\left(\Pr[A^*(x) = 1] - \frac{1}{2}\right)\right).$$

[1] The version we state is a slight generalization of the version in [KS], which only allows B to be a deterministic boolean function of X. However, the more general version follows readily from almost the same proof.

Unfortunately, both proofs in [Imp] yield a non-optimal hardcore density of δ. Following Nisan's proof using Min-Max Theorem, Holenstein [Hol1] proves the hardcore theorem with optimal hardcore density of 2δ (Theorem 4.1), by strengthening the above lemma to Lemma 4.2 below (using a trick from Levin's proof of the XOR Lemma).

Lemma 4.2 (From universal circuit to *optimal* predictor [Hol1]). *Let (X, B) be a joint distribution on $\{0,1\}^n \times \{0,1\}$. Let A^* be the uniform distribution over a multiset of S circuits of size t. Suppose for every joint distribution $(\hat{X}, \hat{B})$ that is 2δ-dense in (X, B) it holds that $\Pr\left[A^*(\hat{X}) = \hat{B}\right] > (1+\epsilon)/2$. Then there is a circuit P of size $O(S \cdot t)$ such that $\Pr[P(X) = B] > 1 - (1-\epsilon)\,\delta$.*

Specifically, we can let $P(x)$ output 1 with probability $p(x)$ truncated at 0 and 1 (i.e. $\min\{\max\{p(x), 0\}, 1\}$), for

$$p(x) = \frac{1}{2}\left(1 + \frac{\Pr[A^*(x) = 1] - \frac{1}{2}}{\phi}\right)$$

where ϕ is the least number s.t. $\Pr_{X,B}\left[\Pr_{A^*}\left[A^*(X) = B\right] \leq 1/2 + \phi\right] \geq 2\delta$. (WLOG ϕ is a multiple of $1/S$.)

The drawback of these proofs based on the standard Min-Max Theorem is that they are non-constructive, and that the complexity blow-up is non-optimal (with non-optimal settings of S due to probabilistic construction of the multiset).

A constructive proof such as the one by Impagliazzo [Imp] can be interpreted as a hardcore theorem for the *uniform* setting of hardness, where the hardness is with respect to efficient algorithms rather than small circuits. (See Theorem 4.2 below for the exact formulation). This *Uniform Hardcore Theorem* is needed for several important applications ([KS,Hol1,Hol2,HHR,HRV]). Building on the constructive proof in [Imp], Holenstein [Hol1] also shows a *uniform* hardcore theorem with optimal hardcore density, but is rather involved and fails to achieve the optimal complexity blow-up $O(\log(1/\delta)/\epsilon^2)$. Subsequently, Barak, Hardt, and Kale ([BHK]) gave an alternative proof of uniform hardcore theorem achieving optimal complexity blow-up of $O(\log(1/\delta)/\epsilon^2)$ (but without optimal hardcore density), based on ideas of multiplicative weights and Bregman projection.

As an application of the Uniform Min-Max Theorem (which itself is inspired by [BHK]), we offer a new proof of the Uniform Hardcore Theorem. Essentially, our proof simply replaces the use of Min-Max Theorem in Holenstein's proof (of the non-uniform hardcore theorem, Theorem 4.1) with the Uniform Min-Max Theorem. Consequently it has the advantages of (i) optimal hardcore density 2δ; (ii) optimal complexity blow-up $O(\log(1/\delta)/\epsilon^2)$; (iii) being more modular (e.g. compared to [BHK]) and simpler (e.g. compared to Holenstein's uniform proof [Hol1]).

Notation. For a distribution Z, let O_Z denote the oracle that gives a random sample from Z when queried.

Theorem 4.2 (Uniform Hardcore Theorem). *Let n be a security parameter, $m = m(n) = \text{poly}(n)$, $\delta = \delta(n)$, $\epsilon' = \epsilon'(n)$, $q = q(n)$ all computable in*

$\mathrm{poly}(n)$ *time, and* $(X, B) = g(U_m)$ *be a joint distribution where* $g : \{0,1\}^m \to \{0,1\}^n \times \{0,1\}$ *is computable in* $\mathrm{poly}(n)$ *time. Suppose that* (X, B) *has no hard-core distribution of density at least* 2δ*, i.e. there is a time* t *oracle algorithm* A *and infinitely many* n*, such that for every* $C \in \mathcal{C}_{m,2\delta}$*,*

$$\Pr_{(x,b)\leftarrow g(C)}\left[A^{O_C}(x) = b\right] > \frac{1}{2} + \epsilon'.$$

Then there is a time $\mathrm{poly}(t, n, 1/\delta, 1/\epsilon')$ *randomized algorithm* P *such that for infinitely many* n*,*

$$\Pr[P(X) = B] > 1 - \delta.$$

Moreover, P *is computable with* $O(\log(1/\delta)/\epsilon'^2)$ *oracle queries to* A*.*

Proof (Sketch). (See the full version [VZ2] for a complete proof). We will apply Theorem 3.1 (Uniform Min-Max Theorem), with

- $\mathcal{V} = \mathcal{C}_{m,2\delta}$;
- $\mathcal{W} = \{$(deterministic) circuits of size $tm + \mathrm{poly}(t)\}$;
- $f(z, W) = I(W(x) = b)$, where $(x, b) = g(z)$ and $I(\cdot)$ is the indicator function.

This corresponds to the two-player zero-sum game where Player 1 chooses some distribution $C \in \mathcal{C}_{m,2\delta}$, and Player 2 chooses a $tm+\mathrm{poly}(t)$ sized circuit W, with expected payoff $F(C, W) = \mathbb{E}[f(C, W)] = \Pr_{(x,b)\leftarrow g(C)}[W(x) = b]$ for Player 2. It turns out that using A, Algorithm 3.1 (Finding Universal Strategy) with KL projection on the set $\mathcal{V} = \mathcal{C}_{m,2\delta}$ can be implemented efficiently, such that for infinitely many n, in each iteration we obtain (from running A) some W with good prediction probability. This gives us an efficient universal predictor A^* of B given X, by the Uniform Min-Max Theorem. From the universal predictor, we then obtain a $(1 - \delta)$-predictor of B using Lemma 4.2, by searching for the correct ϕ.

5 Application: Impossibility of Black-Box Construction of Succinct Non-interactive Argument

A result of Gentry and Wichs [GW] shows that there is no black-box construction of succinct non-interactive arguments (SNARGs) from any natural cryptographic assumption (formally, they consider *falsifiable* cryptographic assumptions: ones that are defined by a polynomial-time security game). Their result relies on the (mild) assumption that there exist *hard subset membership problems*, which is equivalent to the existence of subexponentially hard one-way functions. One limitation is that they need to work in the non-uniform setting, in part due to their use of the Min-Max Theorem (in [GW] Lemma 3.1). In this section we show how to obtain the analogous result in the *uniform setting* by using the Uniform Min-Max Theorem. More specifically, assuming that there exist subexponentially hard one-way functions that are secure against uniform algorithms,

we show that there is no black-box construction of SNARGs based on cryptographic assumptions where security is measured against uniform algorithms (unless the assumption is already false).

A succinct non-interactive argument (SNARG) is a non-interactive argument system where the proof size is bounded by a fixed polynomial, for all instances and witnesses whose size can be an arbitrarily large polynomial. Formally,

Definition 5.1 (SNARG). *Let L be an* **NP** language associated with relation R. *We say that a tuple (G, P, V) of probabilistic polynomial-time (PPT) algorithms is a* succinct non-interactive argument for R *if the following properties hold:*

- ***Completeness:*** *For all $(x, w) \in R$, if we choose $(\mathsf{CRS}, \mathsf{PRIV}) \leftarrow G(1^n), \Pi \leftarrow P(\mathsf{CRS}, x, w)$,* then

$$\Pr\left[V\left(\mathsf{PRIV}, x, \Pi\right) = 0\right] = \mathrm{negl}(n).$$

- ***Soundness:*** *For every PPT algorithm (efficient* adversary*) A, if we choose $(\mathsf{CRS}, \mathsf{PRIV}) \leftarrow G(1^n), (X, \Pi) \leftarrow A(1^n, \mathsf{CRS})$,* then

$$\Pr\left[V\left(\mathsf{PRIV}, X, \Pi\right) = 1 \wedge X \notin L\right] = \mathrm{negl}(n).$$

- ***Succinctness:*** *For all $(x, w) \in \mathrm{supp}(X, W)$ and $\mathsf{crs} \in \mathrm{supp}(\mathsf{CRS})$, the length of the proof $\pi = P(\mathsf{crs}, x, w)$ is $|\pi| = \mathrm{poly}(n)(|x| + |w|)^{o(1)}$. We also consider a weaker variant called* slightly succinct*, where we require the length of a proof to be $|\pi| = \mathrm{poly}(n)(|x| + |w|)^{\alpha} + o(|x| + |w|)$ for some constant $\alpha < 1$.*[2]

Our notion of a falsifiable cryptographic assumption is analogous to [GW], except that the adversary A is a *uniform* algorithm instead of circuit:

Definition 5.2 (Falsifiable assumption). *Given an interactive PPT algorithm* Chal *(the challenger), the* uniform falsifiable (cryptographic) assumption (associated with) Chal *states that for all (uniform) PPT algorithms H, the probability that* $\mathrm{Chal}(1^n)$ *outputs a special symbol* win *after interacting with $H(1^n)$ is at most* $\mathrm{negl}(n)$ *for all sufficiently large n.*

For any randomized (possibly inefficient) function H, we let $\mathrm{Break}_H(n)$ *denote the above probability and say that H* breaks the assumption *if* $\mathrm{Break}_H(n) \geq 1/\mathrm{poly}(n)$ *for infinitely many n.*

Remark 5.1. An alternative definition of falsifiable assumption allows specifying a constant β, and says that the probability $\mathrm{Chal}(1^n)$ outputs win is at most $\beta + \mathrm{negl}(n)$. However, it turns out that setting $\beta = 0$, i.e. our definition above, is without loss of generality [HH]. We adopt the simpler definition because it is convenient for our proof.

[2] Earlier versions of [GW] contained a minor bug in the definition of slight succinctness. We use the corrected definition from the current version of their paper.

Next we define black-box reductions:

Definition 5.3 (Adversary and reduction). *For a randomized function A and a constant $c \in \mathbb{N}$, we say (A, c) is a* (G, P, V)-adversary *if $|A(1^n, \mathsf{crs})| \leq n^c$ and A violates the soundness condition infinitely often, i.e. if we choose $(\mathsf{CRS}, \mathsf{PRIV}) \leftarrow G(1^n), (X, \Pi) \leftarrow A(1^n, \mathsf{CRS})$, then*

$$\Pr\left[V\left(\mathsf{PRIV}, X, \Pi\right) = 1 \wedge X \notin L\right] \geq n^{-c}$$

for infinitely many n. We say (A, c) is an a.e. (G, P, V)-adversary *if A violates soundness for all sufficiently large n.*

A uniform black-box reduction showing the soundness of (G, P, V) based on a falsifiable assumption Chal *is a family of (uniform) probabilistic oracle algorithms $\{\mathrm{Red}_c\}$ (one for each $c \in \mathbb{N}$) such that for every (G, P, V)-adversary (A, c), $\mathrm{Red}_c^A(1^n)$ breaks the assumption and runs in time $\mathrm{poly}_c(n)$ (i.e. a polynomial that depends on c).*

For a probabilistic oracle algorithm Red*, we say a query $(1^m, \mathsf{crs})$ of $\mathrm{Red}(1^n)$ has* length m*. In general, $\mathrm{Red}(1^n)$ may make queries of various lengths. We say* Red *is* length-mapping *if for all n, all queries of $\mathrm{Red}(1^n)$ are of the same length $m = m(n)$;* denote this m by $\mathrm{query}_{\mathrm{Red}}(n)$*. Most reductions in cryptography set $m = n$ i.e. preserve length; that is, the security parameter of (G, P, V) is equal to that of the assumption.*

Following [GW], our results assume the existence of *hard subset membership problem.*

Definition 5.4 (Uniformly hard subset membership problem). *Let n be a security parameter, L be an* **NP** *language associated with relation R. We say $((X, W), U)$ is a* subset membership problem for R *if $(X, W) = (X, W)(n)$ is a $\mathrm{poly}(n)$-time samplable joint distribution whose support lies in R, and $U = U(n)$ a $\mathrm{poly}(n)$-time samplable distribution with $\Pr[U \notin L] \geq n^{-O(1)}$.*

A subset membership problem $((X, W), U)$ is a subexponentially hard *if X and U are $(2^{\Omega(n^\delta)}, 2^{-\Omega(n^\delta)})$-indistinguishable for a constant $\delta > 0$. We say it is* exponentially hard *if the above occurs and $|x| + |w| = O(n^\delta)$ for every $(x, w) \in \mathrm{supp}(X, W)$.*

This is a relatively mild assumption; for subexponentially hard subset membership problems, their existence is equivalent to the existence of subexponentially hard one-way functions.

Remark 5.2. Our definition of a hard subset membership problem is a variant of [GW] that is needed in the uniform setting, but also can be used in the nonuniform setting of [GW]. In [GW], they require that X is indistinguishable from a (not necessarily samplable) distribution U whose support is disjoint from L, whereas we require that U is samplable and allow it to hit L with negligible probability.

We now state the uniform analogue of the main result of [GW]. Compared to [GW], our Theorem 5.1 makes the weaker assumption of subexponentially hard

subset membership problem with respect to *uniform* algorithms, with the conclusion that a *uniform* falsifiable assumption cannot be broken also being weaker (unless the assumption is false).

Theorem 5.1 (Main theorem). *Let L be an* **NP** *language associated with relation R that has* a *subexponentially hard subset membership problem, and (G, P, V) be an non-interactive proof system for R that satisfies the completeness and succinctness properties. Then for every uniform falsifiable assumption* Chal*, one of the following must hold:*

- *The assumption* Chal *is false, or*
- *There is no uniform black-box reduction showing the soundness of (G, P, V) based on* Chal*.*

The same conclusion also holds if we assume an exponentially *hard subset membership problem, and (G, P, V) is only* slightly *succinct.*

To prove it in the nonuniform setting, the main idea of [GW] is showing that any SNARG (G, P, V) has an inefficient adversary A that can be (efficiently) "simulated" i.e. there exists an efficient algorithm Sim (the simulator) such that $\mathrm{Red}^A(1^n) \approx \mathrm{Red}^{\mathsf{Sim}}(1^n)$ for all PPT oracle algorithms Red (cf. [GW] Lemma 4.1). Thus, if there were a black-box reduction Red showing the soundness of (G, P, V) based on a falsifiable assumption, then Red^A would break the falsifiable assumption (since A is an adversary) and so would $\mathrm{Red}^{\mathsf{Sim}}$ (since $\mathrm{Red}^A(1^n) \approx \mathrm{Red}^{\mathsf{Sim}}(1^n)$). In other words, the assumption would be false.

To prove it in the uniform setting, we do the same showing that there is an adversary (A, c) that can be simulated by a *uniform* algorithm Sim, with several necessary tweaks:

Lemma 5.1 (Existence of simulatable adversary). *Let L be an* **NP** *language associated with relation R that has* a *subexponentially hard subset membership problem $((X, W), U)$, and (G, P, V) be a non-interactive proof system for R that satisfies the completeness and succinctness properties. Let n be a security parameter, $(\mathsf{PRIV}, \mathsf{CRS}) = G(1^n)$, $((X, W), U) = ((X, W), U)(n)$,* and *$\Pi = P(\mathsf{CRS}, X, W)$. Let $\ell = \ell(n) \geq n$ be a polynomial bound on the running time of $G(1^n)$ as well as the proof size $|\Pi|$, and c be a constant such that $|X| + |\Pi| \leq n^c$.*

Then for every length-mapping PPT oracle algorithm Red *such that $\mathrm{query}_{\mathrm{Red}}(k) = \omega(1)$, there is a PPT algorithm* Sim *and randomized function A satisfying:*

- *(A, c) is an a.e. (G, P, V)-adversary; and*
- Sim *simulates A: For all sufficiently large k, w.p. at least $1/\mathrm{poly}(k)$, $\mathsf{Sim}(1^k)$ outputs a randomized circuit B such that*

$$\mathrm{Break}_{\mathrm{Red}^A}(k) - \mathrm{Break}_{\mathrm{Red}^B}(k) = \mathrm{negl}(k).$$

(WLOG B only takes inputs $(1^n, \cdot)$ where $n = \mathrm{query}_{\mathrm{Red}}(k)$.)

The same conclusion also holds if we assume an exponentially *hard subset membership problem, and that* (G, P, V) *is only* slightly *succinct.*

Note that Lemma 5.1 is only stated for length-mapping reductions (unlike [GW]). We remove this restriction in the full version [VZ2] where we prove the main theorem (for which we use the fact that the simulatable adversary (A, c) is an a.e. adversary).

We defer the complete proof of Lemma 5.1 to the full version [VZ2], and offer an overview below.

Overview of Proof of Lemma 5.1. The proof is set up as follows. Given a subexponentially hard subset membership problem $((X, W), U)$, we can WLOG assume that X and U are $(2^{d\ell}, 2^{-d\ell})$-indistinguishable for a sufficiently large constant d, where $\ell = \ell(n)$ is a bound on the length of the proof output by $P(\mathsf{crs}, x, w)$ for $(x, w) \in \mathrm{supp}(X, W)$ and $\mathsf{crs} \in \mathrm{supp}(\mathsf{CRS})$. (If X and U are only $(2^{n^\delta}, 2^{-n^\delta})$-indistinguishable for some $\delta > 0$, we simply re-index, replacing $X(n)$ with $X((d\ell)^{1/\delta})$.) If $((X, W), U)$ is exponentially hard, we can also ensure that X and U are $(2^{d\ell}, 2^{-d\ell})$-indistinguishable by re-indexing so that $\ell \leq \mathrm{poly}(n) \cdot (|x|+|w|)^\alpha + o(|x|+|w|) = O(|x|+|w|)/d$ for all $(x, w) \in \mathrm{supp}(X, W)$ and $\mathsf{crs} \in \mathrm{supp}(\mathsf{CRS})$.

Consider the joint distribution (CRS, X, Π) where $\mathsf{CRS} = \mathsf{CRS}(n)$ is the distribution of common reference string, and $\Pi = \Pi(n)$ is the ℓ-bit proof produced by P for the instance/witness pair (X, W). Using the fact that Π is short (by succinctness), it turns out that the $2^{-d\ell}$-indistinguishability of X and U — and hence of (CRS, X) and (CRS, U), by samplability of CRS — implies there is no universal distinguisher D^* (as a $2^{O(\ell)}$ time algorithm) that $2^{-O(\ell)}$-distinguishes (CRS, X, Π) from all (CRS, U, Π'), where Π' is an ℓ-bit string arbitrarily jointly distributed with (CRS, U). This is extracted from the proof of a technical lemma of Gentry and Wichs ([GW] Lemma 3.1) and doesn't require the use of the Min-Max Theorem.

We consider the two-player zero-sum game where Player 1 selects a distribution Π' on $\{0,1\}^\ell$ jointly distributed with (CRS, U), then Player 2 selects a small circuit D and receives (expected) payoff $\mathbb{E}[D(\mathsf{CRS}, X, \Pi)] - \mathbb{E}[D(\mathsf{CRS}, U, \Pi')]$. Recall that the Uniform Min-Max Theorem – Average Case (Theorem 3.2) builds a sequence — which we denote by List_n — of Π' jointly distributed with (CRS, U), and says that if for each $\Pi' \in \mathsf{List}_n$ we can obtain a $2^{-O(\ell)}$-distinguisher D between (CRS, X, Π) and (CRS, U, Π') e.g. by some $2^{O(\ell)}$ time algorithm $\mathsf{FindDist}$, then we can obtain a universal $2^{-O(\ell)}$-distinguisher D^* (as a $2^{-O(\ell)}$ time algorithm) for *all* possible Π'. Since such D^* cannot exist (by previous discussion), it must be that for every $2^{O(\ell)}$ time algorithm $\mathsf{FindDist}$ there is some $\Pi' \in \mathsf{List}_n$ for which $\mathsf{FindDist}$ fails to produce a $2^{-O(\ell)}$-distinguisher D. Note that List_n actually depends on $\mathsf{FindDist}$ (indeed it is obtained by running Algorithm 3.2 using $\mathsf{FindDist}$ to select the actions for Player 2).

We will use $\mathsf{FindDist}$ to construct the simulatable adversary A as follows. Consider any $\Pi' \in \mathsf{List}_n$ for which $\mathsf{FindDist}$ fails to produce a $2^{-O(\ell)}$-distinguisher. We let A be the randomized function such that $A(1^n, \mathsf{CRS}) = (U, \Pi')$. For an appropriate choice of $\mathsf{FindDist}$, such an A will always be an a.e. adversary. Indeed,

if A is not an a.e. adversary then (PRIV, U, Π') does not pass the soundness test, whereas (PRIV, X, Π) passes the completeness test, hence we can use the verifier V to construct a distinguisher between (CRS, X, Π) and (CRS, U, Π'). Choosing $\mathsf{FindDist}$ to produce this distinguisher yields an a.e. adversary A.

Thus we only need to argue that, for an appropriate choice of $\mathsf{FindDist}$, A is also simulatable. Our simulation is the algorithm S such that $S(1^n, \mathsf{CRS}) = (X, \Pi)$. If we appropriately construct $\mathsf{FindDist}$ from the reduction Red and challenger Chal, then we can show that

$$\mathrm{Break}_{\mathrm{Red}^A}(k) - \mathrm{Break}_{\mathrm{Red}^S}(k) \leq 1/\mathrm{poly}(k) \cdot 2^{-O(\ell)},$$

where $\ell = \ell(n)$ for $n = \mathrm{query}_{\mathrm{Red}}(k)$. (Otherwise, we could use Red and Chal to construct a $2^{-O(\ell)}$-distinguisher between $(\mathsf{CRS}, A(1^n, \mathsf{CRS})) = (\mathsf{CRS}, X, \Pi)$ and $(\mathsf{CRS}, S(1^n, \mathsf{CRS})) = (\mathsf{CRS}, U, \Pi')$.) This completes the proof provided that $2^{-O(\ell)} \leq 1/\mathrm{poly}(k)$, which follows if Red does not make queries that are too short. If instead $2^{-O(\ell)} > 1/\mathrm{poly}(k)$, then we construct a simulator for A differently — by simply outputting a random element of List_n, which will equal A and be a perfect simulator w.p. $1/|\mathsf{List}_n| = 1/2^{O(\ell)} \geq 1/\mathrm{poly}(k)$. (Gentry and Wichs [GW] handle short queries using nonuniformity, by hardcoding all the answers.)

Acknowledgments. We thank Kai-Min Chung for many helpful discussions, especially on SNARGs.

References

BHK. Barak, B., Hardt, M., Kale, S.: The uniform hardcore lemma via approximate bregman projections. In: SODA 2009: Proceedings of the Nineteenth Annual ACM -SIAM Symposium on Discrete Algorithms, Philadelphia, PA, USA, pp. 1193–1200. Society for Industrial and Applied Mathematics (2009)

BSW. Barak, B., Shaltiel, R., Wigderson, A.: Computational analogues of entropy. In: Arora, S., Jansen, K., Rolim, J.D.P., Sahai, A. (eds.) RANDOM 2003 and APPROX 2003. LNCS, vol. 2764, pp. 200–215. Springer, Heidelberg (2003)

CLP. Chung, K.-M., Lui, E., Pass, R.: From weak to strong zero knowledge using a new non-black-box simulation technique (unpublished manuscript)

CT. Cover, T.M., Thomas, J.A.: Elements of information theory, 2nd edn. Wiley (2006)

DP. Dziembowski, S., Pietrzak, K.: Leakage-resilient cryptography. In: FOCS, pp. 293–302. IEEE Computer Society (2008)

FK. Frieze, A., Kannan, R.: Quick approximation to matrices and applications. Combinatorica 19(2), 175–220 (1999)

FR. Fuller, B., Reyzin, L.: Computational entropy and information leakage (2011), `http://www.cs.bu.edu/fac/reyzin`

FS. Freund, Y., Schapire, R.E.: Adaptive game playing using multiplicative weights. Games and Economic Behavior 29, 79–103 (1999)

GT. Green, B., Tao, T.: The primes contain arbitrarily long arithmetic progressions. Ann. of Math. 167(2), 481–547 (2008)

GW. Gentry, C., Wichs, D.: Separating succinct non-interactive arguments from all falsifiable assumptions. In: Fortnow, L., Vadhan, S.P. (eds.) STOC, pp. 99–108. ACM (2011)

HH. Haitner, I., Holenstein, T.: On the (Im)Possibility of key dependent encryption. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 202–219. Springer, Heidelberg (2009)
HHR. Haitner, I., Harnik, D., Reingold, O.: Efficient pseudorandom generators from exponentially hard one-way functions. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 228–239. Springer, Heidelberg (2006)
Hol1. Holenstein, T.: Key agreement from weak bit agreement. In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC), pp. 664–673 (2005)
Hol2. Holenstein, T.: Pseudorandom generators from one-way functions: A simple construction for any hardness. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 443–461. Springer, Heidelberg (2006)
HRV. Haitner, I., Reingold, O., Vadhan, S.: Efficiency improvements in constructing pseudorandom generators from one-way functions. In: Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC), pp. 437–446 (2010)
HW. Herbster, M., Warmuth, M.: Tracking the best linear predictor. Journal of Machine Learning Research 1, 281–309 (2001)
Imp. Impagliazzo, R.: Hard-core distributions for somewhat hard problems. In: Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS), pp. 538–545 (1995)
KS. Klivans, A.R., Servedio, R.A.: Boosting and hard-core set construction. Machine Learning 51(3), 217–238 (2003)
PJ. Pietrzak, K., Jetchev, D.: How to fake auxiliary input. In: ICITS 2012 Invited Talk (2012)
RTTV. Reingold, O., Trevisan, L., Tulsiani, M., Vadhan, S.: Dense subsets of pseudorandom sets. In: Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2008), October 26-28, pp. 76–85. IEEE (2008)
TTV. Trevisan, L., Tulsiani, M., Vadhan, S.: Regularity, boosting, and efficiently simulating every high-entropy distribution. In: Proceedings of the 24th Annual IEEE Conference on Computational Complexity (CCC 2009), July 15-18, pp. 126–136 (2009); Preliminary version posted as ECCC TR08-103
TZ. Tao, T., Ziegler, T.: The primes contain arbitrarily long polynomial progressions. Acta Math. 201(2), 213–305 (2008)
VZ1. Vadhan, S., Zheng, C.J.: Characterizing pseudoentropy and simplifying pseudorandom generator constructions. In: Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC 2012), May 19-22, pp. 817–836 (2012)
VZ2. Vadhan, S.P., Zheng, C.J.: A uniform min-max theorem with applications in cryptography. To appear on the Cryptology ePrint Archive (in preparation, 2013)

A Omitted Lemmas

Lemma A.1 (Multiplicative weight update decreases KL). *Let A, B be distributions over $[N]$ and $f : [N] \to [0,1]$ any function. Define random variable A' such that*

$$\Pr[A' = x] \propto e^{\epsilon \cdot f(x)} \Pr[A = x]$$

for $0 \leq \epsilon \leq 1$. Then $\mathrm{KL}(B \parallel A') \leq \mathrm{KL}(B \parallel A) - (\log e)\epsilon\,(\mathbb{E}[f(B)] - \mathbb{E}[f(A)] - \epsilon)$.

Proof. See the full version [VZ2].

Limits of Provable Security for Homomorphic Encryption

Andrej Bogdanov[1,⋆] and Chin Ho Lee[2]

[1] Dept. of Computer Science and Engineering and Institute for Theoretical Computer Science and Communications, Chinese University of Hong Kong
andrejb@cse.cuhk.edu.hk
[2] Dept. of Computer Science and Engineering, Chinese University of Hong Kong
chlee@cse.cuhk.edu.hk

Abstract. We show that public-key bit encryption schemes which support weak (i.e., compact) homomorphic evaluation of any sufficiently "sensitive" collection of functions cannot be proved message indistinguishable beyond AM ∩ coAM via general (adaptive) reductions, and beyond statistical zero-knowledge via reductions of constant query complexity. Examples of sensitive collections include parities, majorities, and the class consisting of all AND and OR functions.

We also give a method for converting a strong (i.e., distribution-preserving) homomorphic evaluator for essentially any boolean function (except the trivial ones, the NOT function, and the AND and OR functions) into a rerandomization algorithm: This is a procedure that converts a ciphertext into another ciphertext which is statistically close to being independent and identically distributed with the original one. Our transformation preserves negligible statistical error.

1 Introduction

In this work we revisit the question of basing cryptography on NP-hardness. If P equals NP then computationally secure encryption is impossible. Is the converse true? Despite considerable efforts, there is no candidate encryption scheme whose security can be plausibly reduced to the worst-case hardness of some NP-complete problem. Neither is there conclusive evidence that rules out constructions of secure encryption schemes from NP-complete problems, although several obstacles have been pointed out over the years.

Restricting the encryption. Brassard [Bra79] shows that no public-key encryption scheme can be proved secure beyond NP ∩ coNP, but under the implicit assumption that every public key-ciphertext pair (queried by the reduction) can be decrypted uniquely. Goldreich and Goldwasser [GG98] argue that this assumption is unrealistic by giving examples of encryption schemes that do not satisfy it. They show that the conclusion holds under the relaxed assumption

⋆ Work supported by grants RGC GRF CUHK410111 and CUHK410112.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 111–128, 2013.

that invalid queries to the decryption oracle can be efficiently certified as such. (If the reduction is randomized, the limitation weakens to $\mathrm{AM} \cap \mathrm{coAM}$.)

Goldreich and Goldwasser warn that these assumptions are unrealistic as they do not apply to many known proofs of security. Bogdanov and Trevisan [BT06] point out the following example of Even and Yacobi [EY80]. They construct a public key encryption scheme and show how to solve an NP-hard problem using a distinguishing oracle. Their notion of security is unrealistic, as they require a perfect distinguishing oracle. However, their example illustrates that the restrictions imposed by Brassard and Goldreich and Goldwasser do not capture the difficulty of basing cryptography on NP hardness.

Akavia, Goldreich, Goldwasser, and Moshkovitz [AGGM06] rule out reductions from NP-complete problems to inverting one-way functions (the basis of private-key encryption) assuming that sizes of preimage sets are worst-case certifiable in NP. The same considerations apply to their argument. There are natural examples of conjectured one-way functions (for example, Goldreich's function [Gol00]) not known to satisfy the aforementioned assumptions.

Restricting the reduction. Another line of works makes restrictive assumptions about the type of reduction used to prove NP-hardness. Feigenbaum and Fortnow [FF93] show that a decision problem cannot be proven NP-hard on average (unless the polynomial hierarchy collapses) by a reduction that is non-adaptive and each of its queries is uniformly distributed. Bogdanov and Trevisan [BT06] obtain the same conclusion without restricting the distribution of queries, but still under non-adaptive reductions. More precisely, they show that if there is a non-adaptive reduction from a decision problem L to a problem in distributional NP, then L must be in $\mathrm{AM/poly} \cap \mathrm{coAM/poly}$. In particular their result applies to the problem of inverting a one-way function. For this important case, Akavia et al. improve the limitation to $\mathrm{AM} \cap \mathrm{coAM}$, also assuming the reduction is non-adaptive.

Haitner, Mahmoody, and Xiao [HMX10] show that collision resistant hash functions and statistically hiding commitments cannot be proved secure beyond $\mathrm{AM} \cap \mathrm{coAM}$ via reductions that make a constant number of rounds of calls to the adversary.

Lattice-based cryptography provides examples of encryption schemes whose insecurity would imply worst-case solutions to conjectured hard problems, like finding short vectors in lattices [Ajt96]. The reduction of Regev [Reg09], which gives the most efficient cryptosystems of this kind with a proof of security (against quantum algorithms), is adaptive. For certain settings of parameters, these cryptosystems support homomorphic evaluation of a bounded class of functionalities (and general functionalities under additional security assumptions) [Gen09, vDGHV10, BV11].

Our Results

We say a public-key encryption scheme supports weak (i.e. compact) homomorphic evaluation of a function $f\colon \{0,1\}^* \to \{0,1\}$ if for every n and $x_1 \dots x_n \in$

$\{0,1\}^n$ takes as inputs the public key and encryptions of the bits $x_1, \ldots, x_n$ and produces an output of length polynomially bounded in the security parameter that decrypts to $f(x_1 \ldots x_n)$. See Section 2 for a formal definition.

Our main theorem (Theorem 1) shows that any public key encryption scheme that supports efficient weak homomorphic evaluation of any sufficiently "sensitive" collection of functions cannot be proved message indistinguishable beyond AM $\cap$ coAM, even under adaptive reductions. Examples of such functions are parities, majorities, and the collection of all AND and OR functions.

Examples of encryption schemes that our result applies to include El Gamal encryption [Gam85], Paillier encryption [Pai99], as well as the more recent somewhat and fully homomorphic encryption schemes of Gentry [Gen09], Van Dijk et al. [vDGHV10], and Brakerski and Vaikuntanathan [BV11] (which build upon the lattice-based cryptosystems of Regev [Reg09] and Peikert [Pei09]).

In Theorem 2 we show that if the reduction has constant query complexity, then message indistinguishability cannot be proved beyond statistical zero knowledge (SZK), which is a subclass of AM $\cap$ coAM.

The reductions we consider are randomized and meet the following definition: Given an input, the reduction makes arbitrary (adaptive) queries to a distinguishing oracle for bit encryptions. We require that for any (not necessarily efficient) distinguishing oracle, which may depend on the input to the reduction, the reduction outputs the correct answer. We do not know of any cryptographic reductions that treat the adversary as a black box which fall outside our definition.

Lemma 5, which is used in the proofs of Theorems 1 and 2, gives a way to obtain rerandomization of ciphertexts from any homomorphic evaluator for the function of interest. While rerandomization has been used in constructions of homomorphic evaluators [Gen09, vDGHV10], it is not a priori clear that it is necessary for homomorphic evaluation. Homomorphic evaluation may be implemented deterministically while rerandomization requires randomness.

The statistical error of the rerandomization in Lemma 5 is noticeable. While this is sufficient for our main application, a negligible error would be desirable for most applications of rerandomization in cryptography. In Theorem 3 we show a transformation of a strong homomorphic evaluator for almost any function into a rerandomization that preserves negligible statistical error. Essentially the only exceptions to which our result does not apply are that AND, OR, and NOT functions.

Our Proof

From homomorphic evaluation to rerandomization (Section 4). To begin with let's assume that we have a *strong* (i.e., distribution-preserving) homomorphic evaluator H for the majority function maj_n on n inputs. This is an algorithm that takes as inputs independent encryptions of $x_1, \ldots, x_n$ and outputs a ciphertext which is statistically close to an encryption of $maj_n(x_1, \ldots, x_n)$. We show that H can be used to obtain an approximate *rerandomization* **Rer**: This is a procedure that takes an encryption as its input and produces an independent

and identically distributed encryption as its output. Our rerandomization will be approximate in the sense that the input and output of **Rer** will be only statistically close to independent.

One way to obtain rerandomization is as follows: Given a ciphertext C, generate $(n-1)/2$ independent encryptions of 0, $(n-1)/2$ independent encryptions of 1, randomly shuffle them together with C and feed the n resulting ciphertexts to the homomorphic evaluator for majority. By the strong homomorphic property, the output of the homomorphic evaluator will be identically distributed with C. But why should they be independent? From the point of view of the homomorphic evaluator, if C is an encryption of b, then it is indistinguishable from the other $(n-1)/2$ encryptions of b. Since the output of the homomorphic evaluator is bounded in length, the evaluator must "forget" most of the information about most of the ciphertexts it is given as inputs, including C as it is indistinguishable from the others. Therefore the output is forced to look almost statistically independent of C.

In Lemma 5 we generalize this argument to a much wider class of functions which we call *sensitive* (see Section 2) and to weak (i.e., compact) homomorphic evaluators, in which case we obtain a weaker notion of rerandomization.

A strong rerandomization procedure can be used to distinguish encryptions in statistical zero-knowledge by reduction to the "statistical distance" problem: A rerandomized encryption of 0 is statistically close to an encryption of 0, but statistically far from an encryption of 1. Mahmoody and Xiao's simulation of $\mathrm{BPP}^{\mathrm{SZK}}$ in AM [MX10] can then be used to emulate the reduction by a proof system. When only weak one-sided rerandomization is available, it is not clear that encryptions are distinguishable in statistical zero-knowledge, and we construct a somewhat different proof system. For the sake of clarity, however, in the rest of this discussion we will assume the availability of strong rerandomization.

From rerandomization to a distinguishing protocol (Section 5). To turn a reduction from distinguishing encryptions to L into a proof system for $\overline{L}$, we proceed as in previous works: The verifier plays the role of the reduction and the prover plays the role of the distinguishing oracle. The challenge is to force the prover to give answers that are consistent with a specific, fixed distinguishing oracle.

To illustrate the difficulties in the context of public key encryption, let us point out the deficiencies of some naive proof systems. Suppose the verifier submits a public key-ciphertext query (PK, C) to the prover, who is supposed to act as a distinguishing oracle. A natural attempt is to ask the prover to provide the message m and randomness R such that C is an encryption of m under public key PK with randomness R. This fails to account for the possibility that C may not be a valid ciphertext at all: Perhaps there is no pair (m, R) that encrypts to C under PK. It is not clear how a prover can certify such a statement. Another attempt would be to ask the prover for the secret key SK associated to PK. Again, it is not clear how to achieve completeness in case the public key is invalid and there is no corresponding secret key, or soundness in case the public key can be paired with several different secret keys (the choice of which may affect how different invalid ciphertexts decrypt).

Our protocol works as follows: Given a query (PK, C), the verifier asks the prover for the value b that encrypts to C, together with a proof that the rerandomization of C is statistically close to encryptions of b but statistically far from encryptions of $\overline{b}$. If the pair (PK, C) is properly distributed, this forces the prover to give a unique correct answer. But since statistical closeness and statistical farness are both efficiently verifiable [BBM11, SV03], the prover can now also certify that a pair (PK, C) is *not* a valid public key-ciphertext pair. We call this protocol DP (the distinguishing protocol).

One important detail is that the protocols for statistical closeness and statistical farness are only guaranteed to solve promise versions of these problems: For a given gap $[\ell, r)$, they can distinguish distributions that are within statistical distance ℓ from those that are at distance at least r, but give no guarantee about the outcome for instances that fall inside the gap. Therefore DP is only complete and sound provided that none of the underlying instances fall inside the respective gaps.

The proof system (Section 7). Given a reduction R from a decision problem L to distinguishing encryptions, this suggests the following constant-round proof system for $\overline{L}$: On a given input, the verifier chooses randomness for the reduction and sends this randomness to the prover. The prover sends back a transcript of the reduction interacting with a distinguishing oracle, which includes a list of queries (PK_i, C_i) made by the reduction together with an answer a_i saying if C_i encrypts 0 or 1 under PK_i, or the pair (PK_i, C_i) is invalid ($\perp$). The verifier and prover then apply the DP protocol to certify that all the answers a_i are correct.

This proof system is complete and sound, given that all inputs (PK_i, C_i, a_i) to the DP protocol satisfy its promise. But in general the verifier does not know in advance if the promise is satisfied or not. We resolve this issue by choosing the width of the gaps $[\ell, r)$ to be sufficiently small and by having the verifier randomize the location of the gaps. This should make it unlikely for any of the queries to fall inside the promise gap of DP.

This approach was also used by Bogdanov and Trevisan [BT06] in the context of non-adaptive reductions. An additional twist is required when the reduction is adaptive because the location of the gaps may affect the answers of the honest prover. For example, imagine an adaptive reduction that does a "binary search" for the gap $[\ell, r)$: If the first answer a is to the right of r, its next query will be $a/2$, and so on until it hits the gap. To handle such reductions, we want to make the location of the gaps in each round independent of the answers of the honest prover in the previous rounds. On the other hand, the locations of these gaps must be consistent with a specific, fixed distinguishing oracle that the prover is required to emulate.

To achieve both objectives we specify a randomized family of distinguishing oracles, where for each query to the oracle the gap location is random, and the gap locations among the various queries are q-wise independent, where q is an upper bound on the number of queries performed by the reduction. In the first round of the reduction the verifier chooses a random oracle from this family and sends its (polynomial length) description to the prover. The honest prover is

then expected to give answers that are consistent with this instantiation of the distinguishing oracle. By independence, the probability that any of the queries made by the honest prover falls inside the gap will be small. In Section 6.1 we develop the relevant complexity-theoretic framework and we prove Theorem 1 in Section 7.1.

To prevent any of the queries from falling into the gaps $[\ell, r)$, the size of the gaps needs to be inverse proportional to the number of queries made by the reduction. Unless the reduction makes a bounded number of queries, this requires protocols for statistical closeness and statistical farness where the verifier runs in time inverse polynomial to the size of the gap and the gap can be at an arbitrary location. Such protocols were developed by Bhatnagar, Bogdanov, and Mossel [BBM11][1] and we use them in the proof of Theorem 1.[2]

For reductions that make a constant number of queries, it is sufficient to have statistical closeness/farness protocols over a constant number of disjoint gaps $[\ell, r)$. Sahai and Vadhan [SV03] give implementations of such protocols in SZK. Using their protocols and the closure properties of SZK which we recall in Section 6.2, we prove Theorem 2 in Section 7.2.

Better rerandomization from strong homomorphic evaluation. The rerandomization procedure we described above comes with a non-negligible statistical error. It is not difficult to construct examples showing that this error is inherent, even if the homomorphic evaluation is perfect, i.e. it induces no statistical error. In Section 8 we show that the statistical error can be reduced exponentially by iteratively applying the rerandomization on its output, provided f is not "exceptional". This proves Theorem 3.

2 Definitions

Homomorphic evaluation and rerandomization. Let $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ be a bit encryption scheme. Fix a security parameter s and let $(PK, SK) \sim \mathbf{Gen}(1^s)$ the distribution on key pairs. (We will assume that s is implicit in the public and secret keys.)

Definition 1. *Let $f\colon \{0,1\}^n \to \{0,1\}$ be a boolean function. We say H is a* strong homomorphic evaluator *for f with error ε if for all m in the domain of f, the random variables $(PK, H_{PK}(\mathbf{Enc}_{PK}(m_1), \ldots, \mathbf{Enc}_{PK}(m_n)))$ and $(PK, \mathbf{Enc}_{PK}(f(m)))$ (where all encryptions are independent) are within statistical distance ε.*

This definition extends to functions from $\{0,1\}^* \to \{0,1\}$ in a straightforward way. We omit the details.

[1] Technically their statement is not as strong as the one we need here, but their proof can be easily adapted. We provide the details in the full version.

[2] Similar issues arise in the work of Mahmoody and Xiao [MX10]. They work with the SZK-complete problem entropy difference. While their proof can be adapted to our setting, we find it more natural to work directly with instances of statistical difference.

Definition 2. *Let* $f\colon \{0,1\}^* \to \{0,1\}$ *be a boolean function. We say* H *is a* weak homomorphic evaluator *for* f *with error* ε *if (1) the output length of* H *is bounded by a function that depends only on the security parameter and (2) for all* n *and* $m \in \{0,1\}^n$ *in the domain of* f*,*

$$\Pr[\mathbf{Dec}_{SK}(PK, H_{PK}(\mathbf{Enc}_{PK}(m_1), \ldots, \mathbf{Enc}_{PK}(m_n))) = f(m)] \geq 1 - \varepsilon,$$

where all encryptions are independent.[3]

A bit encryption scheme is *efficient* if $\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec}$ all run in time polynomial in the security parameter s. A homomorphic evaluator H is efficient if it is computable in time polynomial in s and n and its output length is polynomially bounded in s.

Definition 3. *Let* $\mathbf{Rer}$ *be a randomized function that takes as input a public key and a ciphertext. In the following definitions* R *and* R' *are independent choices of randomness for* $\mathbf{Rer}$*.*

- *We say* $\mathbf{Rer}$ *is a* strong rerandomization *with error* ε *if for every* $m \in \{0,1\}$*, the random variables* $(PK, E, \mathbf{Rer}_{PK}(E,R))$ *and* (PK, E, E') *where* $E, E' \sim \mathbf{Enc}_{PK}(m)$ *are independent are within statistical distance* ε*.*
- *For* $b \in \{0,1\}$*, we say* $\mathbf{Rer}^b$ *is a* one-sided weak rerandomization *with decryption error* ε *and rerandomization error* ρ *if for every* $m \in \{0,1\}$*,* $\Pr[\mathbf{Dec}_{SK}(\mathbf{Rer}^b_{PK}(\mathbf{Enc}_{PK}(m))) = m] \geq 1 - \varepsilon$ *and the random variables* $(PK, \mathbf{Rer}^b_{PK}(E,R), \mathbf{Rer}^b_{PK}(E,R'))$ *and* $(PK, \mathbf{Rer}^b_{PK}(E,R), \mathbf{Rer}^b_{PK}(E',R'))$ *where* $E, E' \sim \mathbf{Enc}_{PK}(b)$ *are independent are within statistical distance* ρ*.*

We say the rerandomization is *efficient* if it can be evaluated in time polynomial in the security parameter.

Sensitivity of boolean functions. We will use the following notion of sensitivity for boolean functions. For $x \in \{0,1\}^k$ let $x|_i$ be the string obtained by flipping the i-th bit of x and leaving the others unchanged. Let $f\colon \{0,1\}^k \to \{0,1\}$ be a boolean function and $b \in \{0,1\}$. We say f has *b-sensitivity* at least s if there exists an input $x \in \{0,1\}^k$ and a set $S \subseteq [k]$ of size s such that $f(x) = b$, $x_i = b$ for every $i \in S$, and $f(x|_i) = \overline{b}$ for every $i \in S$. We call (x, S) a witness that f has b-sensitivity at least s.

We say a family of functions $f = \{f_k\colon \{0,1\}^k \to \{0,1\}\}$ has *certifiable polynomial b-sensitivity* if there exists a constant $\alpha > 0$ so that on input k we can compute in time polynomial in k a witness that f_k has b-sensitivity at least k^α.

Examples of functions that have certifiable polynomial 0-sensitivity and 1-sensitivity include parity and majority. The AND function has certifiable polynomial 0-sensitivity while the OR function has certifiable polynomial 1-sensitivity.

[3] Some works adopt the terms "distribution preserving" and "compact" homomorphic evaluation. We prefer the terms "strong" and "weak" for this work, as we are concerned with questions of computational complexity.

Examples of functions whose 0-sensitivity and 1-sensitivity is less than s are functions that depend on at most $s-1$ of their inputs, i.e. $(s-1)$-juntas. Simon [Sim82] gives an example of a function on k bits that depends on all its inputs but has 0-sensitivity and 1-sensitivity $O(\log k)$.

3 The Main Theorems

We say $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ supports weak homomorphic evaluation of f with error ε if it has an efficient homomorphic evaluator for f with error ε.

A γ-*distinguishing oracle* for $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ is a function D such that

$$\Pr[D(PK, \mathbf{Enc}_{PK}(0)) \text{ accepts}] - \Pr[D(PK, \mathbf{Enc}_{PK}(1)) \text{ accepts}] > \gamma.$$

A *reduction* from a decision problem L to γ-distinguishing encryptions is an efficient randomized oracle algorithm $R^?$ such that for every valid input x there exists a γ-distinguishing oracle D such that $R^D(x) = L(x)$ with probability at least $8/9$. (For our results the exact constant won't matter, as long as it is strictly greater than $1/2$.)

Theorem 1. *Let f_0 and f_1 be functions with certifiable polynomial 0-sensitivity and 1-sensitivity respectively (possibly the same function). Let $\varepsilon \in (0, 1/18)$ be any constant and $\delta \geq 2\sqrt{\varepsilon}$. Let $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ be a public key encryption scheme that supports efficient homomorphic evaluations of both f_0 and f_1 with error at most ε. If there is a reduction from L to $(1-\delta)$-distinguishing $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$, then L is in $\mathrm{AM} \cap \mathrm{coAM}$.*

We will assume that the reduction is *query length regular*: On input x, the reduction first computes a query length $m \geq |x|$ and only makes queries of length m. The theorem can be proved without this assumption, but we make it for notational convenience.

In the case when the reduction has constant query complexity, a stronger conclusion can be obtained.

Theorem 2. *Let f_0 and f_1 be functions with certifiable polynomial 0-sensitivity and 1-sensitivity respectively (possibly the same function). Let q be any constant, $\delta > 0$, and $\varepsilon = \varepsilon(q, \delta)$ a sufficiently small constant. Let $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ be a public key encryption scheme that supports efficient homomorphic evaluations of f_0 and f_1 with error at most ε. If there is a reduction from L to $(1-\delta)$-distinguishing $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ that makes at most q queries, then L is in statistical zero-knowledge.*

In particular, Theorems 1 and 2 apply to the following cases: (1) f_0 and f_1 are the parity function; (2) f_0 and f_1 are the majority function; (3) f_0 is OR and f_1 is AND.

Ron Rothblum [Rot11] shows how to turn a private-key encryption scheme into a public-key one using a homomorphic evaluator for parity. Combining the two results, the conclusions of Theorems 1 and 2 can be extended to private-key encryption schemes that support homomorphic evaluation of parity.

Our last result shows how to obtain strong rerandomization given a homomorphic evaluator for almost any function. We call a function $f\colon \{0,1\}^n \to \{0,1\}$ *exceptional* if it is one of the following functions of the inputs that it depends on: the constant 0, the constant 1, the identity, the NOT function, the AND function, the OR function.

Theorem 3. *Assume $f\colon \{0,1\}^n \to \{0,1\}$ is not exceptional. If $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ is a public key encryption scheme that supports efficient strong homomorphic evaluation of f with negligible error, then $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ has an efficient strong rerandomization with negligible error.*

4 One-Sided Rerandomization from Homomorphic Evaluation

In this section we show how to convert a homomorphic evaluation algorithm for a sensitive function into a one-sided rerandomization. In Section 8 we extend these ideas to obtain stronger notions of rerandomization under stronger assumptions. Let H denote entropy and I denote mutual information.

Claim 4. *Let $X_1, \ldots, X_n$ be i.i.d. random variables and $I \sim \{1, \ldots, n\}$ a uniformly random index. Let F, G, G' be random variables such that (1) F and G are independent conditioned on X_I, (2) F is independent of I, (3) G and G' are identically distributed and (4) F and G' are independent. Then the random variables (F, G) and (F, G') are within statistical distance $\sqrt{2\,\mathrm{H}(F)/n}$.*

Proof

$$\mathrm{H}(X_I \mid F) \geq \mathrm{H}(X_I \mid F, I) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{H}(X_i \mid F)$$

$$\geq \frac{1}{n}\,\mathrm{H}(X_1, \ldots, X_n \mid F) \geq \frac{1}{n}(\mathrm{H}(X_1, \ldots, X_n) - \mathrm{H}(F)) = \mathrm{H}(X_I) - \frac{\mathrm{H}(F)}{n}.$$

Since F and G are conditionally independent of X_I, $\mathrm{I}(F; G) \leq \mathrm{I}(F; X_I)$ and so

$$\mathrm{I}(F; G) \leq \mathrm{I}(F; X_I) = \mathrm{H}(X_I) - \mathrm{H}(X_I \mid F) \leq \frac{\mathrm{H}(F)}{n}.$$

The conclusion follows by Pinsker's inequality [Pin64]. □

The following lemma shows how to obtain one-sided rerandomization from homomorphic evaluation of a sensitive function. This lemma will be used in the proofs of Theorems 1 and 2. In Section 8 we give a version of this lemma that applies to a more restricted class of functions but allows us to achieve a stronger notion of rerandomization. That version will be used for the proof of Theorem 3.

Lemma 5. *Assume f has certifiable polynomial b-sensitivity and let δ be any function inverse polynomial in the security parameter. If $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ has a weak efficient homomorphic evaluator for f with error ε, then it has a one-sided weak rerandomization $\mathbf{Rer}^b$ with decryption error ε and rerandomization error δ.*

Proof. Suppose f_k has b-sensitivity k^α. Choose $k = (2c/\delta^2)^{1/\alpha}$, where c is the length of ciphertexts (for the given security parameter). Let (x, S) be the witness for b-sensitivity of f_k. Given public key PK and ciphertext E define $\mathbf{Rer}^b$ as follows:

1. Choose a random $I \sim S$.
2. Let
$$X_i = \begin{cases} \mathbf{Enc}_{PK}(x_i, R_i) & \text{if } i \neq I, \\ E & \text{if } i = I. \end{cases}$$
3. Output $F = H_{PK}(X_1, \ldots, X_k)$.

We first condition on the choice of the public key PK, letting ε_{PK} denote the statistical distance between the two distributions in the definition of strong homomorphic evaluator conditioned on PK.

The decryption error of $\mathbf{Rer}^b$ follows directly from the definition. We now show the rerandomization error is at most δ. Let F, G be two independent instantiations of $\mathbf{Rer}^b$ on the same input E. Conditioned on PK, the random variables $X_i : i \in S$ and F, G satisfy the assumptions of Claim 4. It follows that (F, G) and (F, G'), where G' is i.i.d with G and therefore with F, are within statistical distance $\sqrt{2c/k^\alpha}$, which is at most δ by our choice of parameters. Averaging over ε_{PK} we prove the lemma. □

5 The Distinguishing Protocol

In this section we describe a constant-round interactive proof system DP that, given input (PK, C, b), certifies that C is an encryption of b under PK when $b \in \{0, 1\}$ and that (PK, C) is an invalid pair when $b = \perp$. The proof system is parametrized by two gaps $[\ell, r)$ and $[\ell', r')$, which describe a promise on the inputs.

We will assume we have the following constant-round protocols for statistical closeness ($SC_{[\ell,r)}$) and statistical farness ($SF_{[\ell,r)}$), where $0 \leq \ell < r \leq 1$. The protocols take as inputs a pair of sampler circuits D, D' producing distributions over the same set $\{0, 1\}^m$ with the following completeness / soundness properties:

- If D, D' are at statistical distance less than ℓ / at least r, $SC_{[\ell,r)}(D, D')$ accepts / rejects with probability $1 - \sigma$.
- If D, D' are at statistical distance at least r / less than ℓ, $SF_{[\ell,r)}(D, D')$ accepts / rejects with probability $1 - \sigma$.

Here σ can be any inverse polynomial in the size of the input. The following two theorems state the existence of these protocols. The second one is stronger as it provides statistical zero-knowledge implementation, but makes a stronger assumption about the gaps.

Formally we will view SC and SF as promise problems that take ℓ, r, D, D' as their inputs. Theorem 6 essentially follows from work of Bhatnagar, Bogdanov, and Mossel [BBM11]. We provide the missing details in the full version..

Theorem 6. *For $r > \ell$, the problems SC and SF are in* AM *where the running time of the verifier is polynomial in the size of D, the size of D′, and $1/(r-\ell)$.*

Theorem 7 is proved by Sahai and Vadhan [SV03].

Theorem 7. *For $r^2 > \ell$, the problems SC and SF are in* SZK *where the running time of the verifier is polynomial in the size of D, the size of D′, and $1/\ell^{1/\log(r^2/\ell)}$.*

The protocol DP will certify that the rerandomization of C is close to an rerandomized encryption of b but far from a rerandomized encryption of $\overline{b}$ when $b \in \{0,1\}$. When $b = \perp$, it certifies that either the rerandomized encryptions of 0 and 1 are close, or the rerandomized encryption of C is far from both.

Let $Z_{PK,b}$ ($b \in \{0,1\}$) be the following circuit: On input R, R', output $\mathbf{Rer}^b_{PK}(\mathbf{Enc}_{PK}(b,R),R')$, i.e. a one-sided rerandomized encryption of b.

The distinguishing protocol $DP_{[\ell,r),[\ell',r')}$

On input (PK, C, b), where $b \in \{0,1,\perp\}$:
1. If $b = 0$ or $b = 1$:
2. Verifier and Prover execute $SF_{[\ell,r)}(Z_{PK,0}, Z_{PK,1})$.
3. If the protocol rejects, reject. Otherwise:
4. Verifier and Prover execute $SC_{[\ell',r')}(Z_{PK,b}, \mathbf{Rer}^b_{PK}(C))$.
5. If the protocol accepts, accept, else reject.
6. If $b = \perp$:
7. Verifier and Prover execute $SC_{[\ell,r)}(Z_{PK,0}, Z_{PK,1})$.
8. If the protocol accepts, accept. Otherwise:
9. Verifier and Prover execute $SF_{[\ell',r')}(Z_{PK,0}, \mathbf{Rer}^0_{PK}(C))$.
10. Verifier and Prover execute $SF_{[\ell',r')}(Z_{PK,1}, \mathbf{Rer}^1_{PK}(C))$.
11. If both accept, accept, else reject.

The distinguishing oracle. We define an oracle π that distinguishes between encryptions of 0 and encryptions of 1. This oracle answers $\perp$ on all queries (PK, C) that do not represent valid key-ciphertext pairs and answers bad on all queries that fall inside the gaps of the underlying protocols SC and SF. We then show that for every pair (PK, C) that falls outside the gaps, $b = \pi(PK, C)$ is the unique answer that makes $DP(PK, C, b)$ accept. Owing to lack of space the definition of π, as well as the proof of the following claim which shows π is a distinguishing oracle, are given in the full version.

Claim 8. *Assume $\mathbf{Rer}^0, \mathbf{Rer}^1$ are one-sided rerandomizations with decryption error $\varepsilon < (1-r)^2/2$ and rerandomization error $\rho < \ell'^2$. Then $\Pr[\pi(PK, \mathbf{Enc}_{PK}(b)) = b] \geq 1 - \sqrt{2\varepsilon} - \sqrt{\rho}$ for every $b \in \{0,1\}$.*

The following claims are immediate from the definitions and the completeness and soundness assumptions on SC and SF.

Claim 9. *(Completeness) Assume $\ell' < r/2$ and $\pi(PK, C) \neq$ bad. Then $DP(PK, C, \pi(PK, C))$ accepts with probability at least $1 - 3\sigma$.*

Claim 10. *(Soundness) Assume $\ell' < r/2$. If $DP(PK, C, b)$ accepts with probability more than 3σ, then $\pi(PK, C) \in \{b, \text{bad}\}$.*

6 Complexity Theoretic Setup

In this section we cover the complexity-theoretic framework for the proofs of Theorems 1 and 2. Proof of the claims can be found in the full version.

6.1 Promise Oracles for Adaptive Reductions

Let Ξ be any finite set of values that includes the special symbol bad. An *oracle family* over input length m with size d is a multiset Π of functions $\pi\colon \{0,1\}^m \to \Xi$. We say Π is ε-*bad* if for every input x, $\Pr_{\pi \sim \Pi}[\pi(x) = \text{bad}] \leq \varepsilon$.

Let $F\colon \{0,1\}^m \to [d]$ be a function. The oracle $\Pi_F\colon \{0,1\}^m \to \Xi$ is given by $\Pi_F(z) = \pi_{F(z)}(z)$. In the lemma below F will be a randomized function of the same form.

Lemma 11. *Let $R^?$ be a reduction that on an input of length n, makes at most q queries of length m. Let Π be an oracle family of size d. Assume d is a power of two. There exists a randomized function $F\colon \{0,1\}^m \to [d]$ such that:*

- *F is computable in time (and hence uses randomness) polynomial in m, q, and d.*
- *For every input x of length n, the probability that $R^{\Pi_F}(x)$ never receives* bad *as an answer to any of its queries is at least $(1 - \varepsilon)^q$.*

6.2 Statistical Zero-Knowledge

We recall some results about the complexity of statistical zero-knowledge SZK. Sahai and Vadhan [SV03] show that the statistical distance problem $SD = SF_{[1/9,8/9)}$ is complete for SZK under many-one reductions.

We also need the following result of Sahai and Vadhan [SV03] that states the closure of SZK under truth-table reductions.

Theorem 12. *There is a deterministic algorithm that takes as input instances $x_1, \ldots, x_k$ of SD and a boolean predicate $P\colon \{0,1\}^k \to \{0,1\}$ and outputs an instance x of SD such that $SD(x) = P(SD(x_1), \ldots, SD(x_k))$. The running time of the algorithm is polynomial in 2^k and the sizes of $x_1, \ldots, x_k$.*

We also need the following fact, which says that reductions within SZK can without loss of generality be assumed deterministic.

Claim 13. *If there is a randomized many-one reduction R from L to SD such that $\Pr[SD(R(x)) = L(x)] \geq p$, where p is any constant above $1/2$, then L is in* SZK.

Combining Theorem 12 and Claim 13 we get the following corollary.

Corollary 14. *Suppose there is a randomized algorithm A that on input x of length n and randomness r computes inputs $x_1, \ldots, x_k$ and a predicate $P\colon \{0,1\}^k \to \{0,1\}$, where $k = O(\log n)$ and accepts if $P(SD(x_1), \ldots, SD(x_k))$ is true. If $\Pr[A(x) = L(x)] \geq p$, where p is any constant greater than $1/2$, then L is in* SZK.

7 Proofs of the Main Theorems

7.1 Proof of Theorem 1

Let $F_\omega\colon \{0,1\}^m \to [d]$ be the randomized function from Lemma 11, with ω describing the randomness. We set $I_j = \left[\frac{1}{3} + \frac{j-1}{3d}, \frac{1}{3} + \frac{j}{3d}\right)$ and $I'_j = \frac{1}{3} I_j$, where $1 \leq j \leq d$.

The decision protocol DL: On input x:

V: Compute the oracle query length m. Let d be the smallest power of two above $90q$ where q is an upper bound on the number of queries $R^?(x)$ makes. Choose randomness r for the reduction and randomness ω for F_ω. Send r, d, ω to the prover.

P: Send a sequence $((PK_i, C_i), b_i)$, $1 \leq i \leq q$ of oracle query-answer pairs.

V: Verify that the received query-answer pairs determine an accepting computation of $R^?(x, r)$. If not, reject. For every query i, compute $j = F_\omega(PK_i, C_i)$ and let $[\ell_i, r_i) = I_j$ and $[\ell'_i, r'_i) = I'_j$.

B: Execute in parallel the protocols $DP_{[\ell_i, r_i), [\ell'_i, r'_i)}(PK_i, C_i, b_i)$ for $1 \leq i \leq q$ with completeness/soundness gap $\sigma = 1/9q$. If any of them result in rejection, reject. Otherwise, accept.

Let $\pi_j = \pi_{I_j, I'_j}$ and Π_F be the oracle from Lemma 11.

Claim 15. *The oracle family $\{\pi_j\}_{1 \leq j \leq d}$ is at most $3/d$-bad.*

Proof of Theorem 1. It is sufficient to prove that $L \in$ AM. By applying the same argument to its complement $\overline{L}$ we also get $L \in$ coAM.

Assume $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ supports homomorphic evaluation of f with error at most ε and there is a reduction $R^?$ from L to $(1-\delta)$-distinguishing encryptions.

We instantiate the constructions with the following parameters. Let ε be the homomorphic evaluation error and c an upper bound on the length of ciphertexts queried by the reduction on input x. Let $\mathbf{Rer}^b$ be the rerandomization from Lemma 5 with parameters chosen so that the decryption error is ε and the rerandomization error is at most $\rho \leq \varepsilon^2$. The protocol DP is instantiated with the rerandomizations $\mathbf{Rer}^0$ and $\mathbf{Rer}^1$.

Claim 16. *For an appropriate choice of parameters and for every F, Π_F is a $(1-\delta)$-distinguishing oracle.*

By Theorem 6, the verifier for DL can be implemented in polynomial time. Theorem 1 the follows by the next two claims:

Claim 17. *(Completeness) If $x \in L$, there exists a prover that makes $DL(x)$ accept with probability at least $2/3$.*

Claim 18. *(Soundness) If $x \notin L$ then no prover makes $DL(x)$ accept with probability at least $1/3$.*

7.2 Proof of Theorem 2

Let $I_j, 1 \leq j \leq d$ be the following collection of intervals: $I_j = [\ell_j, r_j)$ where $r_1 = 1/2$, $\ell_j = r_j^2/4$, and $r_{j+1} = \ell_j$. Let $I'_j = \frac{1}{3} I_j$. Assume the reduction makes at most q queries on every input and let $d = 27q \cdot 3^q$.

By Theorem 7, for every j the problems $SC_{I_j}, SC_{I'_j}, SF_{I_j}, SF_{I'_j}$ are all in SZK so by Theorem 12 and the completeness of SD, DP_{I_j,I'_j} is also in SZK for every j.

Consider the following algorithm A. On input x, choose randomness r for R and a random $j \sim [d]$ and accept if there exists a sequence of answers $(a_1, \ldots, a_q) \in \{0, 1, \perp\}^q$ such that $R(x, r)$ accepts given these oracle answers and $DP_{I_j,I'_j}(Q_i, a_i)$ accepts for all $1 \leq i \leq q$. Since DP_{I_j,I'_j} is in SZK and SD is complete for SZK, A satisfies the assumption of Corollary 14, so if we can prove that $\Pr[A(x) = L(x)] \geq 2/3$, it will follow that L is in SZK.

Say j is bad if $\pi_j = \pi_{I_j,I'_j}$ answers bad on any pair (Q, a) queried by A. Since A makes at most $q3^q$ queries, by Claim 15 and a union bound the probability that A answers bad on any of its queries is at most $1/9$.

Fix an input x. By our choice of parameters, when ε is sufficiently small and $\rho = \varepsilon^2$, Claim 8 guarantees that π_j is a $(1 - 4\varepsilon)$-decryption oracle for every $1 \leq j \leq d$. So for at least $8/9$ fraction of r, $R^{\pi_j}(x, r) = L(x)$. Therefore with probability at least $7/9$, both $R^{\pi_j}(x, r) = L(x)$ and π_j never answers bad on any of A's queries. By Claims 9 and Claim 10, it must then hold that $a = \pi_j(Q)$ for all query-answer pairs (Q, a) made by A, and so $A(x) = L(x)$.

8 Strong Rerandomization from Strong Homomorphic Evaluation

In this Section we prove Theorem 3. We begin by defining "t-symmetric functions". The proofs of the claims in this section can be found in the full version.

t-symmetric functions. Let G be a subgroup of the symmetric group S_k and $x \in \{0, 1, \star\}^k$ be a string containing exactly one $\star$. Let $t_0(G, x)$ (resp., $t_1(G, x)$) be the number of transpositions $\tau \in G$ that transpose a 0 and a $\star$ (resp., a 1 and a $\star$) when acting on x. Observe that $t_b(G, \sigma x) = t_b(G, x)$ for every $\sigma \in G$.

Let $x|_{\star \to 0}, x|_{\star \to 1}$ be the string obtained when the $\star$ in x is replaced by a 0 and a 1 respectively. We will say a boolean function $f \colon \{0,1\}^k \to \{0,1\}$ is t-symmetric if there exist x and G with $t_0(G, x), t_1(G, x) > t$ and $f(\sigma x|_{\star \to b}) = b$ for every $\sigma \in G$.

For example, the majority function on 3 bits is 2-symmetric: Take $G = S_3$ and let $x = 01\star$. So is parity on 4 bits: Take $G = S_4$ and $x = 110\star$. The DNF $(x_{11} \wedge x_{12}) \vee (x_{21} \wedge x_{22})$ is also 2-symmetric. To see this take x to be the string $x_{11} = \star, x_{12} = 1, x_{21} = 0, x_{22} = 1$ and G to be the "wreath product" $S_2 \wr S_2$, which acts on x by first permuting the inputs in each term of the DNF independently, then permuting the terms.

Proof of Theorem 3 The theorem follows from the next two claims, proved below.

Claim 19. *Let $f\colon \{0,1\}^k \to \{0,1\}$, $k \geq 2$ be any boolean function that depends on all its inputs and is not one of OR / AND. If* $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ *supports efficient strong homomorphic evaluation of f with error ε, then* $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ *supports efficient strong homomorphic evaluation of a 2-symmetric function with error at most 12ε.*

Claim 20. *Let $f\colon \{0,1\}^k \to \{0,1\}$ be a 2-symmetric function. If* $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ *is a public key encryption scheme that supports efficient strong homomorphic evaluation of f with negligible error, then* $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ *has an efficient strong rerandomization with negligible error.*

8.1 Proof of Claim 19

Claim 21. *Let $f\colon \{0,1\}^k \to \{0,1\}$, $k \geq 2$ be a monotone function that depends on all its inputs.*

1. *If f is not the AND function, then f has 0-sensitivity at least 2.*
2. *If f is not the OR function, then f has 1-sensitivity at least 2.*

Let $f\colon \{0,1\}^k \to \{0,1\}$ be a boolean function. We say f is an *extension* of g if there exists a set $S \in [k]$ and $z \in \{0,1\}^{\overline{S}}$ such that g is the restriction of f to S using z, i.e. $f_{S|z}(x) = g(x)$ for every $x \in \{0,1\}^S$.

Claim 22. *Let g be a function with b-sensitivity at least s and f be any extension of g. Let* $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ *be a public key encryption scheme. If* $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ *supports strong homomorphic evaluation of f with error ε,* $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ *supports strong homomorphic evaluation of g with error ε.*

Claim 23. *Let $g\colon \{0,1\}^k \to \{0,1\}$ be a boolean function. For every $i \in [k]$, let $f_i\colon \{0,1\}^{k_i} \to \{0,1\}$ be a boolean function. Let* $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ *be a public key encryption scheme. If* $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ *supports strong homomorphic evaluation of g with error ε and each of the f_i's with error ε_i, then* $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ *supports strong homomorphic evaluation of $g(f_1, \ldots, f_k)$ with error $\varepsilon + \varepsilon_1 + \cdots + \varepsilon_k$.*

Proof (of Claim 19). First, we show that $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ supports homomorphic evaluation of f_0 and f_1 with error at most 4ε, where f_b has b-sensitivity 2. Consider the 2-symmetric function $g\colon \{0,1\}^4 \to \{0,1\}$ defined by $g(x_{11}, x_{12}, x_{21}, x_{22}) = f_0(f_1(x_{11}, x_{12}), f_1(x_{21}, x_{22}))$. Since g is a composition of

f_0 and f_1, by Claim 23 $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ has a strong homomorphic evaluation of g with error at most 12ε.

Now we show that $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ supports homomorphic evaluation of f_0 and f_1. This follows from Claim 21 and 22 if f is monotone. If f is not monotone, there is an $x \in \{0,1\}^k$ and $i \in [k]$ such that $x_i = 1$, $f(x) = 0$ and $f(x|_i) = 1$. Let g be the restriction of f to the rest of the bits using x_i. Note that g is the NOT function and so by Claim 22 $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ supports homomorphic evaluation of the NOT function with error ε. It is easy to see that one can obtain f_0 and f_1 by composing g with a restriction of f. The rest follows by Claim 23. □

8.2 Proof of Claim 20

We start with the following Corollary of Claim 4 for the special case when $G = X_I$.

Corollary 24. *Let $X_1, \ldots, X_n$ be i.i.d and $I \sim \{1, \ldots, n\}$ a uniformly random index and F be independent of I. Then (F, X_I) and (F, X) are within statistical distance $\sqrt{2\,\mathrm{H}(F)/n}$, where X is i.d. with $X_1, \ldots, X_n$ and independent of F.*

The following lemma shows how to obtain strong rerandomization from any t-symmetric function. The resulting rerandomization error is noticeable. It is similar to Lemma 5 and the proof is given in the full version.

Lemma 25. *Let $f\colon \{0,1\}^k \to \{0,1\}$ be any t-symmetric function. If $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ has a strong efficient homomorphic evaluator for f with error ε, then it has a strong efficient rerandomization $\mathbf{Rer}$ with error at most $\varepsilon + \sqrt{2c/t}$ (resp. decryption error ε and rerandomization error $\sqrt{2c/t}$), where c is the length of ciphertexts.*

We now show that for strong homomorphic evaluation, the error can be reduced and prove Theorem 3.

For a boolean function $f\colon \{0,1\}^k \to \{0,1\}$, Let $f^{(r)}\colon \{0,1\}^{k^r} \to \{0,1\}$ be defined recursively by first applying $f^{(r-1)}$ on k independent tuples of k^{r-1} inputs and then applying f to these k values. For the base case we take $f^{(1)} = f$.

Claim 26. *If f is t-symmetric, then $f^{(r)}$ is t^r-symmetric.*

Proof (of Claim 20). Let $\mathbf{Rer}$ be the rerandomization of f from the proof of Lemma 25. We define $\mathbf{Rer}^{(r)}$ recursively by $\mathbf{Rer}^{(1)} = \mathbf{Rer}$ and

$$\mathbf{Rer}^{(r)}_{PK}(E, (R_1, \ldots, R_r)) = \mathbf{Rer}_{PK}(\mathbf{Rer}^{(r-1)}_{PK}(E, (R_1, \ldots, R_{r-1})), R_r).$$

where $R_1, \ldots, R_r$ are independent random strings. We now argue that $\mathbf{Rer}^{(r)}$ has the desired properties.

Let $\mathbf{Rer}'^{(r)}$ be the rerandomization obtained by applying the construction of Lemma 25 to the function $f^{(r)}$. We claim that the distributions $(PK, E, \mathbf{Rer}^{(r)}_{PK}(E))$ and $(PK, E, \mathbf{Rer}'^{(r)}_{PK}(E))$, where $E \sim \mathbf{Enc}_{PK}(b)$, are within statistical distance at

most εk^{r-1}. We show this by induction. The base case $r = 1$ is obvious (the statistical distance is zero).

For the inductive step, we can describe $\mathbf{Rer}^{(r)}_{PK}(E)$ as follows: First, choose X by applying a random permutation π to the indices of $x \in \{0, 1, \star\}$. Then $\mathbf{Rer}^{(r)}_{PK}(E) = H_{PK}(e_1, \ldots, e_k)$ where $e_i = \mathbf{Enc}_{PK}(X_i)$ when $X_i \neq \star$ and $e_i = \mathbf{Rer}^{(r-1)}_{PK}(E)$ when $X_i = \star$. On the other hand $\mathbf{Rer}'^{(r)}_{PK}(E)$ can be described as follows: First, choose X by applying a random permutation π to the indices of $x \in \{0, 1, \star\}$. Then $\mathbf{Rer}'^{(r)}_{PK}(E) = H_{PK}(e'_1, \ldots, e'_k)$ where $e'_i = \mathbf{Rer}'^{(r-1)}_{PK}(\mathbf{Enc}_{PK}(X_i))$ when $X_i \neq \star$ and $e'_i = \mathbf{Rer}'^{(r-1)}_{PK}(E)$ when $X_i = \star$. By inductive assumption, the statistical distance between $(PK, \mathbf{Rer}^{(r-1)}_{PK}(E))$ and $(PK, \mathbf{Rer}'^{(r-1)}_{PK}(E))$ is at most εk^{r-2}. Since H_{PK} has error ε, the statistical distance between $(PK, \mathbf{Enc}_{PK}(b))$ and $(PK, \mathbf{Rer}'^{(r-1)}_{PK}(\mathbf{Enc}_{PK}(b))$ can also be bounded by εk^{r-2} using an inductive argument. Applying a hybrid argument we conclude that the distributions $(PK, e_1, \ldots, e_k)$ and $(PK, e'_1, \ldots, e'_k)$ are within distance at most εk^{r-1} and therefore so are the distributions $(PK, \mathbf{Rer}^{(r)}_{PK}(E))$ and $(PK, \mathbf{Rer}'^{(r)}_{PK}(E))$.

By Claim 26, $f^{(r)}$ is t^r symmetric. It follows from Claim 23 that the function $H^{(r)}_{PK}$ defined recursively by $H^{(1)}_{PK} = H_{PK}$ and $H^{(r)}_{PK} = H_{PK}(H^{(r-1)}_{PK}, \ldots, H^{(r-1)}_{PK})$ is a homomorphic evaluation of $f^{(r)}$ with error at most εk^r. By Lemma 25, $\mathbf{Rer}'^{(r)}$ has error $k^r\varepsilon + \sqrt{2c/t^r}$. Therefore $\mathbf{Rer}^{(r)}$ has error at most $\varepsilon(k^{r-1} + k^r) + \sqrt{2c/t^r}$. Let $\alpha = \log t / \log k$. By choosing $r = 1/(2+\alpha) \cdot \log(2c/\varepsilon^2)/\log k$ we get that $\mathbf{Rer}^{(r)}$ has error $O(\varepsilon^{\alpha/(2+\alpha)})$, which is negligible when ε is negligible. □

Acknowledgments. We thank Benny Applebaum for explaining the relevance of the work of Mahmoody and Xiao, Alon Rosen for helpful discussions about the power of our rerandomization procedure, and Cheuk Ting Li for supplying a counterexample to a conjectured generalization of Lemma 25.

References

[AGGM06] Akavia, A., Goldreich, O., Goldwasser, S., Moshkovitz, D.: On basing one-way functions on NP-hardness. In: Proceedings of the 38th ACM Symposium on Theory of Computing (2006)

[Ajt96] Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: Proceedings of the 28th ACM Symposium on Theory of Computing, STOC 1996, pp. 99–108. ACM, New York (1996)

[BBM11] Bhatnagar, N., Bogdanov, A., Mossel, E.: The computational complexity of estimating MCMC convergence time. In: Goldberg, L.A., Jansen, K., Ravi, R., Rolim, J.D.P. (eds.) RANDOM 2011 and APPROX 2011. LNCS, vol. 6845, pp. 424–435. Springer, Heidelberg (2011)

[Bra79] Brassard, G.: Relativized cryptography. In: Proceedings of the 20th IEEE Symposium on Foundations of Computer Science, pp. 383–391 (1979)

[BT06] Bogdanov, A., Trevisan, L.: On wost-case to average-case reductions for NP problems. SIAM J. Comp. 36(4) (2006)

[BV11] Brakerski, Z., Vaikuntanathan, V.: Efficient Fully Homomorphic Encryption from (Standard) LWE. In: Proceedings of the 53rd Annual Symposium on Foundations of Computer Science (2011)

[EY80] Even, S., Yacobi, Y.: Cryptography and NP-completeness. In: de Bakker, J.W., van Leeuwen, J. (eds.) ICALP 1980. LNCS, vol. 85, pp. 195–207. Springer, Heidelberg (1980)

[FF93] Feigenbaum, J., Fortnow, L.: Random-self-reducibility of complete sets. SIAM Journal on Computing 22, 994–1005 (1993)

[Gam85] El Gamal, T.: A public-key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Info. Theory 31(4), 469–472 (1985)

[Gen09] Gentry, C.: Fully Homomorphic Encryption Using Ideal Lattices. In: STOC, pp. 169–178 (2009)

[GG98] Goldreich, O., Goldwasser, S.: On the possibility of basing cryptography on the assumption that P $\neq$ NP (1998) (unpublished manuscript)

[Gol00] Goldreich, O.: Candidate one-way functions based on expander graphs. In: Electronic Colloquium on Computational Complexity (ECCC), vol. 7(90) (2000)

[HMX10] Haitner, I., Mahmoody, M., Xiao, D.: A new sampling protocol and applications to basing cryptographic primitives on NP. In: Proceeedings of 25th IEEE Conference on Computational Complexity (CCC), pp. 76–87 (2010)

[MX10] Mahmoody, M., Xiao, D.: On the power of randomized reductions and the checkability of sat. In: Proceeedings of 25th IEEE Conference on Computational Complexity (CCC), pp. 64–75 (2010)

[Pai99] Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)

[Pei09] Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: Proceedings of the 41th ACM Symposium on Theory of Computing, pp. 333–342. ACM, New York (2009)

[Pin64] Pinsker, M.S.: Information and information stability of random variables and processes. Holden-Day (1964)

[Reg09] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM 56(6) (2009)

[Rot11] Rothblum, R.: Homomorphic encryption: From private-key to public-key. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 219–234. Springer, Heidelberg (2011)

[Sim82] Simon, H.-U.: A tight $\log \log n$-bound on the time for parallel ram's to compute nondegenerated boolean functions. Information and Control 55(1), 102–107 (1982)

[SV03] Sahai, A., Vadhan, S.: A complete problem for statistical zero knowledge. J. ACM 50, 196–249 (2003)

[vDGHV10] van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully Homomorphic Encryption from Integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (2010)

Counter-Cryptanalysis

Marc Stevens

CWI, Amsterdam, The Netherlands
marc@marc-stevens.nl

Abstract. We introduce *counter-cryptanalysis* as a new paradigm for strengthening weak cryptographic primitives against cryptanalytic attacks. Redesigning a weak primitive to more strongly resist cryptanalytic techniques will unavoidably break backwards compatibility. Instead, counter-cryptanalysis exploits unavoidable anomalies introduced by cryptanalytic attacks to detect and block cryptanalytic attacks while maintaining full backwards compatibility. Counter-cryptanalysis in principle enables the continued secure use of weak cryptographic primitives.

Furthermore, we present the first example of counter-cryptanalysis, namely the efficient detection whether any given single message has been constructed – together with an *unknown* sibling message – using a cryptanalytic collision attack on MD5 or SHA-1.

An immediate application is in digital signature verification software to ensure that an (older) MD5 or SHA-1 based digital signature is not a forgery using a collision attack. This would certainly be desirable for two reasons. Firstly, it might still be possible to generate malicious forgeries using collision attacks as too many parties still sign using MD5 (or SHA-1) based signature schemes. Secondly, any such forgeries are currently accepted nearly everywhere due to the ubiquitous support of MD5 and SHA-1 based signature schemes. Despite the academic push to use more secure hash functions over the last decade, these two real-world arguments (arguably) will remain valid for many more years.

Only due to counter-cryptanalysis were we able to discover that Flame, a highly advanced malware for cyberwarfare uncovered in May 2012, employed an as of yet unknown variant of our chosen-prefix collision attack on MD5 [SLdW07, SSA+09]. In this paper we disect the revealed cryptanalytic details and work towards the reconstruction of the algorithms underlying Flame's new variant attack. Finally, we make a preliminary comparision between Flame's attack and our chosen-prefix collision attack.

1 Introduction

1.1 Weak Cryptographic Primitives

Cryptographic primitives that are broken or weak due to the existence of cryptanalytic attacks should be retired in favor of a more secure one. However, in practice, widely used cryptographic primitives that are broken continue to be

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 129–146, 2013.

used long after their expiration date. This phenomenom is caused by many reasons among which are cost and/or risk considerations, unconvincing real-world abuse scenarios and even laxness.

However, in the case of weak digital signature schemes there is also the issue of supporting old signatures. It may well be impossible to replace all old weak signatures with more secure ones, as signatures tend to proliferate beyond the control of the original signer. It seems that therefore signature verifiers will continue to accept weak – and possibly malicious – signatures for a long time to come. Unfortunately, signature verifiers have no way of knowing whether all signers have actually retired the weak scheme and whether an 'old' weak signature is really an old one or just forged to look like one.

This is exactly what we're currently seeing for MD5-based signatures in practice. MD5 was first proven to be broken in 2004 by Wang et al.[WY05], however the first truly convincing attack scenario using MD5 collisions was our construction of a rogue Certification Authority from 2008 using a more powerful attack called the chosen-prefix collision attack [SSA+09]. MD5 has been explicitly disallowed for digital signatures for Certification Authorities ever since, but it's still used by some and still supported nearly everywhere.

1.2 Flame

An example showing that the continued support for weak signature schemes leaves one vulnerable is Flame [Cry12, Kas12]. Flame is a highly advanced malware for cyberwarfare discovered in May 2012, which spread itself locally by impersonating as a properly, but illegitimately, signed Windows Update security patch. Flame's code-signing certificate was obtained by fooling Microsoft into signing an colliding and innocuous-looking certificate using an MD5-based signature algorithm. As the to-be-signed part of both certificates were carefully crafted to result in the same MD5-hash using a chosen-prefix collision attack, the MD5-based signature is valid for both certificates.

Even though Microsoft was fully aware of these severe weaknesses of MD5 and spent great effort in migrating to more secure hash functions for *new* digital signatures at least since 2008, their software continued to accept (old) MD5-based digital signatures. Also, in their efforts they overlooked their use of MD5-based signatures for licensing purposes in their Terminal Server Licensing Service up to the discovery of Flame in 2012. This, together with other unforeseen circumstances, allowed the creation of Flame's properly, but illegitimately, signed security patch that was trusted by *all* versions of the Windows [MS12a].[1]

1.3 Counter-Cryptanalysis

We introduce *counter-cryptanalysis* as a new paradigm for strengthening weak cryptographic primitives against cryptanalytic attacks by exploiting subtle, unavoidable anomalies introduced by the cryptanalytic attack. This might seem to

[1] Any license certificate produced by the Terminal Server Licensing Service could directly be used to attack Windows Vista and earlier versions, but not later versions.

be impossible for, e.g., passive cryptanalytic attacks on public and/or private key encryption schemes. But any active cryptanalytic attack that feeds carefully crafted inputs to the cryptographic primitive may thereby introduce subtle unavoidable anomalies that can be exploited to detect such attacks. In effect, counter-cryptanalysis protects against cryptanalytic attacks and thereby may prevent significant leaks or damages.

Note that in contrast to a strengthened redesign of the cryptographic primitive, applying counter-cryptanalysis does not alter the cryptographic primitive intrinsically. Thus counter-cryptanalysis can be applied transparantly in the cryptographic primitive, only altering its behaviour when a cryptanalytic attack is detected. Thereby in principle enabling the continued secure use of a weak primitive for full backwards compatibility.

1.4 Collision Attack Detection

We also introduce the first example of counter-cryptanalysis, namely the efficient detection whether any given single message has been constructed using a cryptanalytic collision attack on MD5 or SHA-1. In particular our novel techniques solves the above verifiers problem as he can now assess whether a message having a MD5-based or SHA-1-based signature is part of a forgery attack using a cryptanalytic collision attack.

Although one way to use our novel technique is to obtain, together with the MD5 or SHA-1 hash, an auxilary boolean output indicating whether a cryptanalytic attack has been detected. This auxilary boolean output can then be used by the application to decide to invalidate the signature as well as informing the user of a forged signature. Another possibility to effectively invalidate a forgery (attempt) that does not require changes at the application level is to ensure that the two colliding messages result in different outputs, e.g., by outputting the truncated SHA-256 hash or outputting a random hash value instead.

1.5 Overview

The rest of this paper is split into two parts. In Sect. 2, we first introduce our novel collision detection algorithm and apply it to both MD5 and SHA-1. Next in Sect. 3, we discuss the discoveries made by analyzing Flame's malicious certificate using our counter-cryptanalysis technique and our work towards the reconstruction of the underlying algorithms and our preliminary conclusions.

2 Detection of Cryptanalytic Collision Attacks

2.1 Brief Background on Collision Attacks

MD5 and SHA-1 are cryptographic hash functions that use the Merkle-Damgård construction in which the security of the hash function is reduced to that of a compression function that takes as input an Intermediate Hash Value IHV and

512-bit message block B. The compression starts with a working state WS_0 initialized with IHV and goes through 64 (MD5) or 80 (SHA-1) steps $t = 0, \ldots$ computing state WS_{t+1} from WS_t. Finally it outputs the sum of IHV and the last working state. A collision for a hash function is a pair of messages (M, M') that have the same hash. For any named variable X related to M, we denote by X' the same variable for M'.

The first collision attack on MD5 is due to Wang et al.[WY05] and is constructed from two sequential near-collision attacks on the compression function. Each near-collision attack starts with a given (IHV, IHV')-pair with a known difference denoted by δIHV and uses specific message block differences denoted by δB. It is based on a differential path that describes exactly how the input differences δIHV and δB propagate through the compression function, for which then a solution (B, B') with $B' = B + \delta B$ is found. As Wang et al.'s attack requires a zero δIHV before the two near-collision blocks, this type of collision attack is called an identical-prefix collision attack.

The more powerful chosen-prefix collision attack [SLdW07] can start from an arbitrary (IHV, IHV') pair. It first uses a birthday search to obtain a new (IHV_b, IHV_b') pair whose difference δIHV_b has a specific form. Then it employs a series of near-collision attacks that iteratively reduces δIHV_b to zero and thereby results in a collision.

2.2 Exploiting Cryptanalytic Necessities

The main principle of our novel technique of detecting collision attacks is to detect the last near-collision block of a collision attack and uses two key observations on the literature on MD5 and SHA-1 cryptanalysis:

- There are only a small number of possible message block differences that may lead to feasible near-collision attacks;
- All published MD5 and SHA-1 collision attacks use differential paths that at some step have no differences at all in the working state, or – in the case of MD5 – the differences $(2^{31}, 2^{31}, 2^{31}, 2^{31})$ (see [dBB93]).[2]

Due to these observations it is possible to check for collision attacks given only one message of a colliding pair of messages. First we present our basic algorithm and then prove its correctness if the message was actually constructed using a collision attack. Then we argue that the probability of a false positive is practically negligible. Lastly, we apply our algorithm to MD5 and SHA-1 specifically.

Algorithm. We present our collision detection algorithm in Alg. 2-1 that should work for any Merkle-Damgård hash function with a MD4-style compression function and in particular for MD5 and SHA-1. Our algorithm depends on a list of triples $(\delta B, i, \delta WS_i)$ for which there may exist a feasible collision attack that uses

[2] The reason for this is simple: these working state differences can be maintained at every step of the 64 steps of MD5Compress with probability at least 1/2 if not 1.

Algorithm 2-1. Last near-collision block detection

This algorithm returns `True` if a near-collision attack is detected and `False` otherwise. For a given message M, let $M_0, \ldots, M_{N-1}$ be the N message blocks that result from the padding and the splitting of M by the hash function. For $k \in \{0, \ldots, N-1\}$ do the following:

1. Let IHV_k be the intermediate hash value before the message block M_k is processed.
2. Initialize the working state WS_0 with IHV_k, compute steps $0, \ldots, S-1$ resulting in working states $WS_1, \ldots, WS_S$ and determine IHV_{k+1} using IHV_k and WS_S.
3. For each possible combination of values for message block differences δB, step i and working state differences δWS_i belonging to a feasible near-collision attack do the following:
   (a) Apply the message block differences δB to M_k to obtain M'_k.
   (b) Apply the working state differences δWS_i to WS_i to obtain WS'_i.
   (c) Compute steps $i-1, \ldots, 0$ backwards to obtain the working states $WS'_{i-1}, \ldots, WS'_0$.
   (d) Compute steps $i, \ldots, S-1$ to obtain the working states $WS'_{i+1}, \ldots, WS'_S$.
   (e) Determine IHV'_k from WS'_0 and IHV'_{k+1} from IHV'_k and WS'_S.
   (f) If $IHV'_{k+1} = IHV_{k+1}$ then (M_k, M'_k) is a near-collision block pair: return `True`
4. Return `False`

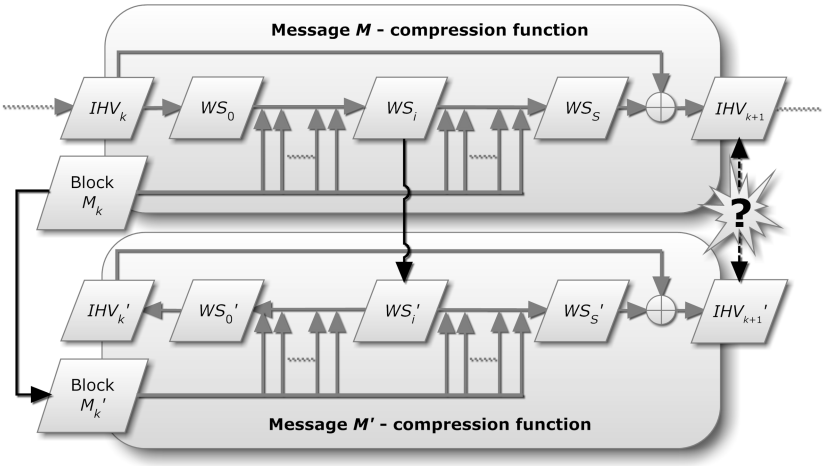
message block differences δB and always uses working state differences δWS_i at step i. For a total of C triples, the runtime-complexity of Alg. 2-1 for a message M is approximately $C+1$ times the runtime-complexity of computing the hash value of M.

Correctness. First we assume that our list of triples $(\delta B, i, \delta WS_i)$ is exhaustive for all possible feasible collision attack. For two colliding messages M and M' constructed with a feasible collision attack, let (M_k, M'_k) be the last near-collision block pair of a collision attack. Let IHV_{k+1} and IHV'_{k+1} be the intermediate hash values just after applying the compression function to M_k and M'_k in the hash value computation of M and M', respectively. Evidently, it follows that

$$IHV_{k+1} = IHV'_{k+1},$$

however as only the message M is given, the values of M', M'_k and IHV'_{k+1} are not directly known. Since M and M' were constructed with a feasible collision attack, there exists a triple $(\delta B, i, \delta WS_i)$ in our list such that $\delta B = \delta M_k$ and this last near-collision attack uses working state differences δWS_i at step i. As illustrated in Fig. 1, step 3 of Alg. 2-1 computes IHV_{k+1} and IHV'_{k+1} and tests for the telltale condition $IHV_{k+1} = IHV'_{k+1}$ in the following manner.

The hash value computation of M gives us values for the input IHV_k, output IHV_{k+1} and intermediate state WS_i (the working state before step i) of the compression function applied to IHV_k and M_k. Since we know the message block differences and the working state differences by assumption, we can determine the message block M'_k and the working state WS'_i associated with the message



If the message (upper half) was constructed using a collision attack and we correctly guess both the working state differences at a certain step and the used message block differences, then we obtain values of the internal computation of its sibling message (lower half). These are sufficient to reconstruct the entire compression function computation for this sibling block and verify whether there is a collision: $IHV'_{k+1} = IHV_{k+1}$.

Fig. 1. Detection of near-collisions

M' that collides with M. Computing steps $i+1, \ldots, S$ (where $S = 64$ for MD5 and $S = 80$ for SHA-1) of the compression function using M'_k and WS'_i, we obtain working states $WS'_{i+1}, \ldots, WS'_S$. As the step functions of MD5 and SHA-1 are reversible we can also compute working states $WS'_{i-1}, \ldots, WS'_0$. The value of IHV'_k can be derived from WS'_0 and the value of IHV'_{k+1} can be computed from IHV'_k and WS'_S.

It is clear that Alg. 2-1 on input M for the value k in step 3 for the triple $(\delta B_i, i, \delta WS_i)$ will determine the correct value IHV'_{k+1}, verify that indeed $IHV'_{k+1} = IHV_{k+1}$ and therefore return `True`. What remains is to argue that the probability of a false positive, i.e., it returns `True` for a given message M which was *not* constructed using a cryptanalytic collision attack, is negligible.

False Positives. If the given message block is not part of a near-collision block pair then the guessed WS'_i is passed through all 64 or 80 steps of the compression function to determine IHV'_{k+1}. Therefore, we argue that if there was no cryptanalytic attack then the distribution of the resulting value IHV'_{k+1} is close to the uniform distribution. Hence, the probability of a false positive, namely that $IHV'_{k+1} = IHV_{k+1}$, is thereby approximately $C \cdot 2^{-L}$ where L is the bit length of the hash value and C is the number of triples attempted in step 3 of Alg. 2-1 as before.

Interestingly, the false positive probability may be prove to be higher when there exists a differential path compatible with one of the combinations of δB, i and δWS_i that holds with probability higher than 2^{-L}. So far only one non-zero differential path is known with probability higher than 2^{-128} for MD5 (and none for SHA-1), namely the differential path consisting of differences in the most significant bit [dBB93], and this differential path is treated as a special case below and also checks for the necessary second-last near-collision block. So if nevertheless the false positive probability proves to be higher than we conjecture then this may well point towards interesting unknown cryptanalytic weaknesses.

2.3 Application to MD5

Alg. 2-1 can be directly applied to MD5. What remains is to determine possible combinations of values for message block differences δB, step i and working state differences δWS_i that belong to a feasible near-collision attack. The message block differences are additive in $\mathbb{Z}/2^{32}\mathbb{Z}$ and for each message block M_k the message block M'_k can be either $M_k + \delta B$ or $M_k - \delta B$. There are two trivial different working state differences δWS_i that can be used for MD5, namely $(0, 0, 0, 0)$ and $(2^{31}, 2^{31}, 2^{31}, 2^{31})$, written more compactly as $\underline{0}$ and $\underline{2^{31}}$.

We refer to Sect. A for a list of 222 triples $(\delta B, i, \delta WS_i)$ derived from the literature. We do not guarantee that this list forms the exhaustive list of all combinations that lead to feasible near-collision attacks. However, it should be noted that interesting message block differences have been studied extensively for nearly a decade, which has resulted in the above mentioned list. Nevertheless, other combinations from future collision attacks can easily be added to this list.

All published near-collision attacks require complex differential steps in the first round, thereby requiring a high number of bitconditions, say at least 200. E.g., the differential paths by Wang et al. require roughly 300 bitconditions [WY05]. This implies that the probability of a false positive is dominated by the general $C \cdot 2^{-L}$ term explained earlier. Hence, the probability of a false positive is estimated as $222 \cdot 2^{-128}$ and thus negligible.

However, there is a special case. Due to the *pseudo-collision* attack against MD5's compression function by den Boer and Bosselaers [dBB93], there is also a special near-collision attack not yet included in the above list. It uses zero message block differences and $\delta WS_i = \underline{2^{31}}$ for all $i \in \{0, \ldots, 64\}$. One can test for this pseudo-collision attack using $\delta B = 0$, $i = 32$ and $\delta WS_{32} = \underline{2^{31}}$. The probability of a false positive is 2^{-48} which is not negligible. However, since it requires $\delta WS_0 = \underline{2^{31}}$ and thus $IHV_{\text{in}} = \underline{2^{31}}$, this pseudo-collision attack requires at least one preceding near-collision block to form a collision attack against MD5.

This observation calls for the following modification of Alg. 2-1 for MD5 to reduce the chance of a false positive to $222 \cdot 2^{-128} \cdot 2^{-48}$ for the case $\delta B = 0$. Whenever a near-collision block is detected in step 3.(f) for the combination $\delta B = 0$, $i = 32$ and $\delta WS_{32} = \underline{2^{31}}$ and before returning `True`, perform steps 1–4 of Alg. 2-1 on the previous message block M_{k-1} using all combinations that have $\delta B \neq 0$ and using the condition $IHV'_k = IHV_k + \underline{2^{31}}$ instead of

the condition $IHV'_k = IHV_k$. If this sub-instance returns `False` then the main instance continues with the next combination of δB, i and δWS_i. Otherwise, the main instance returns `True`.

Given a message M, the average complexity to detect whether M is constructed by a collision attack against MD5 using one of the given message differences is about $222 + 1 + 1 = 224$ times the complexity of computing the MD5 hash of M. It has a conjectured false positive probability of about $222 \cdot 2^{-128}$.

2.4 Application to SHA-1

Alg. 2-1 can be directly applied to SHA-1. Note that this is possible even though no actual colliding messages for SHA-1 are known yet. What remains is to determine possible combinations of values for message block differences δB, step i and working state differences δWS_i that belong to a feasible near-collision attack.

All known attempts at a SHA-1 collision attack are based on combining local collisions according to a disturbance vector $(DV_i)_{i=0}^{79} \in (\mathbb{Z}/2^{32}\mathbb{Z})^{80}$. Furthermore, Manuel [Man11] has found that all proposed disturbance vectors can be categorized into two classes. A disturbance vector from the first class denoted by $\mathrm{I}(j,b)$ is defined by $DV_j = \ldots = DV_{j+14} = 0$ and $DV_{j+15} = 2^b$. Similarly, a disturbance vector from the second class denoted by $\mathrm{II}(j,b)$ is defined by $DV_{j+1} = DV_{j+3} = RL(2^{31}, b)$ and $DV_{j+15} = 2^b$ and $DV_{j+i} = 0$ for $i \in \{0,2,4,5,\ldots,14\}$. For both classes, the remaining $DV_0,\ldots,DV_{j-1}$ and $DV_{j+16},\ldots,DV_{79}$ are determined through the message expansion relation.

For a given disturbance vector $(DV_i)_{i=0}^{79}$, the necessary message block differences are the XOR differences $(DW_i)_{i=0}^{15} = M_k \oplus M'_k$ determined as:

$$DW_i := \bigoplus_{(j,r)\in\mathcal{R}} RL(DV_{i-j}, r), \quad \mathcal{R} = \{(0,0),(1,5),(2,0),(3,30),(4,30),(5,30)\},$$

where $DV_{-1},\ldots,DV_{-5}$ are given by the reversed message expansion relation:

$$DV_i = RR(DV_{i+16}, 1) \oplus DV_{i+2} \oplus DV_{i+8} \oplus DV_{i+13}, \quad i = -1,\ldots,-5.$$

For both disturbance vector $\mathrm{I}(j,b)$ and $\mathrm{II}(j,b)$ there are no differences at step $j+8$, hence to test for near-collision block pair using either disturbance vector we use Alg. 2-1 with the combination $(DW_t)_{t=0}^{15}$, $i = j+8$ and $\delta WS_i = (0,0,0,0,0)$.

Given the fact that no actual collisions are known yet, it is somewhat difficult to decide which triples to include. For this we refer to our recent analysis [Ste13] that seems to use the most appropriate cost function, namely one that is exact, exhaustive and takes the dependence of local collisions fully into account. However, due to the complex nature of constructing a collision attack, this cost function is not perfect as it does not accurately predict the final attack complexity. Nevertheless, we propose (a bit arbitrarily) to limit ourselves to the following 14 best disturbance vectors:

I(46,0), I(48,0), I(49,0), I(50,0), I(51,0), I(48,2), I(49,2),
II(46,0), II(50,0), II(51,0), II(52,0), II(53,0), II(54,0), II(56,0).

Similar to the case of MD5, it is always possible to add extra disturbance vectors to the above list in the future whenever it is believed it can lead to a feasible collision attack. Ignoring the first round, each disturbance vector has a probability in the order of 2^{-70} that a false positive occurs. Taking into account the complex differential steps necessary in the first round, we can safely assume that the probability of a false positive is negligible.

Given a message M, the average complexity to detect whether M is constructed by one of the above possibly feasible collision attacks against SHA-1 is about $14 + 1 = 15$ times the complexity of computing the SHA-1 hash of M. It has a conjectured false positive probability of about $14 \cdot 2^{-160}$.

3 Analyzing Flame's Chosen-Prefix Collision Attack

3.1 Background on Flame

Flame is a highly advanced malware for cyberwarfare and was discovered in May 2012 by the Iranian CERT, Kaspersky Lab and CrySyS Lab. It seemed to have targeted the Middle-East, with the most infections in Iran. We refer to the analysis of Kaspersky Lab and CrySyS Lab for more details on the functionality, purpose and origin of Flame. Here, we will focus on Flame's advanced propagation.

For a malware, it has a number of quite uncharacteristic features [Kas12, Cry12]. It has a modular design with up to 20 different plugins with different specific roles, each of which can be carefully selected prior infection. Flame is about 20MB in size as it also includes many different libraries such as for compression (zlib, bz2, ppmd), database (sqlite) and even a Lua virtual machine. Infections seem to occur with surgical precision with each target carefully selected instead of wildly spreading, which may be one of the reasons it has evaded discovery since about 2007 when Flame's main file was first seen. It spread itself locally as a valid, but illegitimate, Microsoft Windows security patch by impersonating Windows Update. Flame seems to be the first to use a chosen-prefix collision attack maliciously in the wild. Lastly, as we've discovered, it employed a yet unknown variant chosen-prefix collision attack.

3.2 Applying Counter-Cryptanalysis

On the 3rd of June 2012, Microsoft blogged that in their initial analysis of Flame they "*identified that an older cryptography algorithm could be exploited and then be used to sign code as if it originated from Microsoft*" [MS12b]. An immediate guess for this cryptically worded attack was a chosen-prefix collision attack on MD5 due to our construction of a rogue Certification Authority [SSA+09]. However, only the certificates in the chain leading to the forged signature on Flame's executable were circulating on the Internet. In particular the sibling innocuous-looking certificate actually used to obtain the forged signature on the malicious certificate was not available to directly verify a collision attack.

We were asked by enthousiasts if we could indeed verify whether the malicious certificate named 'MS' was constructed using a collision attack. We ran a proof-of-concept implementation of our technique from Sect. 2 dating from 2008 on a privately-obtained copy of the 'MS' certificate. In 0.03 seconds, it detected 4 sequential near-collision blocks and reconstructed the underlying differential paths that are given in the full version of this paper (the first differential path is also given in Sect. B). These differential paths indeed indicate a chosen-prefix collision attack that starts with a δIHV containing many bit differences that is gradually reduced to zero by the four near-collision blocks. However, very surprisingly, we discovered that these differential paths are not of the same family we used and also show characteristics that do not match those from known differential path construction methods for MD5. In the following sections we first describe the observed characteristics and then analyze and compare them.

On a historic note, the validity period of the 'MS' certificate started February 19 of 2010. Although the date can be faked, it can be argued that it does not make sense to craft this special code-signing certificate that only becomes valid in the (far) future. This puts Flame's attack years after the first identical-prefix and chosen-prefix collision attack. Also, Project HashClash [HC] released a chosen-prefix collision toolkit in 2009, which was generally expected to have been used before our discovery. Hence, it is our guess that the development of Flame's attack started before this release, but after the publication of the first chosen-prefix collision attack.

3.3 Observed Characteristics of Flame's Differential Paths

1. **Wang et al.'s message block differences.** All four differential paths are based on the same message block differences that were used by Wang et al. for the first MD5 collision [WY05]: $\delta m_4 = \delta m_{14} = 2^{31}$ and $\delta m_{11} = \pm 2^{15}$. The first and third path use $\delta m_{11} = +2^{15}$ and the second and third path use the negated form $\delta m_{11} = -2^{15}$.
2. δIHV **corrections.** The four differential paths are used in a step-wise manner to eliminate the differences in $\delta IHV = (\delta a, \delta b, \delta c, \delta d)$ resulting from the birthday search in their chosen-prefix collision. The corrections each path made are:

Block	$\delta a = \delta Q_{61}$	$\delta b = \delta Q_{64}$	$\delta c = \delta Q_{63}$	$\delta d = \delta Q_{62}$
1	[31]	[31,25,-18,-15,-12,9,1]	[31,25,-14,-12,9]	[31,25]
2	[31,5]	[-26,24,21,-14,-9,5,0]	[31,26,24,20,-9,5]	[31,-25,-9,5]
3	[31]	[30,26,-24,20,-17,15,9,-3]	[31,26,-24,-14,9]	[31,25,9]
4	[31]	[-25,14,-9,-5,3,0]	[31,-25,14,-9]	[31,-25,-9]
1+2	[5]	[31,-24,21,-18,-16,14,-12,5,2,-0]	[27,-24,20,-14,-12,5]	[-9,5]
3+4	[]	[30,24,20,-16,-14,-5,0]	[24]	[]
1+4	[]	[31,-18,-14,-12,-4,-2,-0]	[-12]	[-9]
2+3	[5]	[30,22,-20,-17,14,5,-3,0]	[27,20,-14,5]	[5]
all	[5]	[-30,21,19,17,-12,2]	[27,20,-14,-12,5]	[-9,5]

Note: we use the compact notation $[b_1, \ldots, b_n]$ for $\sum_{i=0}^{n} 2^{|b_i|}\text{sign}(b_i)$.

In comparison, the first collision attack by Wang et al. was based upon the δIHV 'correction' ([31],[31,25],[31,25],[31,25]) used in two sequential near-collision attacks, where the second uses the negated 'correction' such that the two 'corrections' cancel out.

3. bit differences in all bits of ΔQ_6, identical for blocks 1&3 and 2&4

Block	$\mathfrak{q}_6[31] \ldots \mathfrak{q}_6[0]$			
1	++----+-	---+----	-----+++	++++++++
2	+-++++++	+++-----	------+-	--+-----
3	++----+-	---+----	-----+++	++++++++
4	+-++++++	+++-----	------+-	--+-----

4. highest density of bitconditions found on $Q_4, \ldots, Q_8$. The four differential paths have, respectively, only 8, 4, 6 and 5 bits of freedom left out of those 160 bits of $Q_4, \ldots, Q_8$.

5. fixed differences $\delta Q_6, \ldots, Q_{60}$. The differential paths from the first and third block (that use the same message block differences) use the same differences $\delta Q_6, \ldots, \delta Q_{60}$. Similarly, the differential paths from the second and fourth block (that also use the same message block differences) use the same differences $\delta Q_6, \ldots, \delta Q_{60}$.

6. advanced message modification not maximized. One of the key message modifications to speed up to collision search are tunnels [Kli06]. The best and most important tunnel allows a simple message modification that does not affect all bitconditions on $Q_1, \ldots, Q_{24}$. For Flame's differential paths, this tunnel can maxime the time spent on steps 24 and onwards. This tunnel is based on flipping a bit $Q_9[b]$ with no bit condition and requires that $Q_{10}[b] = Q'_{10}[b] = 0$ and $Q_{11}[b] = Q'_{11}[b] = 1$. As shown in the table below, the near-collision blocks show a significantly lower tunnel strength than the maximal strength possible based on just the differential paths.[3]

Block	strength	max. strength	avg. strength
1	7	17	4.25
2	13	18	4.5
3	10	17	4.25
4	9	18	4.5

3.4 Differential Path Construction Analysis

So far there are two known methods for constructing a differential path for MD5. One is the our method [SLdW07] that uses a meet-in-the-middle approach. The second one is due to Mendel et al. [MRS09] that works similar to a probabilistic algorithm from coding-theory that searches for low weight code words.

The fact that all bit positions of ΔQ_6 have non-zero differences for all differential paths and that this does not help the collision search itself, indicates

[3] The 'avg. strength' is the average strength that would be observed if the extra conditions on Q_{10} and Q_{11} are each fulfilled randomly and the tunnel is *not* used.

that this choice was made with a specific purpose for the differential path construction. This choice seems to be a very bad one in combination with a method similar to Mendel et al.'s, as it is unnecessary and leads to significant increases in computational cost and number of differential path conditions. Hence, also given the uncharacteristicly high amount of differences and conditions in the first few steps, we argue that a meet-in-the-middle approach was used with a random starting path and a fixed ending path. However, from Observation 3 it is also clear that it does not use our method to construct a full differential path from the starting and ending paths: none of our differential paths have this characteristic. Evidently, it uses a yet unknown meet-in-the-middle method to construct full differential paths.

However, using the four differential paths, we can make an educated guess on how their method works. In any meet-in-the-middle approach, the lower and upper partial differential paths can be constructed independently except for four differential steps. It appears that Flame's uses a fixed differential path over steps $9, \ldots, 59$, then the meet-in-the-middle steps are $5, 6, 7, 8$. Our educated guess is that they first completed step 5 and then used an exhaustive search over steps 6, 7 and 8. With step 5 completed, the boolean function outcome modular differences δF_t for steps 6, 7 and 8 are completely determined. To complete steps 6, 7 and 8, such an exhaustive search only needs to find bit conditions that achieves these three modular differences simultaneously. The choice for non-zero differences in ΔQ_6 makes a lot of sense in this scenario, as it almost maximizes the number of choices for each $\Delta F_t[b]$ ($t = 6, 7, 8$, $b = 0, \ldots, 31$) and thus results in a higher success probability. Completing step 5 and the exhaustive search can either be done efficiently in a bit-wise approach, e.g., an adaptation of our method, or simply with a brute-force search. So far we were not able to distinguish between these two very different approaches from these differential paths.

Unfortunately, the choice to use non-zero differences in all bit positions of ΔQ_6 strictly reduces the solution space over steps $5, 6, 7, 8$ in comparison to our method and thus requires more lower/upper differential path pairs to succeed. Moreover, the choice to use a fixed upper differential path over steps $9, \ldots, 59$ implies that Flame's method requires many more lower differential paths to obtain the required amount of lower/upper path pairs. Overall, this would imply that Flame's method has higher complexity and results in differential paths with fewer degrees of freedom.

We were able to perform a somewhat simple quantitive comparison of Flame's differential paths with differential paths constructed using our publicly available HashClash toolkit [HC]. In an experiment we tried to find a replacement path for Flame's first differential path with as few bitconditions as possible. The resulting differential path is given in Tbl. C-1 and has only 266 bit conditions over $Q_1, \ldots, Q_{24}$ which are 62 fewer than the 328 bit conditions of Tbl. B-1. In another experiment we tried to construct a differential path with the HashClash toolkit in a very short amount of time, the result was an average runtime of only 15 seconds on an Intel i7-2600 CPU leading to differential paths with about 276 bit conditions, which is still 52 bit conditions fewer than Flame's path. This

experiment used only 20,000 lower and 20,000 upper partial paths leading to a total of 400,000,000 pairs. Future research might provide insights in the minimum complexity of constructing differential paths with the same characteristics of Sect. 3.3, however we have no results in this direction at this point of time.

3.5 Near-Collision Block Search

Though Observation 6 indicates the best tunnel strength is not maximized, it is also clear that this tunnel (or a slightly weaker version) is actually used as the observed tunnel strength is significantly higher than what would be observed if this tunnel was not used (cf. 'avg. strength' at Obs. 6). A reasonable guess is that they used tunnels in a dynamic manner depending on whether the necessary conditions on Q_{10} and Q_{11} were fulfilled.

Given the low number of bitconditions on $Q_{18}, \ldots, Q_{24}$ and sufficiently high tunnel strengths, we can reasonably say that the near-collision block search complexity is dominated by the cost of steps 24 up to 63. We have experimentally determined the success probability over steps 24 up to 63 for each of the near-collision blocks and these are given in Tbl. 3-1 together with lower-bounds for the average complexity in MD5 compression function calls. Note that because the inner-most loop computes at least 9 steps of the compression function, this search is well suited for massively parallel architectures in contrast to our chosen-prefix collision attack.

Table 3-1. Near-collision blocks: complexity lower-bounds

Block	estimated probability of steps 24-63	average complexity lower-bound
1	$2^{-38.8}$	$2^{36.0}$
2	$2^{-46.8}$	$2^{44.0}$
3	$2^{-33.6}$	$2^{30.8}$
4	$2^{-33.3}$	$2^{30.5}$
[WY05]	$2^{-20.5}$	$2^{17.7}$

3.6 Birthday and Reduction Procedures

The δIHV resulting from the birthday procedure can be observed as the differences for $t = -3, -2, -1, 0$ of the first differential path Tbl. B-1:

$$\delta IHV = (-2^5,\ -2^2+2^{12}-2^{17}-2^{19}-2^{21}+2^{30},\ -2^5+2^{12}+2^{14}-2^{20}-2^{27},\ -2^5+2^9).$$

Based on the available space in the certificate, our initial guess is that Flame uses 64 birthday bits over the first and last word of the IHV (matching $t = -3$ and $t = -2$ of the first path). However, this does not immediately imply that Flame's birtday search has complexity $\sqrt{\pi} \cdot 2^{32}$ MD5 compressions, as not every birthday collision is usable. In fact, the two random-looking differences have very low weights of 6 and 5 bit differences, where an uniform distribution that might be expected from an arbitrary birthday collision would actually lead to an average of about 11 bit differences each. Just aiming at such a low weight distribution

would result in a birthday complexity of about 2^{42} MD5 compressions. However, lacking a systematic family of differential path like that of [SSA+09], it is almost certain that the positions of the bit differences are also important, which further increases the birthday complexity.

Further research may provide more insights in which δIHV corrections are possible within the observed near-collision block complexities and the effect thereof on the birthday search and its complexity.

3.7 Preliminary Conclusions

Firstly, Flame's method to construct differential paths seems to be sub-optimal compared to those obtained with our public HashClash toolkit [HC].

Secondly, so far we have been able to provide a weak lower-bound for the birthday search and good lower-bounds for the near-collision block search complexities. These lower-bounds together indicate that Flame's new variant chosen-prefix collision attack likely costs more than $2^{44.3}$ MD5 compressions. How much more remains an open question as the birthday search complexity is inaccurate and it does not yet include the cost of the differential path construction. Also note that we have only one instance of a chosen-prefix collision from Flame's new variant attack, making it uncertain how close the observed near-collision block search complexities are to what can be expected *on average* with Flame's attack.

In comparison, the average complexity of our 2009 chosen-prefix collision attack for four near-collision blocks appears to be dominated by the birthday search complexity of $2^{44.55}$ MD5 compression function calls (cf. [SSA+09, Table 2] using $r = 4$ and $w = 5$). Comparing the weak lower-bound with this cost, the theoretical complexity of Flame's attack is not significantly lower than that of our attack. Nevertheless, Flame's attack might be more cost effective due to the suitabilitiy of the collision search for massively parallel architectures.

4 Conclusion

We have introduced counter-cryptanalysis as a new paradigm for strengthening weak cryptographic primitives. Also, we have presented the first example thereof, namely the efficient detection whether a given message was constructed using a cryptanalytic collision attack on MD5 and/or SHA-1. A reference implementation will be made available on project HashClash [HC].

Using our novel technique, we have analyzed Flame's malicious certificate and exposed its chosen-prefix collision attack. Our proof-of-concept collision detection implementation also reconstructed the four underlying differential paths. These differential paths reveal that Flame used a yet unknown variant chosen-prefix collision attack on MD5. We have analyzed these differential paths, working towards a reconstruction of the underlying algorithm, and found a preliminary *weak lower-bound* of $2^{44.3}$ MD5 compressions for Flame's new variant chosen-prefix collision attack.

Acknowledgements. I'm indepted to Arjen Lenstra for our insightful discussions which have led to the idea of counter-cryptanalysis.

References

[Cry12] CrySyS Lab, sKyWIper (a.k.a. Flame a.k.a. Flamer): A complex malware for targeted attacks, Laboratory of Cryptography and System Security, Budapest University of Technology and Economics (May 31, 2012)

[dBB93] den Boer, B., Bosselaers, A.: Collisions for the Compressin Function of MD5. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 293–304. Springer, Heidelberg (1994)

[HC] HashClash project webpage, `http://code.google.com/p/hashclash`

[Kas12] Kaspersky Lab, The Flame: Questions and Answers, Securelist blog (May 28, 2012)

[Kli06] Klima, V.: Tunnels in Hash Functions: MD5 Collisions Within a Minute. Cryptology ePrint Archive, Report 2006/105 (2006)

[Man11] Manuel, S.: Classification and generation of disturbance vectors for collision attacks against SHA-1. Des. Codes Cryptography 59(1-3), 247–263 (2011)

[MRS09] Mendel, F., Rechberger, C., Schläffer, M.: MD5 Is Weaker Than Weak: Attacks on Concatenated Combiners. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 144–161. Springer, Heidelberg (2009)

[MS12a] Microsoft, Flame malware collision attack explained, Security Research & Defense, Microsoft TechNet Blog (June 6, 2012)

[MS12b] Microsoft, Microsoft certification authority signing certificates added to the Untrusted Certificate Store, Security Research & Defense, Microsoft TechNet Blog (June 3, 2012)

[SLdW07] Stevens, M., Lenstra, A.K., de Weger, B.: Chosen-prefix collisions for MD5 and colliding X.509 certificates for different identities. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 1–22. Springer, Heidelberg (2007)

[SSA+09] Stevens, M., Sotirov, A., Appelbaum, J., Lenstra, A., Molnar, D., Osvik, D.A., de Weger, B.: Short Chosen-Prefix Collisions for MD5 and the Creation of a Rogue CA Certificate. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 55–69. Springer, Heidelberg (2009)

[Ste13] Stevens, M.: New Collision Attacks on SHA-1 Based on Optimal Joint Local-Collision Analysis. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 245–261. Springer, Heidelberg (2013)

[VJBT08] Vábek, J., Joščák, D., Boháček, M., Tůma, J.: A New Type of 2-Block Collisions in MD5. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT 2008. LNCS, vol. 5365, pp. 78–90. Springer, Heidelberg (2008)

[WY05] Wang, X., Yu, H.: How to Break MD5 and Other Hash Functions. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)

[XF09] Xie, T., Feng, D.: How To Find Weak Input Differences For MD5 Collision Attacks. Cryptology ePrint Archive, Report 2009/223 (2009)

[XF10] Xie, T., Feng, D.: Construct MD5 Collisions Using Just A Single Block of Message. Cryptology ePrint Archive, Report 2010/643 (2010)

[XFL08] Xie, T., Feng, D., Liu, F.: A New Collision Differential for MD5 With Its Full Differential Path. Cryptology ePrint Archive, Report 2008/230 (2008)

[XLF08] Xie, T., Liu, F., Feng, D.: Could The 1-MSB Input Difference Be The Fastest Collision Attack For MD5? Cryptology ePrint Archive, Report 2008/391 (2008)

A List of Possible Feasible MD5 Near-Collision Attacks

Used non-zero message block differences in published near-collision attacks are:

- $\delta B = \pm(\delta m_{11} = 2^{15}, \delta m_4 = \delta m_{14} = 2^{31})$ [WY05]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_2 = 2^8, \delta m_{11} = 2^{15}, \delta m_4 = \delta m_{14} = 2^{31})$ [SSA+09]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_{11} = 2^b)$ for $b \in \{0, \ldots, 30\}$ [SLdW07]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_{11} = 2^{31})$ [SLdW07]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_5 = 2^{10}, \delta m_{10} = 2^{31})$ [XF10]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_8 = 2^{31})$ [XLF08]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_6 = 2^8, \delta m_9 = \delta m_{15} = 2^{31})$ [XFL08]: $i = 37$, $\delta WS_{37} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_9 = 2^{27}, \delta m_2 = \delta m_{12} = 2^{31})$ [VJBT08]: $i = 37$, $\delta WS_{37} \in \{\underline{0}, \underline{2^{31}}\}$.

Other non-zero message block differences taken from [XF09] and [XLF08] are:

- $\delta B = \pm(\delta m_4 = 2^{20}, \delta m_7 = \delta m_{13} = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_2 = 2^8)$: $i = 37$, $\delta WS_{37} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_5 = 2^{10}, \delta m_{11} = 2^{21})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_5 = 2^{10}, \delta m_{11} = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_5 = 2^{31}, \delta m_8 = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_2 = 2^8, \delta m_{14} = 2^{31})$: $i = 37$, $\delta WS_{37} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_4 = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_5 = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_{14} = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_4 = 2^{25})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_5 = 2^{10})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_8 = 2^{25})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_{11} = 2^{21})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_{14} = 2^{16})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_4 = 2^{20})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_6 = 2^8)$: $i = 50$, $\delta WS_{50} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_9 = 2^{27})$: $i = 50$, $\delta WS_{50} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_5 = 2^{10}, \delta m_9 = 2^{27})$: $i = 37$, $\delta WS_{37} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_5 = 2^{31}, \delta m_{11} = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_8 = 2^{31}, \delta m_{11} = 2^{21})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_8 = 2^{25}, \delta m_{13} = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$.

B Flame's Differential Paths

Table B-1. Differential path of near-collision block 1

t	Bitconditions: $\mathtt{q}_t[31]\ldots\mathtt{q}_t[0]$
-3	....-.....
-2	00...... .1.1.01. ...1..+. ..-.10..
-1	110-+..1 .1.-.00. .+.+.... ..-110..
0	+-100..0 .-0+^++1 .0.+0.11 .110-+..
1	0+-++..- .-0++-+0 011-0..1 110+++..
2	+0-0-.00 .-++00+- 0-1-+.1+ 1+-0++^.
3	+010-000 .-+++0+1 +--.+^1+ -+-+++-.
4	-00-10+. .11-+-0+ +++11--0 -101-+0.
5	0-+-++-^ ^0110+1- -110+0-0 -0001+1^
6	++----+- ---+---- -----+++ ++++++++
7	111.-111 1101011. 110-1001 +0100.00
8	00+0.111 10111101 -1101100 .1110011
9	..0.1... .....-.. 0.10+... 0-....0.
10	..0^...1 ^....0.. 0^0-1... .1....+.
11	..0-...1 +....-.. .+-01... .0..^.1.
12	.1-1..^+ 1....+.. .0+0.... ....+.1.
13	.0+1..-+ 1....0.. 100....1 ....0...
14	.-+...1. .....1.. 1.+....1 ....1...
15	.0+...10 .... -.0....- ....-...
16	.1+..... .0...... ..^..... ....
17	..1..... .1....0. ^......^ ....^...
18	..0..... .+....1. ....
19	....-. ....
20	0....... .^...... ....
21	0....... ......^. ....
22	-....... ....
23	....
24	^....... ....
25–32	....
33	0....... ....
34	1....... ....
35–59	X....... ....
60	X.11110. ....
61	X.11000. .... .001.00. ....
62	X.+----. ....0.... ....
63	X.?0??+. ....--+.+-. ....
64	X......+ ++++++.. -..-.+-. .....+-.

$$\delta m_4 = \delta m_{14} = 2^{31}, \quad \delta m_{11} = 2^{15}$$

C Replacement Differential Path 1

Table C-1. Replacement differential path for near-collision block 1

t	Bitconditions: $\mathfrak{q}_t[31] \dots \mathfrak{q}_t[0]$			
-3	....	....	....	..-.....
-2	00......	.1.1.01.	...1..+.	..-.10..
-1	110-+..1	.1.-.00.	.+.+....	..-110..
0	+-100..0	.-0+^++1	.0.+0011	.110-+..
1	1+00-..-	.-.+.++.	.1.-11..	.0+10-..
2	1--.-..1	...-.+0.	...1--.^	.-0-1-..
3	.10.0.11	.1.+1+10	1.1+101+	.+0-+.^.
4	.00^+^0.	0..+00+1	1^0+-000	0-1.-1+^
5	^++++-+^	0.^+--0+	----1.+1	10-01.1+
6	-001+1-+	+.+.0-++	+1-++.0-	++0.0.+0
7	100--001	+.001+0.	-1+11.01	010...11
8	1.+00.10	-.0..010	-.0+-..0	1-....1.
9	..0-0...	0....1..	..11-...	.0....1.
10	..-1...1	+....-..	0..+....	.1....+.
11	..++..00	+....-..	...00...	......1.
12	..+1..1+	.....+..	..01....	......1.
13	..-1..-+	0....1..	1.1....1	....1...
14	..-...1.	.....1..	1.+....1	....1...
15	..+...10	....	-.1....-	....-...
16	..+.....	.0......	..1.....	....
17	..1.....	.1....0.	^......^	....^...
18	..0.....	.+....1.	....	....
19	....	......-.	....	....
20	0.......	.^......	....	....
21	0.......	......^.	....	....
22	-.......	....	....	....
23	....	....	....	....
24	^.......	....	....	....

$$\delta m_4 = \delta m_{14} = 2^{31}, \quad \delta m_{11} = -2^{15}$$

Fuming Acid and Cryptanalysis: Handy Tools for Overcoming a Digital Locking and Access Control System

Daehyun Strobel, Benedikt Driessen, Timo Kasper, Gregor Leander, David Oswald, Falk Schellenberg, and Christof Paar

Horst Görtz Institute for IT-Security
Ruhr-Universität Bochum, Germany

Abstract. We examine the widespread SimonsVoss digital locking system 3060 G2 that relies on an undisclosed, proprietary protocol to mutually authenticate transponders and locks. For assessing the security of the system, several tasks have to be performed: By decapsulating the used microcontrollers with acid and circumventing their read-out protection with UV-C light, the complete program code and data contained in door lock and transponder are extracted. As a second major step, the multi-pass challenge-response protocol and corresponding cryptographic primitives are recovered via low-level reverse-engineering. The primitives turn out to be based on DES in combination with a proprietary construction.

Our analysis pinpoints various security vulnerabilities that enable practical key-recovery attacks. We present two different approaches for unauthorizedly gaining access to installations. Firstly, an attacker having physical access to a door lock can extract a master key, allowing to mimic transponders, in altogether 30 minutes. A second, purely logical attack exploits an implementation flaw in the protocol and works solely via the wireless interface. As the only prerequisite, a valid ID of a transponder needs to be known (or guessed). After executing a few (partial) protocol runs in the vicinity of a door lock, and some seconds of computation, an adversary obtains all of the transponder's access rights.

Keywords: Access control, electronic lock, reverse-engineering, real-world attack, hardware attack, cryptanalysis, wireless door openers.

1 Introduction

Despite the fact that nowadays strong and well-analyzed cryptographic primitives are available for a large variety of applications, very weak cryptographic algorithms are still widely deployed in real products all over the world. Examples include algorithms like KeeLoq or the Crypto1 cipher used in the Mifare Classic cards. It is very surprising how big a gap between cryptographic theory on the one hand and cryptographic protocols in real products on the other hand exists and how real-life products have their security mainly based on obscurity and not on cryptographically sound protocols and primitives.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 147–164, 2013.

In this paper, we add one more interesting example to the list of widely deployed ciphers that have severe design flaws. Our hope is that the presented findings contribute to the science of building more secure wireless systems.

1.1 (Digital) Locking Systems and Wireless Technology

For many decades, purely mechanical keys and locks were the only means for securing the access to buildings, rooms, cars, and other property. Starting in the 1950s, the first Remote Keyless Entry (RKE) systems were available on the market to open doors from a distance via a Radio Frequency (RF) interface. These "fixed-code" systems provided no cryptographic protection and could easily be circumvented by means of a replay attack. In the 1980s, manufacturers started to equip cars with this type of wireless door opener on a large scale. After the number of stolen cars rose, it became clear that the new wireless comfort came at the price of reduced security and that more elaborate authentication schemes were required to prevent theft. Combining the benefits of modern wireless technology and cryptography, a new era of access control systems began: Immobilizers and more secure RKE systems were invented and today, all new cars are furnished with remote controls incorporating cryptography, while purely mechanical keys have almost vanished from the market.

Recently, a similar trend can be observed for the access control to buildings. While mechanical keys and locks are still widespread, they suffer from certain disadvantages. For example, keys can often easily be copied and if a key gets lost or stolen, all affected door locks have to be replaced. Also, mechanical locks only allow a rudimentary type of access management. The demand for a flexible assignment of keys to locks and vice versa paved the way for augmenting mechanical locks with wireless technology and replacing mechanical keys by electronic counterparts, e. g., transponders or smartcards. In case of loss or theft, administrators can simply block affected transponders in a database.

Despite all these comforts and benefits, wireless communication implies an increase in attack surface: A transponder residing in a pocket or wallet could be read out or modified without the owner taking note of it. Moreover, the transmission of data via the RF interface can be monitored from a distance. Hence, wireless access control systems require protecting the over-the-air interface with additional security measures.

1.2 Related Work

In the world of access control by electronic means, various manufacturers have been inventing their own cryptographic primitives and protocols, often with low-cost properties and established "security" by keeping the details secret. The past decade has shown that the vast majority of these schemes is flawed and that once the ciphers have been reverse-engineered and become public, they can be broken with low to modest efforts.

One of the first examples is the DST40 cipher. It is used in Texas Instrument's Digital Signature Transponder (DST) and has been reverse-engineered

in 2005 [1]: Knowing at least two challenge/response pairs, the 40-bit secret key of a corresponding transponder can be revealed by means of a brute-force attack in less than one day. Likewise, following the reverse-engineering of NXP's Mifare Classic cards [2] through analyzing the silicon die, the used Crypto1 cipher was found to be weak, relying on a state of only 48 bits. Further mathematical weaknesses of the cipher and implementations flaws, e. g., a weak random number generator, enable to reveal all secret keys and practically circumvent the protection mechanisms with a card-only attack in minutes [3–5]. The Hitag 2 transponders of the same manufacturer, widely used for car immobilizers—but also for RKE systems—were found to be flawed after the cipher became public [6]. Based on the latest results [7], their secret keys can be extracted in six minutes. Further insecure products for access control include HID Global iClass [8] and Legic Prime cards, both based on highly ineffective cryptographic measures [9].

Practically exploiting the vulnerabilities of the above products typically requires to be at least in the vicinity of the targets (cars, cards, card readers in the buildings, etc.). In contrast, attacking the RKE system KeeLoq is feasible from a larger distance: After the cipher became public, mathematical weaknesses were found [10, 11] and—after performing a side-channel attack to obtain the master key of the system—duplicating remote controls is feasible by means of eavesdropping from several hundred meters [12].

1.3 SimonsVoss Digital Locking and Access Control System 3060

One large manufacturer of digital locking systems for buildings is the Germany-based company SimonsVoss Technologies AG. SimonsVoss, the European market leader for electronic locking and access control systems [13], installed its one millionth digital locking cylinder in April 2012 and has sold more than three million corresponding transponders. The list of customers and objects secured with this technology in Europe, USA, and Asia, as listed on the official website [14], is very impressive: It includes residential buildings, tourist apartments, hospitals, universities, embassies, major banks, airports, buildings of the German armed forces and the US army, factory sites of well-known brands, police stations, stadiums, town halls, prisons, insurances, and many others.

One part of the system, termed transponder, serves as a substitute for a mechanical key. It is a battery-powered remote control that, upon pressing a button, activates the second part of the system, an electronically enhanced cylinder. The cylinder that is integrated into the door has the same dimensions as a standardized mechanical locking cylinder. If successfully activated, the door cylinder beeps twice, indicating that the lock can be opened or closed during the next few seconds (with manual force, by turning a knob that is attached to the cylinder).

The digital cylinder is also powered from batteries. In case of worn-out batteries after a few years of operating time or exceptional operating conditions[1], they

[1] The batteries need to be replaced in intervals of up to 10 years or 150,000 door openings according to the information of the manufacturer.

(a) Transponder 3064 (b) Door lock cylinder 3061 (c) Battery replacement tool (d) Opened lock knob

Fig. 1. Components of the digital locking system

can be replaced after dismantling the knob of the cylinder with the commercially available "battery replacement tool" (cf. Fig. 1c). The electronics are contained inside the knob of one side of the door cylinder, as illustrated in Fig. 1d, while the other knob is usually empty. The correct side for a battery change can be visually identified by a small, black plastic ring that is visible between the cap of the knob and the metal parts of the cylinder. This knob should be installed inside the buildings or inside the offices to prevent unauthorized access to the electronics.

The widespread digital locking system 3060 analyzed in this paper is based on a so far undisclosed, proprietary cryptographic protocol. The latest revision is termed "Generation 2" or "G2-based" system by the manufacturer. It supports up to 64,000 digital locking cylinders 3061 (cf. Fig. 1b) per installation, up to 64,000 transponders 3064 (cf. Fig. 1a) per lock, and the storage of up to 1,000 access instances on the transponder. The cost of a transponder 3064 is approx. $ 40 and that of a locking cylinder 3061 approx. $ 440.

The back-end of larger installations is realized as a software running on a standard PC, allowing to configure all door locks of an installation via a wireless link, e. g., in order to re-program the door locks. Likewise, transponders can be programmed to enable access to certain doors by means of the functionality of the back-end. In general, the "Generation 2" system enables to form a network of door cylinders, transponders, and the back-end by means of various communication techniques, e. g., through Ethernet or through multiple routers and nodes (cf. [15]). Small installations can also be configured offline, using the programming transponder 3067.

The door locks can have a permanent connection to a central server through multiple wireless access points ("WaveNet router nodes") at 868 MHz. These router nodes connect to so-called lock nodes [16] placed within the door knob. Lock nodes in turn communicate over a single wire with the circuitry responsible for opening the lock and the 25 kHz connection to the transponders. The (successful or unsuccessful) opening attempt of any transponder at any door lock in the system can be monitored and stored in logfiles. In case of an electrical power outage or when communication between door locks and back-end is interrupted, doors remain fully functional [17, p.5]. The security of the back-end and the wireless link are explicitly not analyzed in this paper. All our findings in the following are solely based on analyzing the door locks and transponders.

1.4 Contribution and Outline

The security level of access control systems relying on the obsolete ciphers mentioned in Sect. 1.2 has already been evaluated and in most cases has been found to be very low. However, to the best of our knowledge, the security of the widespread SimonsVoss digital locking system 3060 G2 and its proprietary, undisclosed schemes for encryption, authentication, and key derivation has not been publicly evaluated yet. The aim of this paper is to close this gap and analyze the security of this system.

By eavesdropping the communication between transponder and lock, it quickly became clear that the protocol is rather involved, with each protocol run consisting of 11 messages being exchanged. Moreover, as non-trivial computations are executed, extracting the details of the protocol by eavesdropping only seemed out of reach. Thus, more invasive methods were needed to advance at this step: In Sect. 2, we reverse-engineer the hardware and software of transponders and digital cylinders, by means of decapsulating chips with acid, circumventing readout protections with UV-C light, and analyzing the internals with a microscope and disassembler. As a result, the proprietary authentication scheme is disclosed in Sect. 3 and details about the proprietary cryptographic primitives and the key derivation mechanism are given in Sect. 4. Compared to various antiquated access control systems (cf. Sect. 1.2), the SimonsVoss realization at a first glance appeared to provide an adequate security level, since a slightly changed version of the Data Encryption Standard (DES) combined with a proprietary obscurity function is applied.

The next step of our work thus consists in cryptanalyzing the cipher and the protocol (cf. Sect. 5). Most surprisingly, due to a crucial flaw in the protocol, the (modified) DES can be circumvented completely. This allowed us to mainly focus on the proprietary obscurity function. After our detailed analysis it turns out that this function can be seen as a (generalized) T-function (cf. [18]), which is the key to invert (parts of) the obscurity function very efficiently.

Our work finally resulted in very practical attacks which enable opening the doors secured by the analyzed system, as illustrated in Sect. 5: An adversary possessing an ID of an valid transponder (for instance obtained by eavesdropping, exhaustive search, or reading it out from a transponder or door lock) simply has to execute a few (partial) protocol runs in the vicinity of a door lock to obtain all access rights the respective genuine transponder possesses. The attack works solely via the wireless interface, thus leaves no traces, and does not require physical access to a valid transponder or lock.

2 Reverse-Engineering

When initially analyzing the SimonsVoss System 3060, we were facing a complete black-box, i. e., had no information on the inner workings of the system. From publicly available information, little can be learned about the actual implementation. Hence, we decided to obtain the necessary knowledge for our security analysis by reverse-engineering the involved components. The Printed

Circuit Boards (PCBs) of transponder and door (cf. Fig. 2a) have a similar layout containing three main components that are involved in the authentication process: A SimonsVoss-proprietary Application Specific Integrated Circuit (ASIC) is connected to a Microchip PIC16F886 Microcontroller (μC) [19]. The third component is an external Electrically Erasable Programmable Read-Only Memory (EEPROM) controlled by the μC over an Inter-Integrated Circuit (I^2C) bus [20]. In this section, we summarize the results of reverse-engineering the functionality of these components.

2.1 Reverse-Engineering the Proprietary ASIC

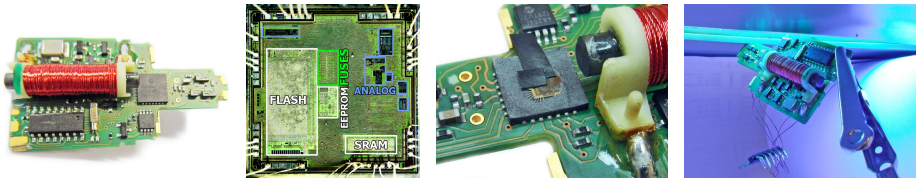
Since we could not obtain any documents regarding the functionality of the ASIC, we decided to analyze the device on the level of the silicon die. To this end, we decapsulated several ASICs using White Fuming Nitric Acid (WFNA) according to the two-step procedure as described in [21, p.10] and took high-resolution pictures of the die with an optical microscope. We found that the ASIC employs a 2 μm gate array design with a total number of 2320 transistors available for CMOS logic. In consequence, the amount of logic that can be implemented is rather limited and insufficient to, e. g., realize cryptographic algorithms. After reverse-engineering most of the digital circuits of the ASIC we found out that the main functions of the ASIC are (1) to implement functions to wake up the μC periodically and (2) to work as a (de)modulator for the RF transmission.

2.2 Reading Out the Firmware of the PIC16F886

Having found that the ASIC is not related to the security-relevant parts of the system, it can be assumed that all (cryptographic) functionality is implemented in the firmware of the Microchip PIC16F886 [19] μC. Like many common μCs, this PIC stores its firmware in an internal flash memory. Moreover, the μC contains an internal EEPROM for storing 256 bytes of user-defined data. In order to protect the firmware and the content of the EEPROM, SimonsVoss enabled the read protection fuse.

After unsuccessful attempts to clear the respective configuration bits using power glitches during the programming operation, we considered a different method: In [22], the author successfully cleared the configuration bits of a PIC18F1330, i. e., changed the state from `0` to `1` by applying Ultraviolet-C (UV-C) light in a certain angle to the decapsulated chip. The idea is that even if the fuses are covered with a (small) metal plate as a shield, UV-C light will bounce off from various parts around the metal plate and the plate itself, eventually hitting the cells storing the fuse bit. In [23], the author confirmed that the attack worked for a PIC12F683 as well.

Although the PIC16F886 comes in a relatively new type of package (QFN), Microchip did not address this issue and leaves the UV-C attack still possible. We decapsulated the μC with WFNA and used an EPROM erasure tool [24] as our UV-C source (cf. Fig. 2d). After testing various positions and angles,



(a) PCB (front) of the door part

(b) PIC16F886 annotated (analog to [23])

(c) PCB after decapsulation, memories covered with tape

(d) Exposure to UV-C light of an EPROM erasure tool

Fig. 2. PCB of the door circuitry, location of the security bits (fuses) and erasing fuses of the PIC16F886

we found that all non-volatile memories and the configuration bits were erased after an exposure of about 20 min. By applying this technique exclusively to the configuration bits, i. e., by covering the non-volatile memories with electrical isolation tape (cf. Fig. 2b and 2c), we were able to recover the complete firmware (stored in the internal Flash memory) and the contents of the internal EEPROM of several transponders and door lock μCs.

In order to disassemble and understand the code running on the μC of both the transponder and the door lock, we utilized IDA Pro [25], a tool often employed for the analysis of regular PC-software. Nevertheless, IDA Pro also includes a module for PIC μCs which greatly aided in the reverse-engineering process. In addition to performing a static analysis of the program code with IDA Pro, we also inserted debug routines into the disassembled code. This routine allows to dump the registers and the SRAM during the execution of the program on the original transponder or door PCB over an unused pin of the μC. Thus, being able to dump the memory contents, e. g., during the execution of a successful authentication protocol run, we were able to verify the results of the static analysis and to understand parts of the code that heavily depend on external input (e. g., from the external flash).

3 Authentication Keys and Protocol

In the following, we present the essential results of reverse-engineering the software running on the transponder and the lock. We focus on the keys, protocol, and the cryptographic primitives used to mutually authenticate transponders and locks.

For a successful execution of the authentication protocol, transponder and lock must be in possession of a shared secret. For this purpose, each transponder has a (unique) 128-bit long-term secret K_T. This key is computed from a 128-bit value $K_{T,\text{ext}}$ stored in the external EEPROM and a 128-bit key $K_{T,\text{int}}$ stored in the PIC's internal EEPROM as

$$K_T = K_{T,\text{ext}} \oplus K_{T,\text{int}}.$$

On the other hand, each lock stores a set of four 128-bit keys $K_{L,j}$ that are *identical for every lock in the entire installation.* Analogous to the transponder's key, one of these keys $K_{L,j}$ is the XOR of a key $K_{L,j,\text{ext}}$ stored in the external EEPROM with one 128-bit internal key $K_{L,\text{int}}$ stored in the internal EEPROM, i. e.,

$$K_{L,j} = K_{L,j,\text{ext}} \oplus K_{L,\text{int}}$$

with $0 \leq j \leq 3$. In the following, we refer to the set of the keys $K_{L,j}$ as the *system key.* Based on this key, the lock can derive any transponder key, as will be explained in the course of the protocol.

The system uses an 11-step challenge-response protocol to achieve mutual authentication between transponder and lock. A protocol run is initiated by the transponder when the central button is pressed in proximity of a lock. In the course of this authentication step, a multitude of messages is exchanged, cf. Fig. 3.

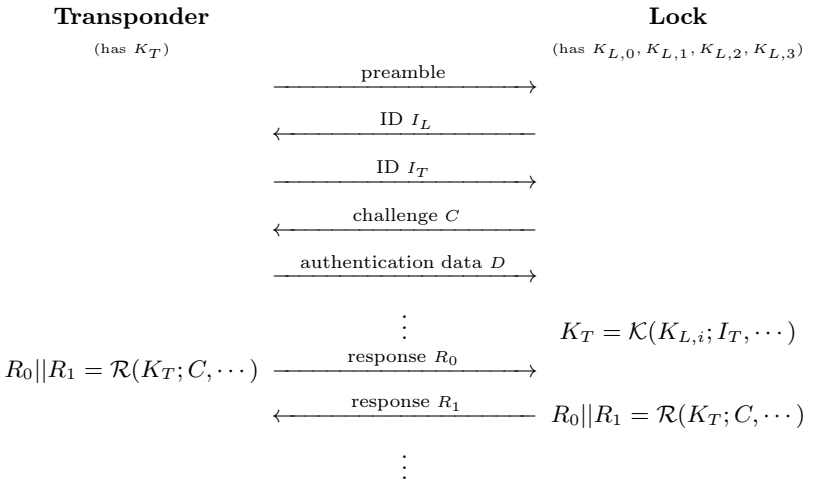


Fig. 3. Protocol for the mutual authentication between a transponder and a lock

Here we only focus on a subset of these messages that we identified to be relevant for our security analysis:

- I_L Each lock has a 24-bit ID that is transmitted to the transponder.
- I_T Each transponder has a 32-bit ID that is transmitted to the door.
- C After the ID exchange, the lock sends an 88-bit challenge C to the transponder.
- D The transponder sends 80 bits of authentication data to the lock. This includes the result of its ID verification and transponder-specific data.

In the first four steps of the protocol, most of the messages are fixed for a transponder/lock combination. Only C (and conversely the responses R_0 and R_1) change between protocol runs—and of this 88-bit value, only 40-bit are actually random. The remaining bits are either fixed or change infrequently. In answer to such a challenge, both transponder and lock derive the same 64-bit response R using the data exchanged in previous messages and the 128-bit long-term secret K_T of the transponder. We denote the function to compute the response as $\mathcal{R}$, with

$$R = R_0||R_1 = \mathcal{R}(K_T; I_L, I_T, C, D).$$

The main part of the authentication is then accomplished by exchanging the following two messages:

R_0 The transponder sends the first 32-bit half of R as the first response to the lock.

R_1 If R_0 matches the first half of R computed by the lock, it sends the second 32-bit half of R to the transponder.

As each party computes the full 64-bit output of $\mathcal{R}$, both can verify the response of the other party and mutually authenticate each other on the basis of K_T. Instead of storing the key K_T for each transponder, a lock is able to derive K_T from a transponder's ID I_T, the authentication data D, and part of the long-term system key. A key derivation function $\mathcal{K}$ is used for this purpose, i. e.,

$$K_T = \mathcal{K}(K_{L,j}; I_T, D)$$

with $0 \leq j \leq 3$.

4 Cryptographic Primitives

In the authentication protocol, the two basic functions $\mathcal{K}$ (for key derivation on the door's side) and $\mathcal{R}$ (for response computation) are used. These functions are proprietary constructions and share two building blocks we denote as $\mathcal{O}$ and $\mathcal{D}$. While it turned out that $\mathcal{D}$ is simply a modified DES [26], what we call "the obscurity function" $\mathcal{O}$ is a more intricate design, which we are describing in the following.

The $\mathcal{O}$ function takes two 128-bit inputs (a plaintext and a key) and returns a 128-bit output. Figure 4 shows the internal structure of $\mathcal{O}$. This function operates byte-wise on two registers with 16 8-bit cells. The upper registers are continuously updated while the lower registers remain constant.

To compute the output of $\mathcal{O}$ the upper x registers are initialized with the plaintext, while the lower y registers are set to the key. After that, the registers are updated successively, all in all each of the x registers is updated for 8 times, according to the following scheme: Updates start with x_0, then x_1, etc., and are computed mostly as sums of 8-bit values modulo 256. Additionally, each cell update incorporates an 8-bit chaining value z_i, which is the result of the update

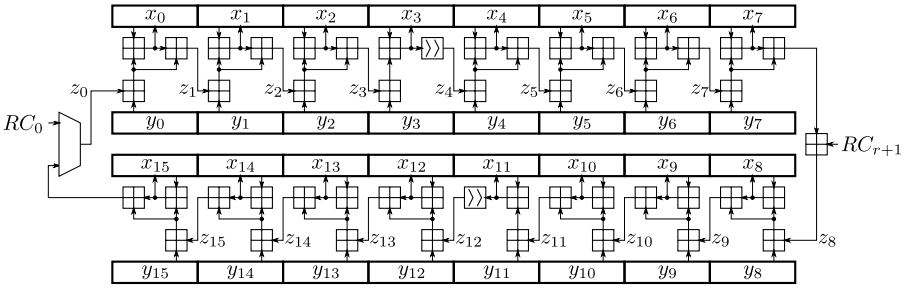


Fig. 4. Structure of the obscurity function

of the preceding cell. The update equation for the successive state x'_i from x_i is given as $x'_i = y_i + z_i + x_i \bmod 2^8$. There are basically three ways the chaining value is computed for any round r with $0 \leq r \leq 7$:

$$z_{i+1} = \begin{cases} (x_i + 2(y_i + z_i)) \bmod 2^8 & \text{if } i \in \{0, 1, ..., 14\} \setminus \{3, 7, 12\}, \\ (x_i + 2(y_i + z_i) + RC_{r+1}) \bmod 2^8 & \text{if } i = 7, \\ \big(\big(y_i + z_i \bmod 2^8\big) + x_i >> 1\big) \bmod 2^8 & \text{if } i \in \{3, 12\}. \end{cases}$$

The computation of the first value z_0 is different: Initially, it is set to RC_0 and then, for the next round of cell updates it is computed as

$$z_0 = (x_{15} + 2(y_{15} + z_{15})) \bmod 2^8.$$

Here, all RC_r with $0 \leq r \leq 8$ are 8-bit round constants, which we do not disclose at this time. The function's output is given by the contents of the x cells after 8 rounds.

4.1 $\mathcal{K}$: Key Derivation Function

In the following we describe how $\mathcal{D}$ and $\mathcal{O}$ are combined to construct the key derivation function, which is used only in the door. This function can be decomposed into three blocks, $\mathcal{D}$ and two instances of the aforementioned obscurity function $\mathcal{O}$, as illustrated in Fig. 5.

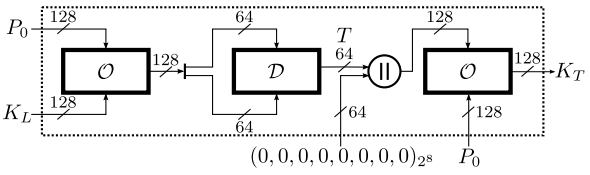


Fig. 5. Construction of the key derivation function

The construction has four inputs, one fixed to 64 zero bits and two other 128-bit inputs that are exchanged in the authentication protocol: The value P_0, used twice during key derivation, is composed of the first three bytes $I_{T,0}, I_{T,1}, I_{T,2}$ of the transponder ID I_T and the first three bytes D_0, D_1, D_2 of the authentication data D. The last of each of these three bytes is masked by a Boolean AND-operation with the fixed constant `0xC7` or `0x3F`, respectively, thus selecting only certain bits. All other bytes are filled with zeros, i. e.,

$$P_0 = (I_{T,0}, I_{T,1}, I_{T,2} \,\&\, \texttt{0xC7}, D_0, D_1, D_2 \,\&\, \texttt{0x3F}, 0, \ldots, 0)_{2^8}.$$

Only one input of $\mathcal{K}$ is secret: One of the four 128-bit keys of the system key set is selected according to the two most significant bits of the third byte of I_T. This 128-bit key is used as key K_L for the first instance of $\mathcal{O}$ to encrypt P_0. The output of this operation is split into two 64-bit halves which are used as plaintext and key for $\mathcal{D}$. The output of $\mathcal{D}$, denoted by T, is then concatenated with 64 zero bits, and the result is encrypted with $\mathcal{O}$—using P_0 as the key. The resulting 128-bit value is the transponder's key K_T, i. e.,

$$\mathcal{K}(K_L; P_0) = \mathcal{O}\Big(P_0; \mathcal{D}\Big(\mathcal{O}(K_L; P_0)_{64..127}; \mathcal{O}(K_L; P_0)_{0..63}\Big) || 0 \ldots 0\Big).$$

4.2 $\mathcal{R}$: Response Computation Function

The structure of the response computation $\mathcal{R}$ is very similar to the key derivation function $\mathcal{K}$. However, the way the building blocks are combined is different. Figure 6 shows the internal structure of $\mathcal{R}$.

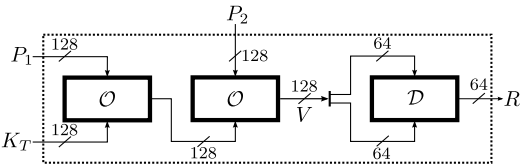


Fig. 6. Construction of the response computation function

Again two instances of the proprietary obscurity function $\mathcal{O}$ are used along with the modified DES $\mathcal{D}$. The 128-bit input P_1 to $\mathcal{R}$ is the concatenation of the challenge C and part of the authentication data D, i. e.,

$$P_1 = (c_0, c_1, ..., c_{10}, D_6, D_7, D_8, D_9, 0)_{2^8}.$$

The output of the first instance of $\mathcal{O}$ is used as key for the second iteration of $\mathcal{O}$. The 128-bit input P_2 is fixed for every transponder/lock combination and is composed of more bytes taken from the IDs of lock and door, i. e.,

$$P_2 = (I_{L,2}, I_{T,2}, I_{T,3}, D_3, D_4, D_5, 0, ..., 0)_{2^8}.$$

The output of this operation is split into two 64-bit halves, whereas the first half is used as plaintext for $\mathcal{D}$ and the second as the respective key. The two halves of the 64-bit result R form the responses R_0 and R_1 used in the protocol, i. e.,

$$\mathcal{R}(K_T; P_1, P_2) = \mathcal{D}\Big(\mathcal{O}\Big(\mathcal{O}(K_T; P_1); P_2\Big)_{64..127}; \mathcal{O}\Big(\mathcal{O}(K_T; P_1); P_2\Big)_{0..63}\Big)$$

5 Attacks

Having revealed the inner structure of the authentication protocol, we now introduce two types of attacks. The physical attacks exploit weaknesses of the platform that is used for cryptographic purposes and the basic system design. The cryptanalytical attack goes more into details of the implemented cryptographic structure, especially of the obscurity function $\mathcal{O}$.

All presented attacks have been verified at several installations of the SimonsVoss 3060 G2 system.

5.1 Physical Attacks

The attacks presented in the following are of invasive nature and assume physical access to either a transponder or a digital cylinder.

Attack 1: Invasive Cloning of a Transponder. As presented in Sect. 2, the program and internal EEPROM memories of the μC can be read after clearing the respective fuse bit with UV-C light after decapsulation. The external EEPROM is unprotected and can thus be read out trivially. Since the proprietary ASIC does not contain, e. g., a unique identifier, it is possible to copy one transponder to another by writing the respective memories. The whole process of duplicating a transponder takes less than 30 minutes, including the decapsulation of the μC. Note that the memory protection fuse bits can also be reset by performing a full chip-erase of the PIC16F886. Of course, this also deletes the program memory and the internal EEPROM and hence cannot be used for reading out the respective contents. However, this characteristic is handy for an adversary when cloning a transponder. Instead of building a custom "emulation" device, an original SimonsVoss transponder can be fully erased and then reprogrammed with the contents of the transponder to be duplicated. It should be noted that this attack does not require an adversary to understand the program code or the contents of the EEPROMs at all. In particular, this means that no knowledge of the cryptographic details of the protocol is necessary.

Attack 2: Cloning Using the System Key Analyzing the key derivation scheme described in Sect. 4, it is obvious that the system key is a single-point-of-failure. Given the system key together with a valid transponder ID, the key for the respective transponder can be generated. Apart from the system key, all other inputs to the key derivation are—by design—publicly known. Since the

system key is stored in every door lock, only one lock PCB has to be in the hands of the adversary temporarily. The PCB could be for instance removed from a door that is rarely locked or that is accessible from the outside, e. g., a door of a main entrance[2]. A "battery replacement" tool to remove the metal casing of the lock is publicly available. Note that these doors are very likely to contain the valid IDs of transponders with a system-wide validity, as required for emergencies. By invasively reading out the μC, an adversary obtains the system key and IDs of valid transponders. Similar to the previous invasive cloning of a transponder, the data can now be programmed onto an original SimonsVoss transponder that optically appears genuine. She can furthermore attempt to cover the tracks of the attack, e. g., the decapsulated μC could be replaced with a new, re-programmed PIC. Again, the complete attack can be carried out in less than 30 minutes.

Compared to the attack cloning a single transponder, the consequences of obtaining the system key are far more severe. Given the ID, any transponder in the system can be cloned without physical access to the actual transponder hardware. Since a list of IDs can be extracted from a door lock together with the system key, the problem for the adversary reduces to obtaining the PCB of one single lock.

5.2 Cryptanalytical Attack

In the following we present a non-invasive attack that allows to recover K_T in a very practical setting within a few seconds resp. minutes. The attack exploits the following properties of the system:

1. When the door computes $R_0^{(t)}||R_1^{(t)} = \mathcal{R}(K_T; C^{(t)}, \cdots)$ to verify the transponder's response, 40 bits of the internally computed DES key $V^{(t)}$ are used as part of the next challenge, i. e.,

$$\left(C_2^{(t+1)}, C_3^{(t+1)}, C_4^{(t+1)}, C_5^{(t+1)}, C_6^{(t+1)}\right)_{2^8} = \left(V_8^{(t)}, V_9^{(t)} V_{10}^{(t)}, V_{11}^{(t)}, V_{12}^{(t)}\right)_{2^8}.$$

2. Looking at one instance of $\mathcal{O}$ and the equations describing the Least Significant Bits (LSBs) of each x cell after 8 rounds, these bits are only dependent on 32 bits of the key. More specifically, the equations reveal that the LSBs of the y cells occur in non-linear combinations, while the bits next to the LSBs occur only in linear combinations.
   This observation can be generalized for any bit b in the 16 output bytes, for cases where M instances of $\mathcal{O}$ are chained (like in $\mathcal{R}$). Here, the $M + b$ lower bits per byte of the key are found in non-linear combinations in the output bits at position b, while bits at position $M + b + 1$ are only appearing in linear combinations.
   Thus, in a sense even multiple instances of $\mathcal{O}$ resemble a T function, making it significantly easier to invert them.

[2] In various real-world systems we found that the electronics of the main entrance's doors are placed facing *outwards* the building and can thus be easily accessed.

3. The output of $\mathcal{K}$, and thus every transponder key, has actually only 64 bits of entropy (the output of $\mathcal{D}$). We denote this 64-bit value as T, which—if recovered—allows to compute the full 128-bit key K_T if the corresponding P_0 is known. Note that this fact alone allows to break the scheme in practice using dedicated hardware.

Especially the first item is a weakness in the protocol. However, it is a well-known fact that obtaining "good" random numbers in (constrained) embedded systems is hard, therefore it is not entirely surprising that these seemingly random looking bits are re-used as challenge.

Lock-Only Attack. For our attack we merge three instances of $\mathcal{O}$; we focus on the part of the computation that maps the output T of the DES function in the key-derivation phase, along with the (known) values of P_0,P_1 and P_2, to the leaked input bits of the final (modified) DES function in the response computation. As mentioned above, in order to obtain the leaked data, the protocol has to be triggered once more, as the those bits leak as part of the challenge in the following protocol run.

Attacking this part allows to circumvent the (except for the small key-size) cryptographically strong, modified DES and focus on the rather weak obscurity function $\mathcal{O}$ only.

We denote by

$$F : \mathbb{F}_2^{128} \times \mathbb{F}_2^{128} \times \mathbb{F}_2^{128} \times \mathbb{F}_2^{64} \to \mathbb{F}_2^{40}$$
$$F(P_0, P_1, P_2, T) \to \mathcal{O}\Big(\mathcal{O}\Big(\mathcal{O}(P_0; T||0\ldots0); P_1\Big); P_2\Big)_{64..103} = V_L$$

the corresponding part of the commutation where V_L corresponds (up to a permutation of bits) to the leaked bits. In order to simplify the following description of our attack, we reordered the bits in such a way that the 5 first output-bits of F correspond to the 5 least significant bits of $V_i^{(t)}$, $8 \leq i \leq 12$. Similarly, the next 5 output-bits of F correspond to the second least significant bits, etc.

As mentioned above the function F inhibits a structure resembling a (slight generalization of a) T-function. More precisely, splitting the output of F in eight 5 bit chunks, i. e.,

$$F(P_0, P_1, P_2, T) = \begin{pmatrix} F_7(P_0, P_1, P_2, T) \\ \vdots \\ F_0(P_0, P_1, P_2, T) \end{pmatrix}$$

it turns out that not all F_i depend on all 64 bits of T. In fact, F_0 depends only on 30 bits of T, F_1 on 38, F_2 on 46, F_3 on 54 and F_4 on 62 bits. Even more, out of the 30 bits influencing F_0 seven bits of T enter linearly in F_0.

Making this more precise, denote for a subset $S \subset \{0, \ldots, 63\}$ by T_S, the projection of T to S, i. e., $(T_S)_i = T_i$ if $i \in S$ and $(T_S)_i = 0$ otherwise. We have

$$F_0(P_0, P_1, P_2, T) = F_0(P_0, P_1, P_2, T_{S_0^{(0)}}) + L_0(T_{S_0^{(1)}})$$

where $\mathcal{S}_0 = \mathcal{S}_0^{(0)} \cup \mathcal{S}_0^{(1)}$ with

$$\mathcal{S}_0^{(0)} = \{0, 1, 2, 8, 9, 10, 16, 17, 18, 24, 25, 26, 32, 33, 40, 41, 42, 48, 49, 50, 56, 57, 58\}$$

and

$$\mathcal{S}_0^{(1)} = \{3, 11, 19, 27, 34, 51, 59\}.$$

and the mapping L_0 is linear with rank 4.

For $i = 1..7$ we get

$$F_i(P_0, P_1, P_2, T) = F_i(P_0, P_1, P_2, T_{\mathcal{S}_{i-1}}) + L_i(T_{\mathcal{S}_i \setminus \mathcal{S}_{i-1}})$$

where

$$\mathcal{S}_1 = \mathcal{S}_0 \cup \{4, 12, 20, 28, 35, 43, 52, 60\} \quad \mathcal{S}_2 = \mathcal{S}_1 \cup \{5, 13, 21, 29, 36, 44, 53, 61\}$$
$$\mathcal{S}_3 = \mathcal{S}_2 \cup \{6, 14, 22, 30, 37, 45, 54, 62\} \quad \mathcal{S}_4 = \mathcal{S}_3 \cup \{7, 15, 23, 31, 38, 46, 55, 63\}$$
$$\mathcal{S}_5 = \mathcal{S}_4 \cup \{39, 47\} \quad \mathcal{S}_6 = \mathcal{S}_7 = \mathcal{S}_5 \cup \{\} = \{0, \ldots, 63\}$$

and the mappings L_i are linear with rank 5 for $i \in \{1, 2, 3, 4\}$, rank 2 for $i = 5$ and rank zero for $i = 6$ and $i = 7$.

Given a set of values $\left(P_0^{(i)}, P_1^{(i)}, P_2^{(i)}\right)$ along with the leaked values $V_L^{(i)} = F\left(P_0^{(i)}, P_1^{(i)}, P_2^{(i)}, T\right)$ this structure of F immediately leads to a recursive attack procedure: One first guesses the 23 key bits in $\mathcal{S}_0^{(0)}$ and sets up a system of linear equation for the bits in $\mathcal{S}_0^{(1)}$ using F_1. For each solution, one recursively sets up and solves the corresponding linear system for the bits in $\mathcal{S}_i$ using the values of F_i for $1 \leq i \leq 5$. All remaining solutions are finally validated against the values of F_6 and F_7 and a key-candidate is output when all values match known data.

Practical Results. The attack assumes that the attacker is able to obtain (or guess) a transponder's ID I_T, thus being able to construct valid P_0, P_1 and P_2. Additionally, a handful of random, consecutive, challenges C are required, which can be obtained from partial consecutive protocol runs. The following two-step procedure will reveal all relevant data:

1. Temporarily obtain a transponder that has access to a desired door, press its button and record I_T. Alternatively, since certain keys (such as emergency keys) are assigned very low IDs (e. g., `0x00000005`)—which might be true across installations—it also seems very likely that IDs with sufficient privileges can be guessed.
2. Find the most convenient door the transponder has access to—this does not have to be the targeted door—and, using some hardware which is able to communicate on the desired frequency, run these steps of the protocol:
   (a) Send the preamble, receive I_L of the door.
   (b) Send the previously obtained I_T of the transponder and receive a challenge C.

(c) Choose a suitable 80-bit string as D and send it to the door.
All interactions are recorded, then the protocol is terminated and the procedure is repeated until a handful of challenges have been collected.

As this attack does not make use of the responses R_0 and R_1 in the protocol, the challenges can be obtained by communication with the lock only (i. e., without a valid transponder). We implemented the attack in C and tested it against real data. Table 1 summarizes the running time of the attack and the number of key candidates in relation to the number of known input output-pairs $\left(P_0^{(i)}, P_1^{(i)}, P_2^{(i)}\right)$, $V_L^{(i)}$. Most importantly, all attack complexities are clearly practical and when using more than two pairs, in all our 1000 tries no false positives were detected as key candidates.

Table 1. Performance of our attack on an Intel(R) Xeon(R) CPU E5540. Note that the number of (partial) protocol runs is the number of pairs used plus one.

# Pairs Used	Aver. Running Time	Aver. # Key Candidates
2	3.36 min.	21.34
3	11.5 sec.	1
4	1.2 sec.	1
5	0.65 sec.	1

6 Conclusion

Our work shows one reason why not more existing products with weak proprietary solutions are broken: It is a challenging, time-consuming task that requires a large variety of skills; from reverse-engineering hardware and software to cryptanalytical abilities. Drawing upon *all* of these skills, we were able to perform a thorough analysis of the widely used SimonsVoss 3060 G2 access control system. We detailed methods, procedures, and results of our work to reveal all relevant physical and logical properties.

Based on the recovered details of the system, we presented attacks exploiting the found weaknesses on the hardware level. The fact that fuse bits can be erased allows to dump secret internal EEPROM contents and the firmware of the used μC. This enables the straightforward invasive cloning of *one* specific transponder. Utilizing this flaw to read out the system key from one lock compromises the long-term secret of *arbitrary* transponders. However, these attacks are heavily invasive and require access to the hardware of the system, time, and special equipment.

A more powerful, logical attack was enabled by further analysis; we found that the locking system actually uses a cryptographically strong primitive (DES) which, especially when compared to KeeLoq or Crypto1, does provide some resistance against attacks. However, here DES is used in such a way that it can be circumvented, resulting in a non-invasive attack that is even more practical than the known (non-invasive) attacks against KeeLoq or DST40. Our presented

attack is able to retrieve an arbitrary transponder key after obtaining its ID and partially running the authentication protocol with a lock only a few times. A second attack, based on the meet-in-the-middle principle, which is optimal with regard to data complexity, can be found in the full version of this paper.

In conclusion it must be said that the attacks we have presented and executed are devastating and—although initially facilitated by the ability to easily bypass fuse bits (which can be attributed to Microchip Technology)—ultimately enabled by a faulty system design. Consequently, the security of *any installation* based on the analyzed system is questionable.

We are in close contact with SimonsVoss and discussed the found vulnerabilities. As a first response, SimonsVoss is currently developing a patch for new and existing G2 systems that prohibits the here presented mathematical attacks; by changing the source of the randomness in the cylinder. This patch does not require a change of the hardware and can be deployed over-the-air via WaveNet (if the system is online, cf. Sect. 1.3) or by a programming transponder. We assume that these mathematical attacks do not apply anymore at the time this paper gets publicly available. However, we have intentionally left out certain details of the cryptographic primitives, i.e., the round constants RC_i and the modifications of the DES. Since the physical attacks are not affected by this patch and are not fixable without changing the hardware, SimonsVoss is planning to move to hardware specially designed for security applications.

References

1. Bono, S.C., Green, M., Stubblefield, A., Juels, A., Rubin, A.D., Szydlo, M.: Security analysis of a cryptographically-enabled RFID device. In: Proceedings of the 14th Conference on USENIX Security Symposium, vol. 14, USENIX Association (2005), http://www.usenix.org/events/sec05/tech/bono/bono.pdf
2. Nohl, K., Evans, D., Starbug, Plötz, H.: Reverse-Engineering a Cryptographic RFID Tag. In: van Oorschot, P.C. (ed.) USENIX Security Symposium, pp. 185–194 (2008), http://www.usenix.org/events/sec08/tech/full_papers/nohl/nohl.pdf
3. Garcia, F.D., van Rossum, P., Verdult, R., Schreur, R.W.: Wirelessly Pickpocketing a Mifare Classic Card. In: IEEE Symposium on Security and Privacy, pp. 3–15. IEEE (2009)
4. Courtois, N.: The Dark Side of Security by Obscurity - and Cloning MiFare Classic Rail and Building Passes, Anywhere, Anytime. In: SECRYPT, pp. 331–338. INSTICC (2009)
5. Kasper, T., Silbermann, M., Paar, C.: All You Can Eat or Breaking a Real-World Contactless Payment System. In: Sion, R. (ed.) FC 2010. LNCS, vol. 6052, pp. 343–350. Springer, Heidelberg (2010)
6. Courtois, N.T., O'Neil, S., Quisquater, J.-J.: Practical Algebraic Attacks on the Hitag2 Stream Cipher. In: Samarati, P., Yung, M., Martinelli, F., Ardagna, C.A. (eds.) ISC 2009. LNCS, vol. 5735, pp. 167–176. Springer, Heidelberg (2009)
7. Verdult, R., Garcia, F.D., Balasch, J.: Gone in 360 seconds: Hijacking with Hitag2. In: USENIX Security Symposium, pp. 237–252. USENIX Association (August 2012), https://www.usenix.org/system/files/conference/usenixsecurity12/sec12-final95.pdf

8. Garcia, F.D., de Koning Gans, G., Verdult, R., Meriac, M.: Dismantling iClass and iClass elite. In: Foresti, S., Yung, M., Martinelli, F. (eds.) ESORICS 2012. LNCS, vol. 7459, pp. 697–715. Springer, Heidelberg (2012)
9. Plötz, H., Nohl, K.: Legic Prime: Obscurity in Depth (2009), http://events.ccc.de/congress/2009/Fahrplan/attachments/1506_legic-slides.pdf
10. Bogdanov, A.: Attacks on the KeeLoq Block Cipher and Authentication Systems. In: Workshop on RFID Security, RFIDSec 2008 (2007), rfidsec07.etsit.uma.es/slides/papers/paper-22.pdf
11. Aerts, W., Biham, E., De Moitie, D., De Mulder, E., Dunkelman, O., Indesteege, S., Keller, N., Preneel, B., Vandenbosch, G., Verbauwhede, I.: A Practical Attack on KeeLoq, pp. 1–22. Springer, New York (2010)
12. Eisenbarth, T., Kasper, T., Moradi, A., Paar, C., Salmasizadeh, M., Shalmani, M.T.M.: On the Power of Power Analysis in the Real World: A Complete Break of the KeeLoq Code Hopping Scheme. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 203–220. Springer, Heidelberg (2008)
13. SimonsVoss Technologies AG, SimonsVoss posts record sales yet again in 2011, http://www.simons-voss.us/Record-sales-in-2011.1112.0.html?&L=6 (June 7, 2013)
14. SimonsVoss Technologies AG, References (2012), http://www.simons-voss.com/References.1163.0.html?&L=1 (June 7, 2013)
15. SimonsVoss Technologies AG, Manual for WAVENET- FUNKNETZWERK 3065 (2011) http://www.simons-voss.de/fileadmin/media/produkte/Handbuch_WaveNet_Funknetzwerk_3065_D.pdf.
16. SimonsVoss Technologies AG, Direct Networking – WaveNet network knob cap (2012), http://www.simons-voss.com/Direct-networking.631.0.html?&L=1 (June 7, 2013)
17. SimonsVoss Technologies AG, Digital Locking System 3060 (2006), http://www.simons-voss.com/fileadmin/media/produkte/english/Manual_digital-locking-system_overview_GB.pdf.
18. Klimov, A., Shamir, A.: A new class of invertible mappings. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 470–483. Springer, Heidelberg (2003)
19. Microchip Technology Inc., PIC16F882/883/884/886/887 Data Sheet, (2009), http://ww1.microchip.com/downloads/en/devicedoc/41291f.pdf.
20. NXP Semiconductors, User Manual UM10204 – I^2C-bus specification and user manual. Rev. 5 (2012), http://www.nxp.com/documents/user_manual/UM10204.pdf
21. Beck, F.: Präparationstechniken für die Fehleranalyse an integrierten Halbleiterschaltungen. VCH Verlagsgesellschaft (1988)
22. Huang, A.: Hacking the PIC 18F1320 (2005), http://www.bunniestudios.com/blog/?page_id=40 (June 7, 2013)
23. Zonenberg, A.: Microchip PIC12F683 teardown (2011), http://siliconexposed.blogspot.de/2011/03/microchip-pic12f683-teardown.html
24. Weltronik, EPROM Löschgerät, apparently the company is out of business, only found reference at http://www.weltronik.de/
25. Hex-Rays, IDA Starter Edition, http://www.hex-rays.com/products/ida/processors.shtml (June 7, 2013)
26. National Bureau of Standards, Data Encryption Standard, in FIPS-Pub.46, Federal Information Processing Standards Publication (1977)

Real Time Cryptanalysis of Bluetooth Encryption with Condition Masking*
(Extended Abstract)

Bin Zhang[1], Chao Xu[2], and Dengguo Feng[2]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, 100093, P.R. China
[2] Institute of Software, Chinese Academy of Sciences, Beijing 100190, P.R. China
{zhangbin,xuchao}@is.iscas.ac.cn

Abstract. The Bluetooth standard authorized by IEEE 802.15.1 adopts the two-level E0 stream cipher to protect short range privacy in wireless networks. The best published attack on it at Crypto 2005 requires 2^{38} on-line computations, 2^{38} off-line computations and 2^{33} memory (which amount to about 19-hour, 37-hour and 64GB storage in practice) to restore the original encryption key, given the first 24 bits of $2^{23.8}$ frames. In this paper, we describe more threatening and real time attacks against two-level E0 based on condition masking, a new cryptanalytic technique that characterizes the conditional correlation attacks on stream ciphers. The idea is to carefully choose the condition to get better tradeoffs on the time/memory/data complexity curve. It is shown that if the first 24 bits of $2^{22.7}$ frames is available, the secret key can be reliably found with 2^{27} on-line computations, $2^{21.1}$ off-line computations and 4MB memory. Our attacks have been fully implemented on one core of a single PC. It takes only a few seconds to restore the original encryption key. This is the best known-IV attack on the real Bluetooth encryption scheme so far.

Keywords: Stream ciphers, Correlation, Condition masking, Bluetooth two-level E0.

1 Introduction

Bluetooth and WiFi wireless networks are ubiquitous nowadays. The Bluetooth standard [3] adopts the two-level E0 stream cipher to protect the privacy between different devices, such as personal computers, laptops and mobile phones, that operate over a short range and at low power. Although being a long standing

* This work was supported by the National Grand Fundamental Research 973 Program of China(Grant No. 2013CB338002), the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDA06010701), IIE's Research Project on Cryptography (Grant No. Y3Z0016102) and the programs of the National Natural Science Foundation of China (Grant No. 60833008, 60603018, 61173134, 91118006, 61272476)

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 165–182, 2013.

problem in stream ciphers, the security analysis of two-level E0 is still of great practical importance, as pointed out by Prof. Preneel in [25].

Correlation attack [28] is a classical method in the cryptanalysis of stream ciphers, which exploits some statistically biased relation between the produced keystream and the output of certain underlying sequence. In the 90's, the correlation properties of combiners with memory is analyzed [9,23] in theory. Based on these correlations, for LFSR-based stream ciphers, the initial state of the target LFSR can be recovered by (fast) correlation attacks [4,5,12,13,22]. Further, in [15,16], the notion of correlation was extended to conditional correlation, that studied the linear correlation of the inputs conditioned on a given output pattern of some nonlinear function. Later at Crypto 2005 [17], the conditional correlation is assigned with a dual meaning, i.e., the correlation of the output of a function conditioned on some unknown input, called condition vector, which is uniformly distributed and is applied to analyze the security of two-level E0. Usually, the condition vector is some key related material and if a good conditional correlation exists, it is expected that the adversary will observe the biased sample sequence for the correct key and unbiased sequences for the wrong candidates. Thus, a distinguisher can be mounted to restore the secret key given a pool of sample sequences derived from the guessed values of the condition vector and some public information.

In practice, the E0 cipher is frequently re-synchronized as a two-level scheme and the keystream generated for each frame is only 2745 bits. Thus, most of the published attacks [1,6,11,14,19,26,27] that work on one impractically long frame of keystream remain the academic interest only and have little impact on the practical usage of Bluetooth encryption. Currently, a few attacks [7,8,10,17,18,24] apply to the two-level E0. The cube attack in [24] works under the unrealistic assumption that the output of LFSRs at any clock cycle is available and it is a chosen-IV attack. The best known-IV attack in [17] requires 2^{38} on-line computations, 2^{38} off-line computations and 2^{33} memory to restore the original encryption key, given the first 24 bits of $2^{23.8}$ frames in theory (while in experiments, it needs about 19-hour, 37-hour and 64GB storage, given the first 24 bits of 2^{26} frames). Note that this attack depends dominantly on the external data transfer rate between the hard disk and main memory and the pre-computation, which has to be done once for *each* key, is too time-consuming.

In this paper, we propose a new cryptanalytic technique, called condition masking, to characterize the conditional correlation attacks on stream ciphers. The attack in [17] considered the correlations conditioned on the whole condition vector, whereas we investigate the correlations only based on a subset of the condition vector. This generalizes the concept of linear mask by depicting the condition as the value selected according to a mask and studying how to choose the condition to achieve better tradeoffs between time/memory/data complexities. Our main observation is that it is of high probability that only a subset of bits in the whole condition vector determine the magnitude of the bias, e.g., in the E0 combiner, only the latest four input bits to the FSM play the most important role. The theoretical framework in [17] is refined based on this

notion and it is shown that the time/memory complexities of the attack against two-level E0 can be significantly reduced by properly choosing the condition mask.

Precisely, we first present the complete[1] formula for fast computation of unconditional correlations in the E0 combiner, and thus efficiently solve the 11-year old open problem of Golić in [10]. Second, we precisely study the conditional correlations in two-level E0 with the condition masking. The target function inherent in E0 used to compute the conditional correlation in [17] is generalized and a large class of correlations conditioned on both the linear mask and the condition mask is presented. Although the correlation conditioned on the full condition vector is maximum in the value, it is not generally optimum in the global time/memory/data complexities aspect. The time/memory complexities are closely associated with the condition. An adversary need not to guess the full condition vector and what he has to guess is determined by the condition mask he has chosen. In this way, the time/memory complexities can be considerably reduced. Third, combined with the vectorial approach[2], the data complexity of our attack can be reduced or at least kept at the same magnitude level of that in [17] as well. A necessary and sufficient condition that determines when the adversary could gain in the correlation by moving from bit to small vector (or from low-dimension to high-dimension) in the conditional correlation attack is proved in theory. Based on it, the vectors used in our attack are constructed and indeed work well to keep the data complexity as small as possible without a penalty in the time or memory complexities. As a result of all the above techniques, it is shown that if the first 24 bits of $2^{22.7}$ frames is available, the secret key can be reliably found with 2^{27} on-line computations, $2^{21.1}$ off-line computations and 4MB memory. Other choices of tradeoff parameters are also possible. Our attacks have been fully implemented in C language on one core of a single PC. Due to the small memory consumption and low time complexity, it is repeated thousands of times with randomly generated keys and IVs, while the attack in [17] is only executed 30 times for a fixed key with 2^{26} frames. On average, it takes only a few seconds to restore the original encryption key. To our knowledge, this is the best and most threatening *known-IV* attack on the real Bluetooth encryption scheme so far.

This paper is organized as follows. A full description of the two-level E0 scheme is presented in Section 2. Various correlation properties about the E0 combiner, e.g., unconditional and conditional correlations based on condition masking are studied in Section 3. Inspired by these findings, both bitwise and vector-wise key recovery attacks based on condition masking are developed in Section 4. In Section 5, the practical implementation of our attack is described with the experimental results. Finally, some conclusions are provided and future work are pointed out in Section 6.

[1] Here 'complete' means that the formula can cover all the correlated input and output linear masks.

[2] Using multiple linear approximations at the same time.

2 Description of Bluetooth Two-Level E0

The description here is according to the official specification in [3]. The size of the secret key used in two-level E0 is 128 bits and the IV is 74 bits. The core is a modification of the summation generator with 4-bit memory. Precisely, the keystream generator consists of four regularly-clocked LFSRs whose lengths are 25, 31, 33 and 39 bits, respectively (128 bits in total). Their outputs are combined by a Finite State Machine (FSM) with 4 bits memory. At each time t, the following steps are executed.

The keystream generation of E0

Parameters:
1: $B_t = (b_t^1, b_t^2, b_t^3, b_t^4) \in GF(2)^4$ denote the output bits of four LFSRs
2: $X_t \in GF(2)^4$ denotes the 4 memory bits $(c_{t-1}, c_t) = (c_{t-1}^1, c_{t-1}^0, c_t^1, c_t^0)$
3: z_t is the keystream bit
Input: X_t, B_t
5: $z_t = b_t^1 \oplus b_t^2 \oplus b_t^3 \oplus b_t^4 \oplus c_t^0$
6: $s_{t+1} = (s_{t+1}^1, s_{t+1}^0) = \lfloor \frac{b_t^1 + b_t^2 + b_t^3 + b_t^4 + 2c_t^1 + c_t^0}{2} \rfloor$
7: $c_{t+1}^0 = s_{t+1}^0 \oplus c_t^0 \oplus c_{t-1}^1 \oplus c_{t-1}^0$, $c_{t+1}^1 = s_{t+1}^1 \oplus c_t^1 \oplus c_{t-1}^0$
8: $(c_{t-1}, c_t) \leftarrow (c_t, c_{t+1})$
9: update the LFSRs

It is easy to see that the four LFSRs are equivalent to a single 128-bit LFSR whose output bit R_t is obtained by xoring the outputs of the four basic LFSRs, i.e., $R_t = b_t^1 \oplus b_t^2 \oplus b_t^3 \oplus b_t^4$ and $z_t = R_t \oplus c_t^0$.

Next, we introduce the two-level E0 scheme, as shown in Fig. 1. We refer the time instant t and t' to the context of E0 level one and level two, and denote $c_t^0, c_{t'}^0$ by $\alpha_t, \beta_{t'}$ respectively.

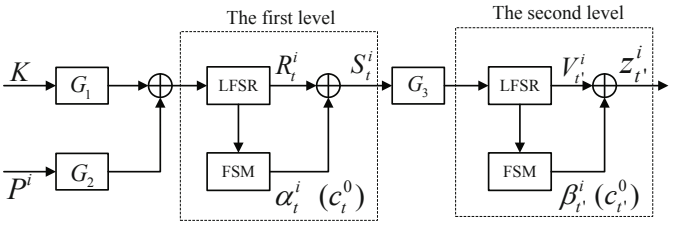


Fig. 1. Two-level E0 encryption scheme

1. *(The first level)* The LFSRs are preset to zero. Given the secret key K and some IV P^i, the LFSRs are initialized linearly as $R^i_{[-199,\cdots,-72]} = (R^i_{-199}, \cdots, R^i_{-72}) = G_1(K) \oplus G_2(P^i)$, where G_1 and G_2 are public affine transformations over $GF(2)^{128}$.[3]

[3] Hereafter we always use the superscript i to indicate the context of the i-th frame.

2. The initial 4 memory bits of FSM are all set to 0. After clocking E0 200 times, we only keep the last produced 128-bit output $S^i_{[-127,\cdots,0]} = R^i_{[-127,\cdots,0]} \oplus \alpha^i_{[-127,\cdots,0]}$. Let M be the state transmission matrix of the equivalent LFSR over $GF(2)^{128}$, i.e., $R^i_{[-127,\cdots,0]} = M^{72}(R^i_{[-199,\cdots,-72]})$. Note that because of the linear functions G_1, G_2 and M, the last 128 bits of R^i_t can be written as $R^i_{[-127,\cdots,0]} = (M^{72} \circ G_1)(K) \oplus (M^{72} \circ G_2)(P^i)$.
3. $S^i_{[-127,\cdots,0]}$ is used to initialized the four LFSRs by a byte-wise affine transformation $G_3 : GF(2)^{128} \to GF(2)^{128}$, detailed in Section 4.1 and Appendix B, this process can be expressed by $V^i_{[1,\cdots,128]} = G_3(S^i_{[-127,\cdots,0]})$.
4. *(The second level)* The FSM initial state remains the same as it was in the end of the first level. Then E0 produces the keystream $z^i_{t'} = V^i_{t'} \oplus \beta^i_{t'}$ of the i-th frame for $t' = 1, \cdots, 2745$.

3 Correlations in the Bluetooth Combiner

In this section, we will carefully study both the unconditional and conditional correlation properties of the E0 combiner.

3.1 Unconditional Linear Correlations

We first give the definition of correlation used in this paper.

Definition 1. *The correlation (or bias) of a random Boolean variable X is* $\epsilon(X) = Pr(X = 1) - Pr(X = 0)$.[4]

Let $\Omega(a, (\omega, u))$ denote the correlation $\epsilon(a \cdot s_{t+1} \oplus \omega \cdot c_t \oplus u \cdot B_t)$, where $a \in GF(2)^2$, $u \in GF(2)^4$, $\omega \in GF(2)^2$ and B_t denote the output bits of four LFSRs at time t. From Section 2, note that s_{t+1} is symmetric with respect to each b^i_t and depends only on $wt(B_t)$.[5] Our complete formula for the computation of unconditional correlations is as follows.

Theorem 2. *Let $h : (x^1, x^0) \to (x^0, x^1 \oplus x^0)$ be a permutation over $GF(2)^2$ and $\delta((a_1, u_1), \cdots, (a_{d-1}, u_{d-1}), a_d) = \epsilon(a_1 \cdot c_1 \oplus u_1 \cdot B_1 \oplus \cdots \oplus a_{d-1} \cdot c_{d-1} \oplus u_{d-1} \cdot B_{d-1} \oplus a_d \cdot c_d)$, where $a_1, \cdots, a_d \in GF(2)^2$ and $u_1, \cdots, u_{d-1} \in GF(2)^4$. If the initial state of the FSM is uniformly distributed, then we have*

$$\delta((a_1, u_1), \cdots, (a_{d-1}, u_{d-1}), a_d) = -\sum_{\omega \in GF(2)^2} \Omega(a_d, (\omega, u_{d-1})) \cdot \delta((a_1, u_1),$$
$$\cdots, (a_{d-3}, u_{d-3}), (a_{d-2} \oplus h(a_d), u_{d-2}), a_{d-1} \oplus a_d \oplus \omega).$$

Theorem 2 is a generalization of the formula in [19,20]. It can compute all the unconditional correlations of the E0 combiner without any miss, e.g., it covers all the results reported in [10].

[4] Note that in some articles, $\epsilon(X) = Pr(X = 0) - Pr(X = 1)$. The only difference is the sign of the correlation.

[5] $wt(\cdot)$ denotes the Hamming weight of a vector.

3.2 Conditional Correlations Based on Condition Masking

There are two sets of inputs to the FSM in E0 encryption scheme at time t, i.e., the four LFSR output bits $B_t = (b_t^1, b_t^2, b_t^3, b_t^4)$ and the 4 memory register bits $X_t = (c_{t-1}, c_t) \in \mathrm{GF}(2)^4$. Consider l continuous time instants and let $\gamma = (\gamma_0, \gamma_1, \cdots, \gamma_{l-1}) \in GF(2)^l$ be a linear mask with $\gamma_0 = \gamma_{l-1} = 1$. $\bar{\gamma} = (\gamma_{l-1}, \gamma_{l-2}, \cdots, \gamma_0)$ is the linear mask in reverse order. Define the inputs $\mathcal{B}_{t+1} = B_{t+1}B_{t+2}\cdots B_{t+l-2} \in \mathrm{GF}(2^{4(l-2)}), X_{t+1} = (c_t, c_{t+1}) \in \mathrm{GF}(2)^4$ and the FSM outputs $C_t = (c_t^0, \cdots, c_{t+l-1}^0)$. Then the function $h^{\gamma}_{\mathcal{B}_{t+1}} : X_{t+1} \to \gamma \cdot C_t$ conditioned on $\mathcal{B}_{t+1}$ is well defined. It is shown in [17] that given $\mathcal{B}_{t+1}$, $\gamma \cdot C_t$ is heavily biased for properly chosen linear mask γ.

Consider the function $h^{\gamma}_{\mathcal{B}_{t+1}} : X_{t+1} \to \gamma \cdot C_t$. With the knowledge of $\mathcal{B}_{t+1}$ and X_{t+1}, we can recursively compute C_t. The bias $\epsilon(h^{\gamma}_{\mathcal{B}_{t+1}})$ can be easily computed by an exhaustive search over all the possible values of X_{t+1}. For different values of $\mathcal{B}_{t+1}$, the bias $\epsilon(h^{\gamma}_{\mathcal{B}_{t+1}})$ may be different, while the mean value $E[\epsilon(h^{\gamma}_{\mathcal{B}_{t+1}})]$ is a good estimate in the attacks. The following definitions are essential in our attacks.

Definition 3. *Let ξ be an arbitrary set, given the function $f : \xi \to GF(2)^r$, the distribution D_f of $f(X)$ with $X \in \xi$ uniformly distributed is $D_f(a) = \frac{1}{|\xi|}\sum_{X\in\xi} \mathbf{1}_{f(X)=a}$, for all $a \in GF(2)^r$. As in [2], the Squared Euclidean Imbalance (SEI) of a distribution D_f is defined as $\Delta(D_f) = 2^r \sum_{a\in GF(2)^r}(D_f(a) - \frac{1}{2^r})^2$. SEI measures the distance between the target distribution and the uniform distribution.*

Specially, for $r = 1$, we have $\Delta(D_f) = \epsilon^2(D_f)$. For brevity, we use the $\epsilon(f)$, $\Delta(f)$ to represent $\epsilon(D_f)$, $\Delta(D_f)$ respectively hereafter. Similarly, $E[\Delta(h_{\mathcal{B}})]$ is used to measure the conditional correlations. Now we are ready for the definition of condition masking.

Definition 4. *Given a function $h : GF(2)^u \times GF(2)^v \to GF(2)^r$ with inputs $\mathcal{B} \in GF(2)^u$, $X \in GF(2)^v$, where $\mathcal{B}$ is the key related part and the possible condition vector. Let $\mathcal{B} = (b_0, \cdots, b_{u-1}) \in GF(2)^u$ and $\lambda = (\lambda_0, \lambda_1, \ldots, \lambda_{u-1}) \in GF(2)^u$ with $supp(\lambda) = \{0 \le i \le u-1 | \lambda_i = 1\} = \{l_1, \cdots, l_m\}$ $(l_j < l_{j+1})$. Then the shrunken vector of $\mathcal{B}$ defined by λ is $\mathcal{B}' = (b_{l_1}, \cdots, b_{l_m}) \in GF(2)^m$. Here λ is called the condition mask of $\mathcal{B}$. Further, other bits in $\mathcal{B}$ form another vector and is denoted by $\mathcal{B}^* \in GF(2)^{u-m}$, which is the complement part of $\mathcal{B}'$. We define an operator $'\backslash'$ to represent the above process and have $\mathcal{B}^* = \mathcal{B} \setminus \mathcal{B}'$.*

This definition indicates that the adversary maybe not use the full vector as the condition, but only search the correlations conditioned on a subset of $\mathcal{B}$ defined by a mask λ. In the cryptanalysis of E0, $\mathcal{B}_{t+1}$ is the key related input. Given a condition mask $\lambda = (\lambda_{t+1}, \cdots, \lambda_{t+l-2}) \in \mathrm{GF}(2)^{4(l-2)}$, where $\lambda_j \in \mathrm{GF}(2)^4$ corresponds to B_j for $j = t+1, \cdots, t+l-2$, denote the condition vector defined by λ by $\mathcal{B}'_{t+1}$ and its complement by $\mathcal{B}^*_{t+1}$ which includes the other bits. The function $h^{\gamma}_{\mathcal{B}_{t+1}}$ can now be generalized as

$$h^{\Lambda}_{\mathcal{B}'_{t+1}} : X_{t+1}, \mathcal{B}^*_{t+1} \to \gamma \cdot C_t \oplus \omega \cdot \mathcal{B}^*_{t+1}, \tag{1}$$

where $\Lambda = (\gamma, \omega)$ and $|\omega| = |\mathcal{B}^*_{t+1}|$.[6] As we can see, this function induces a large class of correlations based on both the linear mask and the condition mask.

Although the computation process of C_t is frustrated by the condition mask $\lambda \neq \mathbf{1}_u$, the bias can still be computed. For example, given $l = 4$ and $\lambda = 0x0f$,[7] we have $\mathcal{B}_{t+1} = B_{t+1}B_{t+2}$, $\mathcal{B}'_{t+1} = B_{t+2}$ and $\mathcal{B}^*_{t+1} = B_{t+1}$. We can guess B_{t+2} and compute $h^{\Lambda}_{B_{t+2}}$ for all the possible choices of B_{t+1}, X_{t+1} to get $\epsilon(h^{\Lambda}_{B_{t+2}})$. Since $\mathcal{B}_{t+1}$ is the outputs of the LFSRs, it is the key related material. In [17], the attacker guesses the full vector $\mathcal{B}_{t+1}$, while now he/she only needs to guess $\mathcal{B}'_{t+1}$, a part of $\mathcal{B}_{t+1}$, to mount the attack. This is the reason that the time/memory complexities of the attack can be significantly reduced.

Note that in the initialization phase, $\mathcal{B}_t$ at level one can be expressed as $\mathcal{B}^i_t = L_t(K) \oplus L'_t(P^i)$, where L_t and L'_t are the public linear functions. The knowledge of $\mathcal{B}^i_t$ will directly lead to the linear equations on the original key. This motivates us to study the bias $\epsilon(h^{\Lambda}_{\mathcal{B}'_{t+1}})$ defined by a certain condition mask λ. For $4 \leq l \leq 6$, we have exhaustively searched the correlations based on condition masking for all the possible condition masks on a PC. All the significant biases obtained are also verified in computer simulations working on sufficiently long output sequences. The time complexity of guessing is determined by $wt(\lambda)$. To get better time/memory complexities, we restrain ourselves to the λs satisfying $1 \leq wt(\lambda) \leq 7$. In the experiments, we have found many important masks, one is listed in the following Table 1. Table 1 is computed with $\lambda = 0x00f, \Lambda = (\gamma, \omega) = (0x1f, \mathbf{0}_{|\omega|})$. We get $E[\Delta(h_{\mathcal{B}'_{t+1}})] \approx 2^{-3.7}$, where $\mathcal{B}'_{t+1} = B_{t+3}$. The following property, shows that the more knowledge of the LFSR bits $\mathcal{B}$, the larger conditional correlation we will obtain.

Proposition 5. *Given a function f with a partial input $\mathcal{B}$ and two condition masks λ_1, λ_2, let $\mathcal{B}_1$ be the condition vector defined by λ_1 and $\mathcal{B}_2$ be the condition vector defined by λ_2. If supp$(\lambda_2) \subseteq$ supp(λ_1), then we have $E[\Delta(f_{\mathcal{B}_1})] \geq E[\Delta(f_{\mathcal{B}_2})]$, where equality holds if and only if $D_{f_{\mathcal{B}_1}}$ is independent of $\mathcal{B}_1 \setminus \mathcal{B}_2$.*

From this proposition, give a function $h : GF(2)^u \times GF(2)^v \to GF(2)^r$ with $\mathcal{B} \in GF(2)^u$, $X \in GF(2)^v$ and a condition mask λ, we have $E[\Delta(h_{\mathcal{B}})] \geq E[\Delta(h_{\mathcal{B}'})] \geq \Delta(h)$. Moreover, for a fixed condition mask λ, its maximum bias among all the

Table 1. The bias with $\Lambda = (\gamma, \omega) = (0x1f, \mathbf{0}_{|\omega|})$ and $\lambda = 0x00f$

$\epsilon(h^{\Lambda}_{\mathcal{B}'_{t+1}})$	$wt(B_{t+3})$	cardinality of B_{t+3}
0.390625	2	6
−0.390625	0, 4	2
0.0625	3	4
−0.0625	1	4

[6] $|\cdot|$ denotes the length of a vector.

[7] For brevity, we use the hexadecimal number to represent a vector.

linear masks Λ is an essential measure of it. The larger the maximum bias, the better the condition mask is. The following proposition indicates how to choose the condition mask to make the bias large. We have verified this property by searching over all the biases of $h^{\Lambda}_{\mathcal{B}'}$ for each combination of λ, γ and ω.

Proposition 6. *For $4 \le l \le 6$, let $\mathcal{B}_{t+1} = B_{t+1} \cdots B_{t+l-2} \in GF(2)^{4(l-2)}$, and $\lambda = (\lambda_{t+1}, \cdots, \lambda_{t+l-2})$, $\lambda' = (\lambda'_{t+1}, \cdots, \lambda'_{t+l-2})$ are two condition masks with $wt(\lambda) = wt(\lambda') \ge 4$, where $\lambda_i, \lambda'_i \in GF(2)^4$ correspond to B_i. If $wt(\lambda_{t+l-2}) = 4$ and $wt(\lambda'_{t+l-2}) < 4$, then $max_{\Lambda}(E[\Delta(h^{\Lambda}_{\mathcal{B}'_{t+1}})]) > max_{\Lambda'}(E[\Delta(h^{\Lambda'}_{\mathcal{B}'_{t+1}})])$,[8] expect that when $l = 4, wt(\lambda_{t+1}) = 1, wt(\lambda_{t+2}) = 4$ and $wt(\lambda'_{t+1}) = 2, wt(\lambda'_{t+2}) = 3$, in which case the maximum values are equal.*

From Proposition 6, $wt(B_{t+l-2})$ in $\mathcal{B}_t$ plays the most important role in the correlation values based on condition masking, which determines the magnitude of the corresponding bias. This fact tells us that when selecting the condition masks, we should set the highest four bits of λ to $0xf$.

4 Our Attacks with Condition Masking

In this section, our attack with the condition masking method is presented in a step-by-step manner.

4.1 Preliminaries

A statistical distinguisher can be constructed based on the biased distribution of $\gamma \cdot C_t$ in [17]. Since $\mathcal{B}_{t+1}$ is the key related material, the adversary can guess the involved key information and collect a set of sample sequences from the keystream, IVs and the guessed key value. By properly choosing the involved parameters, it is expected that with the correct key, the corresponding sample sequence is biased, while for the wrong guesses, the underlying sequence will behave like a random source.

As mentioned before, the essential problem lies in the core of the attack is to distinguish a biased sample sequence from a pool of random-like sample sequences. Since the involved sample sequences are derived from some key related information, this distinguisher can be used to identify the correct key. Formally, given a function $f : GF(2)^m \times GF(2)^{u-m} \times GF(2)^v \to GF(2)^r$ and a condition mask λ, let $f_{\mathcal{B}'}(\mathcal{B}^*, X) = f(\mathcal{B}', \mathcal{B}^*, X)$ with $\mathcal{B} = \mathcal{B}' \cup \mathcal{B}^* \in GF(2)^u$, $X \in GF(2)^v$. Here the condition vector defined by λ is $\mathcal{B}' \in GF(2)^m$ and $\mathcal{B}^* = \mathcal{B} \setminus \mathcal{B}'$. If $\mathcal{B}'$ is determined by k-bit key information, then denote by $\mathcal{B}'^{\mathcal{K}}$ the value derived when the guessing value of the key material is $\mathcal{K}$, now the problem is as follows.

Definition 7. *There are 2^k sequences of n samples with the following characteristics: one biased sequence has n samples $(f_{\mathcal{B}'^{\mathcal{K}}_i}, \mathcal{B}^{\mathcal{K}}_i)$ $(i = 1, \ldots, n)$ with the*

[8] $max_{\Lambda}(\cdot)$ is the maximum function for all Λ.

correct key $\mathcal{K}$; the other $2^k - 1$ sequences consists of n independently and uniformly distributed random variables $(Z_i^K, \mathcal{B}_i'^K)$ $(i = 1, \ldots, n)$ with the wrong keys $K \neq \mathcal{K}$. The problem is to efficiently distinguish the biased sequence from the other sequences with the minimum number n of samples.

Following [2], the minimum number n of samples for an optimal distinguisher using the unconditional correlation to effectively distinguish a sequence of n output samples of f from $(2^k - 1)$ truly random sequences of equal length is $n = \frac{4k \log 2}{\Delta(f)}$, while with the smart distinguisher in [17] based on the condition vector $\mathcal{B}$, the number of samples needed is $n_{\mathcal{B}} = \frac{4k \log 2}{E[\Delta(f_{\mathcal{B}})]}$. Since $E[\Delta(f_{\mathcal{B}})] \geq \Delta(f)$, we have $n_{\mathcal{B}} \leq n$. In our condition masking terminology, detailed in Appendix A Theorem 10, the data complexity becomes $n_{\mathcal{B}'} = \frac{4k \log 2}{E[\Delta(f_{\mathcal{B}'})]}$, and the online time complexity are $O(n_{\mathcal{B}'} + k2^{k+1})$ with pre-computation $O(k2^k)$. Besides, $|\mathcal{B}'| = k$.

We should not ignore the impact of the cardinality of the condition vector $|\mathcal{B}'| = k$ on the time/memory complexities. It is easy to see that for $\lambda \neq \mathbf{1}_u$, the cardinality k can be reduced and accordingly the time/memory complexities can be exponentially reduced. It is expected that with a careful choice of the condition mask, we can get better tradeoffs on the time/memory/data complexity curve compared to the case $\lambda = \mathbf{1}_u$. This is why we introduce the notion of condition masking to represent this phenomenon. Further, note that not all the bits in the condition vector $\mathcal{B}$ have the same influence on the correlation. In fact, some are more important than others, i.e., it is of high probability that only a subset of the condition bits can determine the magnitude of the correlation. For example, Proposition 6 shows that in the E0 FSM, only the latest four bits of $\mathcal{B}_{t+1}$ play the most important role. This is the key observation of our attack.

Next, we build the linear approximations with condition masking. The linear approximation is based on the re-initialization flaw of two-level E0 [18] detailed in the Appendix B. Precisely, we have $\bar{\gamma} \cdot (Z_{t'}^i \oplus \mathcal{L}_{t'}(K) \oplus \mathcal{L}'_{t'}(P^i)) = \bigoplus_{j=1}^{4} (\gamma \cdot C_{t_j}^i) \oplus \bar{\gamma} \cdot C_{t'}^i$, for $i = 1, \cdots, n$ and $\mathcal{L}_{t'}, \mathcal{L}'_{t'}$ are public linear functions. Here we have $t' \in \bigcup_{d=0}^{2} \{8d + 1, \cdots, 8d + 9 - l\}$. By Eq.(1), we can rewrite this equation as follow:

$$\bar{\gamma} \cdot (Z_{t'}^i \oplus \mathcal{L}_{t'}(K) \oplus \mathcal{L}'_{t'}(P^i)) \oplus \bigoplus_{j=1}^{4} (\omega \cdot \mathcal{B}_{t_j+1}^{*i}) = \bigoplus_{j=1}^{4} (\gamma \cdot C_{t_j}^i \oplus \omega \cdot \mathcal{B}_{t_j+1}^{*i}) \oplus \bar{\gamma} \cdot C_{t'}^i. \quad (2)$$

For brevity, given masks λ and Λ, we use the simplified notations $h^{\Lambda}_{\mathcal{B}'^i_{t+1}}, h^{\bar{\gamma}}$ to denote $h^{\Lambda}_{\mathcal{B}'^i_{t+1}}(\mathcal{B}_{t+1}^{*i}, X_{t+1}^i)$, $h^{\bar{\gamma}}(\mathcal{B}_{t'+1}^i, X_{t'+1}^i)$ hereafter. Since $\mathcal{B}_{t+1}^{*i} = \mathcal{B}_{t+1}^i \setminus \mathcal{B}'^i_{t+1}$ is the linear combination of K and P^i. Now Eq.(2) becomes

$$\bar{\gamma} \cdot (Z_{t'}^i \oplus \mathcal{L}_{t'}(K) \oplus \mathcal{L}'_{t'}(P^i)) \oplus \omega \cdot (L_1(K) \oplus L_2(P^i)) = \bigoplus_{j=1}^{4} h^{\Lambda}_{\mathcal{B}'^i_{t_j+1}} \oplus h^{\bar{\gamma}}, \quad (3)$$

where L_1, L_2 are public linear functions. Eq.(3) is the hybrid bitwise linear approximation based on condition masking for two-level E0, where $h^{\Lambda}_{\mathcal{B}'^i_{t_j+1}}$ are

derived from the first level and $h^{\bar{\gamma}}$ contains the unconditional correlation for the second level.

4.2 Key Recovery Attack with Bitwise Linear Approximation

From Section 3, the largest unconditional bias of h^{γ} is $\frac{25}{256}$ with $\gamma = (1,1,1,1,1)$ or $(1,0,0,0,0,1)$. To maximize the bias of Eq.(3), we choose these two γs in the second level approximation, then $|\gamma| = l = 5$ or 6. Due to the high time/memory complexities, the attack in [17] only considered $l < 6$. While in our attack, the time/memory complexities are not dependent on $|\gamma|$, they are determined by $wt(\lambda)$, thus $l = 6$ can also be used in the condition masking setting.

Given the condition mask λ and the linear masks $\Lambda = (\gamma, \omega)$, we define the following sign function to estimate the effective value of $h^{\Lambda}_{\mathcal{B}'^i_{t+1}}$ (Eq.(1)):

$$g^{\Lambda}(\mathcal{B}'^i_{t+1}) = \begin{cases} 1, & \text{if } \epsilon(h^{\Lambda}_{\mathcal{B}'^i_{t+1}}) > 0 \\ 0, & \text{if } \epsilon(h^{\Lambda}_{\mathcal{B}'^i_{t+1}}) < 0 \end{cases} \tag{4}$$

for all $\mathcal{B}'^i_{t+1} \in GF(2)^{wt(\lambda)}$ such that $\epsilon(h^{\Lambda}_{\mathcal{B}'^i_{t+1}}) \neq 0$. For brevity, let

$$\mathcal{B}^i_{\lambda} = (\mathcal{B}'^i_{t_1+1}, \mathcal{B}'^i_{t_2+1}, \mathcal{B}'^i_{t_3+1}, \mathcal{B}'^i_{t_4+1}), \mathcal{X}^i = (Y^i_{t_1+1}, Y^i_{t_2+1}, Y^i_{t_3+1}, Y^i_{t_4+1}, X^i_{t'+1}, \mathcal{B}^i_{t'+1}),$$

where $Y^i_{t_j+1} = (X^i_{t_j+1}, \mathcal{B}^*_{t_j+1})$ is the unknown input to $h^{\Lambda}_{\mathcal{B}'^i_{t_j+1}}$, and $X^i_{t'+1}, \mathcal{B}^i_{t'+1}$ are the inputs to $h^{\bar{\gamma}}$. By Eq.(3), the knowledge of the key K is contained in $\mathcal{B}^i_{\lambda}, \mathcal{L}_{t'}(K)$ and $L_1(K)$. Let the $4wt(\lambda)$ bits $K_1 = (L_{t_1}(K), L_{t_2}(K), L_{t_3}(K), L_{t_4}(K))$ contained in $\mathcal{B}^i_{\lambda}$ and $K_2 = \bar{\gamma} \cdot \mathcal{L}_{t'}(K) \oplus \omega \cdot L_1(K)$ be the subkeys. Denote by $\widetilde{\cdot}$ the guessed value of the argument. The attack is detailed as follow.

First, choose an appropriate condition mask λ and guess the subkeys $\widetilde{K_1}$ and $\widetilde{K_2}$. As P^i is known for each frame $i = 1, \cdots, n$, we can compute the condition vector $\mathcal{B}^i_{\lambda}$. Second, to distinguish the correct keys from the wrong ones, we define a mapping $\mathcal{F}^{\Lambda}_{B^i_{\lambda}}(\mathcal{X}^i)$ as follows.

$$\mathcal{F}^{\Lambda}_{\mathcal{B}^i_{\lambda}}(\mathcal{X}^i) = \begin{cases} \bigoplus_{j=1}^{4}(h^{\Lambda}_{\mathcal{B}'^i_{t+1}} \oplus g^{\Lambda}(\widetilde{\mathcal{B}'^i_{t_j+1}})) \oplus h^{\bar{\gamma}}, & \text{if } \prod_{j=1}^{4} \epsilon(h^{\Lambda}_{\mathcal{B}'^i_{t_j+1}}) \neq 0 \\ \text{a truly random bit,} & \text{otherwise} \end{cases}$$

With Eq.(4) the value of $\mathcal{F}^{\Lambda}_{B^i_{\lambda}}(\mathcal{X}^i)$ can be computed as

$$\mathcal{F}^{\Lambda}_{\mathcal{B}^i_{\lambda}}(\mathcal{X}^i) = \bar{\gamma} \cdot (Z^i_{t'} \oplus \mathcal{L}'_{t'}(P^i)) \oplus \omega \cdot L_2(P^i) \oplus \widetilde{K_2} \oplus \bigoplus_{j=1}^{4} g^{\Lambda}(\widetilde{\mathcal{B}'^i_{t_j+1}}).$$

If n frames are available, we can compute the value of $\mathcal{F}^{\Lambda}_{\mathcal{B}^i_{\lambda}}(\mathcal{X}^i)$ for each possible key by the above equation n times. With appropriate choice of Λ and λ, if K_1, K_2 are correctly guessed, then $E[\Delta(\mathcal{F}^{\Lambda}_{B^i_{\lambda}}(\mathcal{X}^i))] > 0$ and we expect $\mathcal{F}^{\Lambda}_{\mathcal{B}^i_{\lambda}}(\mathcal{X}^i)$ equals one most of the time. Otherwise, $\mathcal{F}^{\Lambda}_{\mathcal{B}^i_{\lambda}}(\mathcal{X}^i)$ is estimated by the uniform

distribution, proved in [17]. Third, we get n outputs of the source for every possible key. Submitting these samples to the distinguisher in Algorithm 1 in Appendix A, with the $k = 4wt(\lambda) + 1, u = 16(l-2), m = wt(\lambda), v = 20 + 20(l-2) - 4wt(\lambda)$ and $r = 1$, we are expected to successfully restore the correct keys.

4.3 Key Recovery Attack with the Vectorial Approach

Now we enhance the above attack by using multiple linear approximations simultaneously. Since the correlations based on condition masking are not likely to be larger than those based on the whole condition vector, we appeal to the vectorial approach to keep the data complexity as low as possible.

Assume we use s mutually independent linear approximations and let $\Gamma = (\Lambda_1, \cdots, \Lambda_s)$ and $\Gamma' = (\bar{\gamma_1}, \cdots, \bar{\gamma_s})$ denote the linear mask of these s approximations, where $\Lambda_i = (\gamma_i, \omega_i)$, and $|\gamma_1| = \cdots = |\gamma_s| = l$ with $s < l$. Especially, Λ_1 is just the linear mask used in the above bitwise attack. For brevity, let $g^\Gamma = (g^{\Lambda_1}(\mathcal{B}'^i_{t+1}), \cdots, g^{\Lambda_s}(\mathcal{B}'^i_{t+1}))$, $h^\Gamma_{\mathcal{B}'^i_{t+1}} = (h^{\Lambda_1}_{\mathcal{B}'^i_{t+1}}, \cdots, h^{\Lambda_s}_{\mathcal{B}'^i_{t+1}})$, $\mathcal{F}^\Gamma_{\mathcal{B}^i_\lambda}(\mathcal{X}^i) = (\mathcal{F}^{\Lambda_1}_{\mathcal{B}^i_\lambda}, \cdots, \mathcal{F}^{\Lambda_s}_{\mathcal{B}^i_\lambda})$ and $h^{\Gamma'} = (h^{\bar{\gamma_1}}, \cdots, h^{\bar{\gamma_s}})$. Here the first $g^{\Lambda_1}(\mathcal{B}'^i_{t+1})$ in g^Γ is determined by Eq.(4). The other bits are determined as follow: e.g., for the j-th bit, we just let it be an uniformly distributed bit if $\epsilon(h^{\Lambda_1}_{\mathcal{B}'^i_{t_j+1}}) = 0$, otherwise take 0 or 1 according to the definition in Eq.(4). Since we have found the efficient condition mask λ and linear mask $\Lambda_1 = (\gamma_1, \omega_1)$ in the bitwise attack, we extend $\mathcal{F}^{\Lambda_1}_{\mathcal{B}^i_\lambda}$ to a s-dimensional vector, i.e.,

$$\mathcal{F}^\Gamma_{\mathcal{B}^i_\lambda}(\mathcal{X}^i) = \begin{cases} \bigoplus_{j=1}^4 (h^\Gamma_{\mathcal{B}'^i_{t_j+1}} \oplus g^\Gamma(\widetilde{\mathcal{B}'^i_{t_j+1}})) \oplus h^{\Gamma'}, & \text{if } \prod_{j=1}^4 \epsilon(h^{\Lambda_1}_{\mathcal{B}'^i_{t_j+1}}) \neq 0 \\ \text{a uniformly distributed } s\text{-bit vector}, & \text{otherwise.} \end{cases}$$

In this way, we have constructed an approximation of two-level E0 in the vectorial approach. For the correct guess $\widetilde{K} = K$, we have $\mathcal{F}^\Gamma_{\mathcal{B}^i_\lambda}(\mathcal{X}^i) = \bigoplus_{j=1}^4 (h^\Gamma_{\mathcal{B}'^i_{t_j+1}} \oplus g^\Gamma(\mathcal{B}'^i_{t_j+1})) \oplus h^{\Gamma'}$ and $E[\Delta(\mathcal{F}^\Gamma_{\mathcal{B}^i_\lambda}(\mathcal{X}^i))] > 0$. For each wrong guess, the components of the s-dimensional vector $\mathcal{F}^\Gamma_{\mathcal{B}^i_\lambda}$ are uniformly distributed and we estimate the distribution $D_{\mathcal{F}^\Gamma_{\mathcal{B}^i_\lambda}(\mathcal{X}^i)}$ as a s-bit uniform distribution for all i such that $E[\Delta(\mathcal{F}^\Gamma_{\mathcal{B}^i_\lambda}(\mathcal{X}^i))] = 0$. With the appropriate choice of $\Gamma = (\Lambda_1, \cdots, \Lambda_s)$, we can get larger correlation values than those in the bitwise case. Thus, the data complexity $n_{\mathcal{B}'}$ is effectively reduced compared to the bitwise attack. Again, submitting 2^k sequences of $n_{\mathcal{B}'}$ pairs $(\mathcal{F}^\Gamma_{\mathcal{B}^i_\lambda}(\mathcal{X}^i), \widetilde{\mathcal{B}^i_\lambda})$ to Algorithm 1 in Appendix A, we can eventually recover the k-bit K.

Now we study how to choose the linear mask vector Γ. We first select a linear mask $\Lambda_1 = (\gamma_1, \omega_1)$ in the bitwise attack. Under this Λ_1, we search for other masks Λ_j $(j \geq 2)$ to maximize the total correlation. The following theorem provides a guideline for an adversary to construct the vector by depicting the criterion when he/she could gain in correlation by moving from $(s-1)$-dimension unit to s-dimension unit.

Theorem 8. *Let $\Gamma_s = (\Lambda_1, \cdots, \Lambda_s)$ be the linear mask in the s-dimensional attack with condition vector $\mathcal{B}$ and condition mask λ. Denote the joint probability by $P_{a_1 \cdots a_s} = P(h^{\Lambda_1}_{\mathcal{B}'} = a_1, \cdots, h^{\Lambda_s}_{\mathcal{B}'} = a_s)$, where $a_i \in GF(2)$ for $1 \le i \le s$. Let $P_{00\cdots00} = \frac{1}{2^s} + \xi_{00\cdots00}$, $P_{00\cdots01} = \frac{1}{2^s} + \xi_{00\cdots01}$, $\cdots$, $P_{11\cdots11} = \frac{1}{2^s} + \xi_{11\cdots11}$, where $-\frac{1}{2^s} \le \xi_j \le \frac{1}{2^s}$ for all $j \in GF(2)^s$ and $\sum_{j \in GF(2)^s} \xi_j = 0$, then $\Delta(h^{\Gamma_s}_{\mathcal{B}'}) \ge \Delta(h^{\Gamma_{s-1}}_{\mathcal{B}'})$, where the equality holds if and only if $\xi_{00\cdots00} = \xi_{00\cdots01}, \xi_{00\cdots10} = \xi_{00\cdots11}, \cdots, \xi_{11\cdots10} = \xi_{11\cdots11}$.*

This theorem indicates that high-dimensional attack will always be better than or at least be the same as low-dimensional attacks. Besides, if an adversary choose the linear masks following the rules in this theorem, then he could always gain in correlation. Further, there are some other rules when choosing Γ. First, the linear masks γ_j for $j = 1, \cdots, s$ should be linearly independent with $s \le l - 2$. Second, when the key is wrong, $\mathcal{F}^{\Lambda_j}_{\mathcal{B}^i_\lambda}$ is an uniformly distributed bit for $1 \le j \le s$ in the bitwise attack. If they are independent to each other, $\mathcal{F}^{\Gamma}_{\mathcal{B}^i_\lambda}$ follows a s-bit uniform distribution. Thus when choosing the new $\Lambda_j = (\gamma_j, \omega_j)$ $(j > 1)$, we should keep the independence among the different components $\mathcal{F}^{\Lambda_j}_{\mathcal{B}^i_\lambda}$ for $j = 1, \cdots, s$. Third, for a fixed Λ_1, when we choose some new $\Lambda = (\gamma, \omega)$ to constitute the vector, we should choose such γ that $\bar{\gamma}$ makes the unconditional correlation $\epsilon(h^{\bar{\gamma}}) = 0$ in the second level approximation, as such γ does not increase the time complexity after the extension to high-dimensional attack, which is shown in the following theorem.

Theorem 9. *Let $\Lambda_1 = (\lambda_1, \omega_1)$ be a linear mask adopted in the bitwise attack, if the jth-dimensional linear mask $\gamma_j (j \ge 2)$ makes the unconditional correlation $\epsilon(h^{\bar{\gamma}_j}) = 0$ in the approximation of the second level E0, then γ_j does not increase the time complexity when extending the $j-1$-dimensional vector to the j-dimensional vector.*

4.4 Theoretical Analysis

Now we present the theoretical justifications of our attack. We first introduce the definition of Walsh Transform and the convolution transform.

Given $f : GF(2)^k \to \mathbf{R}$, the Walsh transform $\hat{f}$ is $\hat{f}(\omega) = \sum_{x \in GF(2)^k} f(x)(-1)^{\omega \cdot x}$, and its inverse transform is $f(x) = 2^{-k} \sum_{\omega \in GF(2)^k} \hat{f}(\omega)(-1)^{\omega \cdot x}$. The convolution function of f and g is $(f \otimes g)(a) = \sum_{b \in GF(2)^k} f(b) \cdot g(a \oplus b)$ for $a \in GF(2)^k$. Further, the convolution and Walsh Transform are transformable, i.e., $\widehat{f \otimes g}(a) = \hat{f}(a) \cdot \hat{g}(a)$, for all $a \in GF(2)^k$.

To compute the convolution function $(f \otimes g)(a)$, we just perform the FWT of f and g, multiply them together and then use the inverse Walsh transform. The time and memory complexities of FWT are $O(k2^k)$ and $O(2^k)$, respectively.

By the definition of g^{Λ}, for a certain $\mathcal{B}^i_\lambda$, $g^{\Lambda}(\widetilde{\mathcal{B}'^i_{t_j+1}})$ is a fixed value not depending on $\mathcal{X}^i$. Consequently, g^{Γ} has no influence on $\Delta(\mathcal{F}^{\Gamma}_{\mathcal{B}^i_\lambda})$. We apply the Piling-up Lemma [21] and have the data complexity[9]

$$n_{\mathcal{B}'} = \frac{4k\log 2}{E[\Delta(\mathcal{F}^{\Gamma}_{\mathcal{B}^i_\lambda})]} = \frac{4k\log 2}{\Delta(h^{\Gamma'})\prod_{j=1}^{4} E[\Delta(h^{\Gamma}_{\mathcal{B}'^i_{t_j+1}})]} = \frac{4k\log 2}{\Delta(h^{\Gamma'})E^4[\Delta(h^{\Gamma}_{\mathcal{B}'^i_{t+1}})]}. \quad (5)$$

Now let us discuss the time complexity of our attack. From the expression of $\mathcal{F}^{\Gamma}_{\mathcal{B}^i_\lambda}$, it can be easily verified that this expression fulfills Theorem 10 in Appendix A, so our attack can also use the FWT to get the optimal time complexity. For all the subkeys $K = (K_1, K_2) \in GF(2)^{k-1} \times GF(2)$, where K_1 and K_2 are defined in Section 4.2, we define $\mathcal{H}, \mathcal{H}'$ as follows:

$$\mathcal{H}(K) = \sum_{i=1}^{n_{\mathcal{B}'}} \mathbf{1}_{L'_{t_1}(P^i),\cdots,L'_{t_4}(P^i)=K_1 \text{ and } (\theta_1,\cdots,\theta_s)=(K_2,1,\cdots,1)},$$

$$\mathcal{H}'(K) = \begin{cases} 0, & \text{if } \prod_{j=1}^{4}\epsilon(h^{\Lambda_1}_{K_{1,j}}) = 0 \\ \log 2^k D_{\mathcal{F}^{\Gamma}_{K_\lambda}}((K_2,1,\cdots,1)\oplus(\eta_1,\cdots,\eta_s)), & \text{otherwise} \end{cases}$$

where $\theta_j = \bar{\gamma}_j \cdot (Z^i_{t'} \oplus \mathcal{L}'_{t'}(P^i)) \oplus \omega_j \cdot L_2(P^i)$ and $\eta_j = \bigoplus_{i=1}^{4} g^{\Lambda_j}(K_{1,i})$ for $j = 1,\cdots,s$. In Algorithm 1 in Appendix A, the grade $G(K)$ is a simple convolution between $\mathcal{H}$ and $\mathcal{H}'$ (also in [17]), thus we have $G(K) = \frac{1}{2^k}\widehat{\mathcal{H}''(K)}$ where $\mathcal{H}''(K) = \widehat{\mathcal{H}}(K)\cdot\widehat{\mathcal{H}'}(K)$. Note that $\widehat{\mathcal{H}'}$ can be pre-computed in time $O(k\cdot 2^k)$ and $O(2^k)$ memory. The preparation of $\mathcal{H}$ needs $O(n_{\mathcal{B}'})$ online computation. $\widehat{\mathcal{H}}$ and $\widehat{\mathcal{H}''}$ need twice of FWT with time complexity $O(k\cdot 2^{k+1})$ and $O(2^{k+1})$ memory. Therefore, the total time complexity is $O(n_{\mathcal{B}'} + k\cdot 2^{k+1})$.

To get the optimal performance of our attack, we should carefully choose the parameters Γ and λ in the linear approximations. The experiments show that there are many large correlations based on condition masking that can be used in our attack. For example, for a condition mask $\lambda = 0x00f$, we choose 3 linear masks in the following Table 2, the experimental results show $\Delta(h^{\Gamma}_{\mathcal{B}'_{t+1}}) \approx 2^{-2.6}$, where $\Gamma = ((0x1f,\mathbf{0}),(0x1d,\mathbf{0}),(0x15,0x1))$. And $\Delta(h^{\Gamma'}) \approx 2^{-6.7}$, so we conclude from Eq.(5) that the data complexity is $n_{\mathcal{B}'} \approx 2^{22.7}$. In this example, we can recover the $k = 17$-bit subkey. Let us look at the time complexity in this

Table 2. Example: $\lambda = 0x00f$

λ	γ	ω	$E[\Delta(h^{\Lambda}_{\mathcal{B}'_{t+1}})]$
$0x00f$	$(1,1,1,1,1)$	0	$2^{-3.7}$
	$(1,1,1,0,1)$	0	$2^{-3.7}$
	$(1,0,1,0,1)$	$0x1$	$2^{-7.6}$

[9] $E[\Delta(h^{\Gamma}_{\mathcal{B}'^i_{t+1}})]$ dose not depend on t.

case. The pre-computation of $\widehat{H'}$ is $17 \cdot 2^{17}$, and we need time $2 \cdot 17 \cdot 2^{17} \approx 2^{21.1}$ to compute $\widehat{\mathcal{H}}, \widehat{\mathcal{H}''}$, and time $n_{\mathcal{B}'} = 2^{22.7}$ to compute $\mathcal{H}$, so the total time is $2^{22.7} + 2^{21.1}$.

5 Practical Implementation

Our attacks have been fully implemented on one core of a single PC, running with Windows 7, Intel Core 2 Q9400 2.66GHz and 4GB RAM. In general, the experimental results match the theoretical analysis quite well. We present the details as follows.

We choose the condition mask $\lambda = 0x00f$ and $\gamma_1 = 0x1f, \omega_1 = \mathbf{0}, \gamma_2 = 0x1d, \omega_2 = \mathbf{0}, t' = 1, n_{\mathcal{B}'} = 2^{24}$ (slightly more than the theoretical estimate $2^{23.1}$) in the experiments. The condition bits $\mathcal{B}'^i_{t+1} = B^i_{t+3}$. We first collect $n_{\mathcal{B}'}$ frames for a random key and store them in a binary file. It takes about 4 minutes and 80MB to fulfill this task. With these samples, we run Algorithm 1 in Appendix A to recover the key. The pre-computation of $\mathcal{H}'$ and $\widehat{\mathcal{H}'}$ needs about one second and the results are stored in a 4MB table in RAM, not on the hard disk. Computing $\mathcal{H}, \widehat{\mathcal{H}}, \mathcal{H}'', \widehat{\mathcal{H}''}$ in total takes about 2 seconds. Compared with the 37 hours and 64GB table in [17], our attack can be easily carried out in real time on a single PC.

Our attack is repeated 6000 times with different randomly generated keys and IVs. In our experiments, the right key does not always rank first. The reason is that when our guess is wrong, the distribution of $g^{\Lambda}(B^i_{t+3})$ does not behave exactly as the uniform distribution from the Table 1. We take the first 256 candidates in the list as the possible keys for each run (corresponding to the 256 key candidates equivalent to each other in the experiment of [17]). There is only one correct key in their equivalent key candidates, thus they also need to test these 256 equivalent key candidates to recover the right key. The success probability of our attack is about 38.6% in this case, which can be raised very high by running it several times or by taking more candidates in the rank list. Note that in [17], the experiments are only carried out in the basic bitwise level with 2^{26} frames and repeated 30 times for a fixed key. If the key is changed, the precomputation of the attack in [17] has to be done again. This fact greatly weakens the practical effect of their attack.

During the experiments, we also found many other different condition masks that can improve the attack in [17], some of which are listed in Table 3. The detailed description of one run of our attack can be found in the full version of the paper.

Table 3. The complexities of our attack with different condition masks

mask	$(\gamma_1, \cdots, \gamma_s)$	$(\omega_1, \cdots, \omega_s)$	*Precom*	*Time*	*Frames*	*Memory*
$0x00f$	$(1f, 1d)$	$(0, 0)$	$2^{21.1}$	2^{27}	$2^{23.1}$	2^{17}
$0x00f$	$(1f, 1d, 15)$	$(0, 0, 1)$	$2^{21.1}$	2^{27}	$2^{22.7}$	2^{17}
$0x101f$	$(21, 23, 31, 35)$	$(0, 0, 0, 0)$	$2^{29.6}$	$2^{30.6}$	$2^{21.4}$	2^{25}
$0x007f$	$(21, 23, 33, 37)$	$(0, 0, 0, 0)$	$2^{33.9}$	$2^{34.9}$	$2^{20.2}$	2^{29}

6 Conclusions

In this paper, we have introduced a new cryptanalytic technique, called condition masking, to characterize the conditional correlation attacks on stream ciphers. Based on this new concept, we have investigated the conditional correlations of the two-level E0 scheme and found many useful conditional correlations for the first time. Combined these correlations with the vectorial approach, we studied the practical security of two-level E0 and developed the best and most threatening *known-IV* attack on the real Bluetooth encryption scheme so far. Our attacks have been fully implemented in C code on one core of a single PC and are repeated thousands of times with randomly generated keys and IVs. On average, it takes only a few seconds to restore the original encryption key. This clearly demonstrates the superiority of our new method. We believe our new method is generic and applicable to other stream and block ciphers as well. It is our future work to study the practical ciphertext-only attack on the real Bluetooth encryption scheme using the condition masking method. Table 4 gives a comparison of our attacks with the best previous attacks on two-level E0.

Table 4. Comparison of our attacks with the previous attacks on two-level E0

Attack	*Precom*	*Time*	*Frames*	*Memory*
[7]	-	2^{73}	-	2^{51}
[8]	2^{80}	2^{65}	2	2^{80}
[10]	2^{80}	2^{70}	45	2^{80}
[18]	-	2^{40}	2^{35}	2^{35}
[17]	2^{38}	2^{38}	$2^{23.8}$	2^{33}
Ours	$2^{21.1}$	2^{27}	$2^{22.7}$	2^{17}
Ours	$2^{29.6}$	$2^{30.6}$	$2^{21.4}$	2^{25}
Ours	$2^{33.9}$	$2^{34.9}$	$2^{20.2}$	2^{29}

References

1. Armknecht, F., Krause, M.: Algebraic attacks on combiners with memory. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 162–175. Springer, Heidelberg (2003)
2. Baignères, T., Junod, P., Vaudenay, S.: How far can we go beyond linear cryptanalysis? In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 432–450. Springer, Heidelberg (2004)
3. SIG Bluetooth. Specification of the bluetooth system. volume 4.0 (2010)
4. Canteaut, A., Trabbia, M.: Improved fast correlation attacks using parity-check equations of weight 4 and 5. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 573–588. Springer, Heidelberg (2000)
5. Chose, P., Joux, A., Mitton, M.: Fast correlation attacks: An algorithmic point of view. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 209–221. Springer, Heidelberg (2002)

6. Courtois, N.T.: Fast algebraic attacks on stream ciphers with linear feedback. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 176–194. Springer, Heidelberg (2003)
7. Fluhrer, S.R., Lucks, S.: Analysis of the E_0 encryption system. In: Vaudenay, S., Youssef, A.M. (eds.) SAC 2001. LNCS, vol. 2259, pp. 38–48. Springer, Heidelberg (2001)
8. Fluhrer, S.R., Cisco Systems Inc.: Improved key recovery of level 1 of the bluetooth encryption system. Cambridge University Press (2002), http://eprint.iacr.org/2002/068
9. Golić, J.: Correlation properties of a general binary combiner with memory. Journal of Cryptology 9, 111–126 (1996)
10. Golić, J.D., Bagini, V., Morgari, G.: Linear cryptanalysis of bluetooth stream cipher. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 238–255. Springer, Heidelberg (2002)
11. Hermelin, M., Nyberg, K.: Correlation properties of the bluetooth combiner. In: Song, J.S. (ed.) ICISC 1999. LNCS, vol. 1787, pp. 17–29. Springer, Heidelberg (2000)
12. Johansson, T., Jönsson, F.: Improved fast correlation attacks on stream ciphers via convolutional codes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 347–362. Springer, Heidelberg (1999)
13. Johansson, T., Jönsson, F.: Fast correlation attacks through reconstruction of linear polynomials. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 300–315. Springer, Heidelberg (2000)
14. Krause, M.: BDD-based cryptanalysis of keystream generators. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 222–237. Springer, Heidelberg (2002)
15. Lee, S., Chee, S., Park, S., Park, S.: Conditional correlation attack on nonlinear filter generators. In: Kim, K.-C., Matsumoto, T. (eds.) ASIACRYPT 1996. LNCS, vol. 1163, pp. 360–367. Springer, Heidelberg (1996)
16. Löhlein, B.: Attacks based on conditional correlations against the nonlinear filter generator, http://eprint.iacr.org/2003/020
17. Lu, Y., Meier, W., Vaudenay, S.: The conditional correlation attack: A practical attack on bluetooth encryption. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 97–117. Springer, Heidelberg (2005)
18. Lu, Y., Vaudenay, S.: Cryptanalysis of bluetooth keystream generator two-level E0. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 483–499. Springer, Heidelberg (2004)
19. Lu, Y., Vaudenay, S.: Faster correlation attack on bluetooth keystream generator E0. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 407–425. Springer, Heidelberg (2004)
20. Lu, Y., Vaudenay, S.: Cryptanalysis of an e0-like combiner with memory. Journal of Cryptology 21, 430–457 (2008)
21. Matsui, M.: Linear cryptanalysis method for DES cipher. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (1994)
22. Meier, W., Staffelbach, O.: Fast correlation attacks on certain stream ciphers. Journal of Cryptology 1, 159–176 (1989)
23. Meier, W., Staffelbach, O.: Correlation properties of combiners with memory in stream ciphers. Journal of Cryptology 5, 67–86 (1992)
24. Petrakos, N., Dinolt, G.W., Michael, J.B., Stanica, P.: Cube-type algebraic attacks on wireless encryption protocols. Computer 42(10), 103–105 (2009)

25. Preneel, B.: Stream ciphers: Past, present and future (2010)
26. Saarinen, M.: Re: Bluetooth and E0. Posting to Sci. Crypt. Research 2(09) (2000)
27. Shaked, Y., Wool, A.: Cryptanalysis of the bluetooth E_0 cipher using oBDD's. In: Katsikas, S.K., López, J., Backes, M., Gritzalis, S., Preneel, B. (eds.) ISC 2006. LNCS, vol. 4176, pp. 187–202. Springer, Heidelberg (2006)
28. Siegenthaler, T.: Decrypting a class of stream ciphers using ciphertext only. IEEE Transactions on Computers C-34, 81–85 (1985)

A The Key Recovery Distinguisher Based on Condition Masking

Algorithm 1. The key recovery method based on condition masking

Parameters: n, λ, $\mathcal{B}$ and $D_{f_{\mathcal{B}'}}$
input:
1: for $i = 1, 2, \ldots, n$, $\mathcal{B}_i'^K$ for all k-bit K
2: $Z_i^{\mathcal{K}} = f_{\mathcal{B}'}(\mathcal{B}_i^{*\mathcal{K}}, X_i)$ for the right key $\mathcal{K}$ with uniformly and independently distributed v-bit vectors X_i and $\mathcal{B}_i^{*\mathcal{K}} = \mathcal{B}_i^{\mathcal{K}} \setminus \mathcal{B}'^{\mathcal{K}_i}$
3: uniformly and independently distributed Z_i^K for all the wrong keys K such that $K \neq \mathcal{K}$
Goal: find $\mathcal{K}$
Processing:
4: **for** all k-bit K **do**
5: $G(K) \leftarrow 0$
6: **for** $i = 1, \cdots, n$ **do**
7: $G(K) \leftarrow G(K) + \log_2(2^r \cdot D_{f_{\mathcal{B}_i'^K}}(Z_i^K))$
8: **end for**
9: **end for**
10: output $\mathcal{K}$ that maximizes the grade $G(\mathcal{K})$

Theorem 10. *Given a condition mask λ, the above Algorithm 1 solves the problem in Definition 7 with $n_{\mathcal{B}'} = \frac{4k \log 2}{E[\Delta(f_{\mathcal{B}'})]}$ samples and the time complexity is $O(n_{\mathcal{B}'} \cdot 2^k)$, where the condition bits $\mathcal{B}'$ is defined by λ, the expectation is taken over all the uniformly distributed $\mathcal{B}'$. Further, if the $\mathcal{B}_i'^K$ and Z_i^K can be expressed by*

$$\mathcal{B}_i'^K = L(K) \oplus a_i,$$

$$Z_i^K = L'(K) \oplus a_i' \oplus g(\mathcal{B}_i'^K),$$

for all k-bit K and $i = 1, 2, \cdots, n$, where g is an arbitrary function, L, L' are linear functions, and a_i, a_i' are independently and uniformly distributed constants known to the distinguisher. Under these assumptions we can use the FWT algorithm to achieve the optimal time complexity $O(n_{\mathcal{B}'} + k2^{k+1})$ with pre-computation $O(k2^k)$. Besides, $|\mathcal{B}'| = k$.

B The Linear Approximation of Two-level E0

Following the specification in [3], the last generated 128 bits $S^i_{[-127,\cdots,0]}$ in the first level are arranged in octets denoted by $S[0],\cdots,S[15]$, e.g., $S[0] = (S^i_{-127}S^i_{-126}\cdots S^i_{-120})$, where $S^i_{[-127,\cdots,0]} = R^i_{[-127,\cdots,0]} \oplus \alpha^i_{[-127,\cdots,0]}$. From Section 2, we have $V^i_{[1,\cdots,128]} = G_3(R^i_{[-127,\cdots,0]}) \oplus G_3(\alpha^i_{[-127,\cdots,0]})$, where G_3 is depicted in Fig.2. For brevity, we define $(U^i_1,\cdots,U^i_{128}) = G_3(R^i_{[-127,\cdots,0]})$. According to Fig.2, $V^i_{[1,\cdots,24]}$ can be expressed as $V^i_{t'} = U^i_{t'} \oplus \alpha^i_{t_1} \oplus \alpha^i_{t_2} \oplus \alpha^i_{t_3} \oplus \alpha^i_{t_4}$, for $t' = 1,\cdots,24$, where t_1, t_2, t_3, t_4 are the fixed time instants of α^i before the application of G_3.

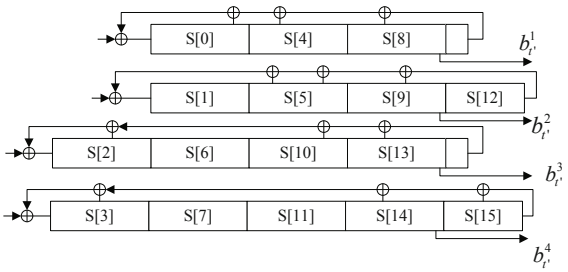


Fig. 2. Distribution of the last 128 bits in the first level

Note that we have $U^i_{t'} = H_{t'}(K) \oplus H'_{t'}(P^i)$, where $H_{t'}, H'_{t'}$ are public linear functions dependent on t'. At the second level, $z_{t'} = V_{t'} \oplus \beta_{t'}$ holds. Hence we have

$$z_{t'} \oplus H_{t'}(K) \oplus H'_{t'}(P^i) = \alpha^i_{t_1} \oplus \alpha^i_{t_2} \oplus \alpha^i_{t_3} \oplus \alpha^i_{t_4} \oplus \beta^i_{t'}, \text{ for } t' = 1,\cdots,24. \quad (6)$$

Given a linear mask γ with $|\gamma| = l$, let $Z^i_{t'} = (z^i_{t'},\cdots,z^i_{t'+l-1})$. Since at level two (in Fig.2), the 128-bit keystream S^i_t are loaded in the reverse order of that at level one, then Eq.(6) can be rewritten with the linear mask notation as

$$\bar{\gamma} \cdot (Z^i_{t'} \oplus \mathcal{L}_{t'}(K) \oplus \mathcal{L}'_{t'}(P^i)) = \bigoplus_{j=1}^{4} (\gamma \cdot C^i_{t_j}) \oplus \bar{\gamma} \cdot C^i_{t'}, \quad (7)$$

for $i = 1,\cdots,n$ and $\mathcal{L}_{t'}, \mathcal{L}'_{t'}$ are fixed linear functions which can be derived from $H_{t'}, H'_{t'}$. Here we have $t' \in \bigcup_{d=0}^{2}\{8d+1,\cdots,8d+9-l\}$.[10] Eq.(7) corresponds to the case of $\lambda = \mathbf{1}_u$.

[10] From Eq.(7), the time instant t_j in $C^i_{t_j}$ are continuous, so the approximation is only set up in this requirement.

Structural Evaluation of AES and Chosen-Key Distinguisher of 9-Round AES-128

Pierre-Alain Fouque[1], Jérémy Jean[2], and Thomas Peyrin[3]

[1] Université de Rennes 1, France
[2] École Normale Supérieure, France
[3] Nanyang Technological University, Singapore

Abstract. While the symmetric-key cryptography community has now a good experience on how to build a secure and efficient fixed permutation, it remains an open problem how to design a key-schedule for block ciphers, as shown by the numerous candidates broken in the related-key model or in a hash function setting. Provable security against differential and linear cryptanalysis in the related-key scenario is an important step towards a better understanding of its construction.

Using a structural analysis, we show that the full AES-128 cannot be proven secure unless the exact coefficients of the MDS matrix and the S-Box differential properties are taken into account since its structure is vulnerable to a related-key differential attack. We then exhibit a chosen-key distinguisher for AES-128 reduced to 9 rounds, which solves an open problem of the symmetric community. We obtain these results by revisiting algorithmic theory and graph-based ideas to compute all the best differential characteristics in SPN ciphers, with a special focus on AES-like ciphers subject to related-keys. We use a variant of Dijkstra's algorithm to efficiently find the most efficient related-key attacks on SPN ciphers with an algorithm linear in the number of rounds.

Keywords: SPN, Block Cipher, AES, Related-Key, Chosen-Key.

1 Introduction

Block ciphers and hash functions are among the most important primitives in cryptography and while their respective goals are different, they are related in many ways. For example, most compression functions, which can in turn be used to define a hash function, are built upon an internal block cipher thanks to classical constructions [19,28] such as Davies-Meyer (DM), Matyas-Meyer-Oseas (MMO) or Miyaguchi-Preneel (MP). One of the main differences between the two primitives is that in the case of the block cipher, the key input is unknown and uncontrolled by the attacker, whereas for the compression function, the attacker has full control over the key schedule (generally called message expansion in that context). Yet, the so-called *related-key* attack scenario [3, 5] is interesting for both cases. This model allows the attacker to incorporate differences not only in the plaintext or ciphertext input, but also in the key input. While less relevant in practice than the classical single-key model, it is important to analyze block ciphers in the light of related-key attacks since the secret keys are

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 183–203, 2013.

often updated in security protocols or differences can be incorporated using fault attacks. Moreover, related-key attacks are also very important when the block cipher is used as inner primitive of a hash function, and in that setting one can even consider the known-key [22] or chosen-key models [8] where the attacker is given knowledge or complete control of the key and his goal is to exhibit some non-ideal property of the primitive.

Avoiding high-probability related-key differential characteristics is one of the goal of the key schedule, and so far various directions have been investigated to construct this component. Resisting to attacks in this setting has for example been taken into account in the conception of the `Whirlpool` hash function [1], by using the same `AES`-like permutation for both the internal permutation and the message expansion part, leading to a strong key schedule in terms of number of Sbox calls, but quite slow as it represents about half of the total amount of computations. As a complete opposite, the designers of the `LED` block cipher [17] chose to use no key schedule at all, at the expense that an important number of rounds is required. These two functions can both provide provable security with regard to related-key differential attacks, but they also both suffer from efficiency issues. In general (see for example `AES` or `PRESENT` [11]), key schedules are built by using an ad-hoc and relatively light function that is quite different from the main permutation, in a hope that this will avoid any correlation between the two components and enforce low-probability related-key differential characteristics. However, because of the heuristic design process and the difficulty of the task, no real security argument is given and this can eventually lead to security issues [7,9]. To help designers and cryptanalysts, many automated differential analyses have already been applied to various primitives [9, 12, 13, 20, 29].

The `AES` block cipher [14] is currently the most interesting candidate to scrutinize with regard to related-key, chosen-key attacks or when used as a black-box in cryptosystems: during the NIST `SHA-3` hash function competition, many candidates [2, 4, 15] reused some components from the `AES` and related-key attacks on the `AES-192` and `AES-256` [7, 8] have been discovered. While differential and linear attacks against the `AES` in the single-key scenario seem to be mastered since the design of the cipher focuses in particular to resist to those class of attacks, provable security against related-key attacks remains more complex to tackle.

Graph Traversal Algorithms. In [24], Matsui proposes an algorithm to find the best differential characteristics for `DES`. The strategy to find the best one on n rounds first starts by computing the best ones on 1 to $n-1$ rounds. The algorithm works by induction and can be seen as a tree traversal in a depth-first manner, where the tree represents all the possible differential characteristics in the cipher layered by round. The nodes represent the actual differences and the edges the possible transitions between them, and are labeled by their probabilities. One differential characteristic is a path in this tree, and its probability equals the product of all traversed edges. We are looking for the path with the highest probability in this tree. The knowledge of the previous best characteristics, i.e. up to some depth in the tree, allows pruning during the procedure like the A^*

heuristic [18]: the target value being known (the exhaustive search bound), we can reduce the possibilities for each one-round transition. Using such algorithm, the complexity is exponential in the number of nodes in the tree, and therefore in the block-size and the number of rounds, except if the pruning is very efficient.

In modern byte-oriented ciphers, designers ensure there is a fast diffusion and that all actual differential transitions occur with the same probability: all differences become equivalent. Consequently, Matsui's search algorithm becomes less efficient since there is no dominant characteristic. Biryukov and Nikolić propose in [9] to restrict the search to truncated differences to decrease the number of edges in the tree. They also introduce a nice representation of truncated differences to consider the branching (combinatorial explosion of differences) in the key schedule.

In this article, we change the tree representation from the previous works into a graph: the nodes and edges have the same signification as before, but we merge all the nodes representing the same differences into a single one. Matsui's tree encodes all the paths of our graph. This merging allows to view the path search as a Markov process: round i is independent of the paths in rounds 0 to $i-1$. Consequently, the numbers of nodes and edges become linear in the number of rounds. Finding the best differential characteristics is reduced to a shortest path problem: we want all the shortest paths in this graph to get all the differential characteristics with the highest probabilities. We use a variant of Dijkstra algorithm combined with the A^* heuristic to explore a kind of *graph product* in a breadth-first manner. Our algorithm uses a dynamic programming method, which was considered too costly in terms of memory in [9]. This approach solves the problem of finding the best related-key characteristics using graph algorithms in polynomial time in the number of rounds and exponential in the state, whereas the previous best known methods were exponential in both parameters using Matsui's algorithm variants (the search in [9] was made possible thanks to an extreme pruning in the AES tree).

Structural Evaluation. By structural evaluation, we mean the domain of cryptography that analyzes a cryptosystem in terms of generic constructions using black-box elements. We are interested in how the building blocks of the primitives interact together, while "ignoring their semantic definitions as particular functions" as in meet-in-the-middle attacks [10].

In this line of research, a major result is the conception of Rijndael, or how to construct a block cipher provably resistant to differential attacks. Daemen and Rijmen show in [14] a lower bound B_r on the number of active Sboxes for any differential characteristic on r rounds of Rijndael, when no difference is introduced in the key. For an Sbox with maximal differential probability p_{max}, this result allows to upper bound the probability of success of any differential attack on r rounds by $p_{max}^{B_r}$. For k-bit keys block ciphers, the resistance to differential cryptanalysis means $p_{max}^{-B_r} > 2^k$, which gives a criteria on r and p_{max} for the security of the cipher. In [10], Biryukov and Shamir analyze the SASAS construction, alternating five layers of non-linear S and affine A functions. They show that five such rounds are vulnerable to a very efficient structural attack, even though the adversary does not know anything about the inner structure of

both S and A. Finally, we mention the work by Miles and Viola in [26], where they prove the security of bounded-input-length pseudo-random functions based on Substitution-Permutation Networks like the AES. In this article, we study the structural analysis of generic Substitution-Permutation Networks in the related-key model. Contrary to the single-key model, it seems impossible to prove anything on the key schedule resistance in the same vein as [14, 26], so we build a tool to study this problem.

Open-Key Model. The open-key model has been introduced so as to investigate the security of block ciphers in relaxed versions of the standard model. In [22], Knudsen and Rijmen studied what they called the *known-key* model where the key is public and the goal of the adversary consists in distinguishing the cipher from a random permutation. At CRYPTO 2009, Biryukov et al. introduced in [8] a more relaxed version called *chosen-key* model where the adversary must exhibit a nontrivial property of the cipher when he has the freedom of the key bytes as extra parameters. They show how to find differential q-multicollisions for AES-256 in time $q \cdot 2^{67}$. For an ideal cipher, constructing q-multicollisions would require at least $O\left(q \cdot 2^{\frac{q-1}{q+1}128}\right)$. At FSE 2010, Gilbert and Peyrin introduced in [16] particular properties for the known-key model by using high-probability differential characteristics on 8 rounds of the block cipher AES-128. Given a key k, two known subspaces Δ_{IN} and Δ_{OUT}, they show how to find one pair of inputs (m, m') to the block cipher E_k such that $m \oplus m' \in \Delta_{IN}$ and $E_k(m) \oplus E_k(m') \in \Delta_{OUT}$ more efficiently than a generic attack on a random permutation, based on a restricted variant of the birthday paradox. In this work, given δ, Δ_{IN} and Δ_{OUT}, we are interested in finding a pair of keys (k, k') and a pair of messages (m, m') such that $k \oplus k' = \delta$, $m \oplus m' \in \Delta_{IN}$ and $E_k(m) \oplus E_{k'}(m') \in \Delta_{OUT}$.

Our Contributions. The goal of this article is twofold. First, we describe an efficient and generic tool that computes all the best differential characteristics for general SPN ciphers, in particular for AES-like ciphers, and then we apply it to the structure of the AES-128. While our algorithm also works in the single-key setting and retrieves the tight proven bounds of the AES structure, we focus this article on the related-key model where the classical XOR difference is the relation in the keys. The search complexity for related-key differential is improved from several days of computations in [9] with a depth-first algorithm to a few hours on a single PC using our breadth-first approach. We also show that the theoretical upper bound $2^{-6 \cdot 17} = 2^{-102}$ mentioned in [9] for the best 5-round characteristic cannot be reached, since the truncated characteristic can only be instantiated with a probability at most 2^{-105}.

As an application of our tool, we study AES-128 as a particular SPN cipher. First, we perform a structural analysis where we consider the MDS property of the diffusion layer, but we do not specify its coefficients. The results show that in order to prove the security of 10 rounds of the cipher in the related-key model, one needs to consider more than just its structure: one needs in particular to consider the differential properties of the non-linear Sbox. Secondly,

we analyze the structure of AES-128 in the hash function setting, where the key schedule and the message parts can be attacked somewhat independently by the adversary. We also show that this setting cannot be proven secure against differential cryptanalysis unless additionnal information about the instantiation of the SPN cipher are provided (e.g., the Sbox and the linear layer). Finally, we construct a chosen-key distinguisher for 9 rounds of AES-128 that requires 2^{55} simple operations and 2^{32} words of memory storage: this was considered an open problem until now, e.g. [9]. Our distinguisher exhibits a non-random property in the chosen-key setting and such a property for an ideal cipher would be detected only after 2^{68} encryptions queries.

Organization of the Paper. In Section 2, we first introduce definitions regarding the ciphers studied and the types of differences we analyze. We then describe our generic tool and give some improvements in the specific case of AES-like ciphers in Section 3. Finally, we present the results of our structural evaluation in Section 4.1, more specifically on AES-128 in Section 4.2, and we precise the construction of the chosen-key distinguisher for 9 rounds of AES-128 in Section 4.3.

2 Definitions

2.1 SPN and AES-Like Ciphers Description

To keep our reasoning as general as possible, we give in this subsection a generic description of Substitution-Permutation Network (SPN) ciphers, and we identify the subgroup of the AES-like ciphers. We refer to the corresponding specifications for a detailed description of the AES [14,27]. We consider that the block ciphers studied here take as input a plaintext or ciphertext of size n bits, and a key of size k bits. The cipher is composed of R successive applications of a round function, and we denote respectively s_i and k_i the successive internal states of the block cipher and the key schedule after the i-th round. The state s_0 is initialized with the input plaintext and k_0 with the input key. One round i is itself composed of three layers: a key extraction and incorporation layer (AK) where a n-bit round-key rk_{i-1} is extracted from k_{i-1} and xored to s_{i-1}, a block cipher permutation layer $\mathcal{BC}$ that updates the n-bit current state of the block cipher after addition of the subkey, i.e. $s_i = \mathcal{BC}(s_{i-1} \oplus rk_{i-1})$, and a key schedule transformation layer $\mathcal{KS}$ that updates the k-bit current state of the key schedule, i.e. $k_i = \mathcal{KS}(k_{i-1})$. The final ciphertext is then defined as $s_R \oplus rk_R$.

Definition 1. (SPN cipher) *Let a block cipher $\mathcal{E}$ whose internal state is viewed as a t_{BC}-cell vector (where $t_{\mathsf{BC}} = \frac{n}{b}$), each cell representing a b-bit word, and the key schedule as a t_{KS}-cell vector (where $t_{\mathsf{KS}} = \frac{k}{b}$). The block cipher $\mathcal{E}$ is called an SPN cipher when its round function $\mathcal{BC}$ is made of a linear function P and a non-linear permutation S, with $\mathcal{BC} = \mathsf{P} \circ \mathsf{S}$, the latter applying one or distinct b-bit S-Boxes to every cell.*

In the even more particular case of AES-like ciphers, the internal state of $\mathcal{BC}$ can be viewed as a square matrix of b-bit cells with d rows and d columns ($n = d^2 \cdot b$).

A cell of s_i is denoted by $s_i^{x,y}$, where x is its row position and y its column position in the square matrix, starting the counting from 0. Then, the linear layer is itself composed of the ShiftRows transformation (ShR), that moves each cell by x positions to the left in its own row, and the MixColumns transformation (MC), that linearly mixes all the columns of the matrix separately. Overall, for AES-like ciphers we have $\mathcal{BC} = \mathsf{P} \circ \mathsf{S} = \mathsf{MC} \circ \mathsf{ShR} \circ \mathsf{S}$ (Figure 1).

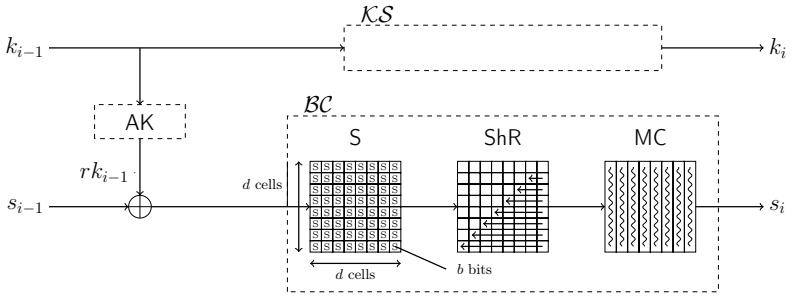


Fig. 1. One round of the generic SPN and AES-like ciphers

2.2 Truncated and Actual Differences

In this article, we are interested in differential attacks [6]. Usually, in this scenario the attacker looks for the bitwise difference between two state values. However, here we also consider truncated differential attacks [21].

Definition 2. *Let $A = [A^{x,y}]$ and $B = [B^{x,y}]$ two states. We denote their **truncated difference** by $\Delta = [\Delta^{x,y}]$ with $\Delta^{x,y} = 1$ if and only if $A^{x,y} \neq B^{x,y}$ (active cell), and $\Delta^{x,y} = 0$ otherwise (inactive cell). We denote their **actual difference** by $\delta = [\delta^{x,y}]$ with $\delta^{x,y} = A^{x,y} \oplus B^{x,y}$.*

First, we need analyze the effect of the cipher transformations on the truncated and actual differences.

The Substitution Layer. One can easily check that the substitution layer S has no effect on the truncated difference of a cell: a cell remains in the same active/inactive situation after application of the transformation. However, S has an effect on the actual difference of every active cells. This effect can be visualized by the differential distribution table (DDT) of the S-Boxes. More precisely, for each possible pair $(\delta_{in}, \delta_{out})$ of actual difference on the input/output of the S-Box S, the table gives the number $\mathrm{DDT}(\delta_{in}, \delta_{out}) = x$ of values X that validate this differential transition, i.e. $S(X) \oplus S(X \oplus \delta_{in}) = \delta_{out}$. Alternatively, $x/2^b$ represents the differential probability of the transition. An important criteria that can be derived from this table is the maximal differential probability p_{max}, which is the highest possible differential probability when $\delta_{in} \neq 0$ and $\delta_{out} \neq 0$.

In order to measure the quality of a truncated differential characteristic, we use the classical counting of the number of active S-Boxes appearing in the characteristic, and we denote it $|\cdot|$.

Definition 3. *Let $v = [\Delta^i]$ be a vector of truncated differences. The* ***weight*** *of v is the number of active differences in v: $\sum_{\Delta^i \neq 0} 1$. We denote it $|v|$ and generalize the notion to any matrix v.*

The Permutation Layer for AES-Like Ciphers. Since the ShR layer only moves the cells around, it only changes the active/inactive cells positions in the internal state, but not their number. The same reasoning applies to the actual differences. The MC transformation being linear, the effect on the values and the actual differences is the same and therefore for each column of the internal state, the output actual differences are simply deduced by the application of the MC linear matrix. Concerning the truncated differences, the effect depends on the branching number B_{MC} of the MC matrix. The branching number is the minimum amount of active cells one can get on both the input and the output of the matrix, excluding the case when there are both null. This measure of the diffusion is crucial for the security of many cryptography primitives and, in general, the MC matrix is Maximum Distance Separable (MDS), that is $B_{MC} = d + 1$ is maximal. A valid truncated differential transition forcing i cells to be inactive on the output happens with probability $2^{-b \cdot i}$.

2.3 Structural Evaluation

Considering only truncated differences enables the cipher designers to get an estimation on the quality of the structure of their primitive in regard to provable security. As an example, the designers of the AES can easily derive the minimal number of active S-Boxes in any number of rounds of the cipher in the single-key model [14], by ignoring the instantiation of the matrix of the diffusion layer. This means that the same reasoning applies for *any* diffusion matrix. In this paper, we are also interested in a generalization of this notion for the related-key model.

More formally, we denote by $E^{S,P}$ a block cipher that uses a substitution layer instantiated by S, and a permutation layer instantiated by P. On the one hand, if E represents the AES family of permutations, we can either plug the AES S-Box S_{AES} ($S \leftarrow S_{\text{AES}}$), or a random bijection S ($S \xleftarrow{\$} \mathfrak{S}_{2^b}$) selected uniformly at random in the set $\mathfrak{S}_{2^b}$ of the permutations on b bits. In the latter setting, we loose the information $p_{max} = 2^{-(b-2)}$ as we can only get $2^{-(b-1)} \leq p_{max} \leq 1$, but we are interested in how the structure of the AES resists to differential cryptanalysis when we relax the strong information on p_{max}. We perform the same abstraction for the P layer, where the matrix is selected uniformly at random in the set of all the $d \times d$ matrices with coefficients in $\text{GF}(2^b)$.

3 Generic Related-Key Differential Characteristic Search Tool for SPN Ciphers

In this section, we explain the inner workings of our generic related-key differential characteristic search tool for SPN ciphers. As a first step, we model the problem by assuming that the cipher round function is a Markov process in regard to the truncated differential characteristic search (Section 3.1). This allows us to reduce the problem to a shortest path search in a special $(r + 1)$-equipartite directed acyclic

graph, for which we provide a simple yet powerful algorithm. The precomputation phase of the process is devoted to building the graphs on which we work on (Section 3.2), while the online phase looks for the shortest paths (Section 3.3). We note that we can tweak the Markov assumption to find not only the best truncated differential characteristics, but also the actual difference ones.

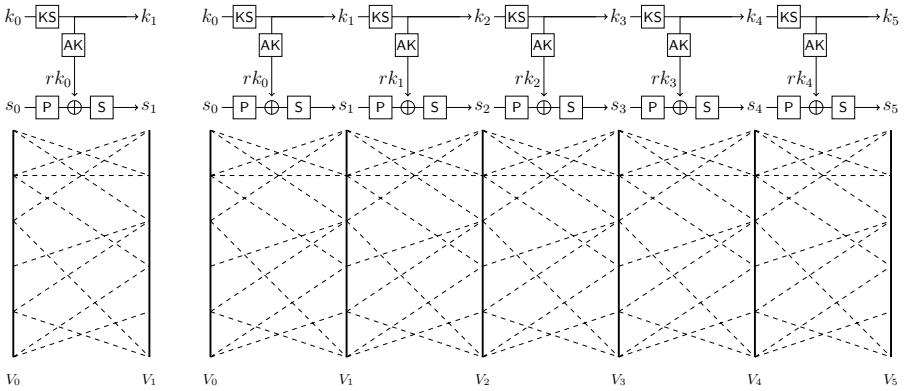
3.1 Differential Characteristic Search as a Graph Modeling of a Markov Process

We describe here an algorithm that generates for any number of rounds *all* the related-key truncated differential characteristics for SPN ciphers with minimal number of active S-Boxes. This analyzes the structure of the cipher in regard to the resistance against related-key attacks. We make a simple assumption: we would like the search to be a Markov process. More precisely, we assume that the possible differential transitions through a round from one truncated state to another one does not depend on previous rounds transitions. If we stick to the real definition of truncated differentials (i.e. without implicit conditions contained), then this assumption is verified for SPN ciphers: knowing the truncated input difference of one round represents all the information needed in order to deduce the possible output ones.

Graph Modeling. In order to find the best r-round related-key truncated differential characteristics, we use a graph modeling of the problem. Let G be the 2-equipartite directed acyclic graph of all the possible one-round transitions. Then all the best r-round related-key truncated differential characteristics correspond to all the shortest paths in the $(r+1)$-equipartite directed acyclic graph G_r built by concatenating r copies of G. Namely, denoting $G = (V_0, V_1 \,;\, E_{0,1})$ the 2-equipartite graph linking with one cipher round a state in set V_0 to a state in set V_1 using some edge in set $E_{0,1}$, we build the graph G_r representing r rounds of the cipher by $G_r = (V_0, \ldots, V_r \,;\, E)$ such that for all i, the subgraph $(V_i, V_{i+1} \,;\, E_{i,i+1})$ is equal to G. Note that all edges are oriented from V_i to V_{i+1}, and that $|V_i| = |V_{i+1}|$. The nodes of the graph stand for all the possible pairs $(\Delta_{\mathsf{KS}}, \Delta_{\mathsf{BC}})$ where Δ_{KS} represents the truncated difference on the key schedule state and Δ_{BC} represents the truncated difference on the block cipher state. Since we have $2^{t_{\mathsf{KS}}}$ possible values Δ_{KS} and $2^{t_{\mathsf{BC}}}$ possible values Δ_{BC}, all V_i in the graph are composed of $2^{(k+n)/b}$ nodes. The edges correspond to a possible one-round related-key truncated differential characteristic from the input to the output vertex and in the worst case where all differential transitions are possible, we have $2^{2(k+n)/b}$ edges. A path in G_r is defined as a sequence of $r+1$ nodes, one in each of the V_i.

We note that the probability costs are not associated to the edges, but to the output nodes. Indeed, the number of active cells in the output node represents the number of active S-Boxes during this round[1]. We denote C_{BC} (resp. C_{KS}) the total number of active S-Boxes in the internal permutation part of the block

[1] To be able to associate the number of active S-Boxes in the key schedule to the output node as well, we make the weak assumption that one round of the key schedule is composed of an S-Box and a linear layer at most.



(a) Graph G. (b) Graph G_5.

Fig. 2. Examples of simplified versions of the two graphs G and G_5. Variables s_i and k_i represent the current internal permutation/key state respectively, while rk_i stands for the subkey generated during the round.

cipher (resp. in the key schedule part) in the whole characteristic. Depending on the situation considered, one might want to minimize $C_{\mathsf{BC}} + C_{\mathsf{KS}}$ for classical scenarios, or instead $\max\{C_{\mathsf{BC}}; C_{\mathsf{KS}}\}$ for hash function settings, where the key schedule and the block cipher parts can be attacked sequentially (first the key schedule part, and then the block cipher one).

Theorem 1. (Search algorithm) *Let $\mathcal{E}$ be a SPN cipher on n-bit blocks using a k-bit internal state in the key schedule. Both states are viewed as vectors of b-bit cells. There exists an algorithm $\mathcal{A}$ with a theoretical time complexity of $O(r \cdot 2^{(2n+k)/b})$ that finds all the best characteristics on r rounds of $\mathcal{E}$.*

We emphasize that algorithm $\mathcal{A}$ will find *all* the shortest paths in G_r representing the differential transitions of r rounds of $\mathcal{E}$. Moreover, we note that the time complexity of $\mathcal{A}$ can be greatly reduced with heuristics. We describe in the next two sections our tool that searches for the best r-round related-key truncated differential characteristics.

3.2 Precomputation Phase

The precomputation phase builds the graph G. It can be built and stored efficiently by observing its inner structure: the block cipher internal state output depends only on the block cipher internal state input and the incoming subkey (deduced by the extraction phase from the key schedule internal state input), while the key schedule internal state output depends only on the key schedule internal state input. Therefore, G can actually be described as a special product of two smaller graphs G_{BC} and G_{KS}, such that an edge $(s_i, k_j) \to (s_{i'}, k_{j'})$ exists in G if and only if $k_j \to k_{j'}$ exists in G_{KS} and $(s_i, k_j) \to s_{i'}$ exists in G_{BC}. On the one hand, G_{BC} is a bipartite directed acyclic graph whose input nodes are all the possible block cipher internal state and subkey pairs, and whose output nodes are all the possible block cipher internal state. The edges represent input nodes that can be mapped

to output nodes through a valid differential transition. On the other hand, G_{KS} is a 2-equipartite directed acyclic graph, whose input and output nodes are all the possible key schedule internal states. The edges represent input nodes that can be mapped to output nodes through a valid differential.

This observation slightly reduces the amount of computation/memory to build/store G: the number of vertices in G_{BC} is $v_{\mathsf{BC}} = 2^{t_{\mathsf{BC}}+t_{\mathsf{KS}}} + 2^{t_{\mathsf{BC}}}$ and the number of vertices in G_{KS} is $v_{\mathsf{KS}} = 2 \times 2^{t_{\mathsf{KS}}}$. This has to be compared with the $2 \times 2^{t_{\mathsf{BC}}+t_{\mathsf{KS}}}$ nodes in G. For example, in the particular case of the AES-128, this trick reduces the number of nodes from 2^{33} in G to $v_{\mathsf{BC}} = 2^{32} + 2^{16}$ in G_{BC} and 2^{16} in G_{KS} and mainly allows to apply an *early-abort* approach to prune edges in G in the online phase. More importantly, the total number of edges shrinks considerably from $e_{\mathsf{BC}} \cdot e_{\mathsf{KS}}$ to $e_G = e_{\mathsf{BC}} + e_{\mathsf{KS}}$, which equals to $2^{33.6} + 2^{22.15}$ in the case of AES-128.

The Graph G_{BC}. It can be built by repeating the three following steps for all the $2^{t_{\mathsf{BC}}}$ possible truncated differences Δ_{in} on the input and all the $2^{t_{\mathsf{BC}}}$ possible truncated differences Δ_{out} on the output.

1. Compute all the possible truncated differences Δ_x that can be obtained from Δ_{in} through the P layer (on the backward direction, a truncated difference Δ_{out} stays the same when inverting the S layer).
2. For every Δ_x found, compute all the possible truncated differences Δ_k on the key state that can be obtained from $\mathsf{AK}^{-1}(\Delta_x \oplus \Delta_{out})$.
3. For every Δ_k found, add an edge in G_{BC} from input node (Δ_k, Δ_{in}) to output node Δ_{out} if none exists.

The time complexity to build G_{BC} depends on the average branching B_{P} of the P layer and on the average branching B_{xor} of the subkey XORing layer. It amounts to $2^{2t_{\mathsf{BC}}} \cdot B_{\mathsf{xor}} \cdot B_{\mathsf{P}}$ operations. The memory cost to store G_{BC} corresponds to the number of edges e_{BC} of G_{BC} and is upper bounded by $2^{2t_{\mathsf{BC}}} \cdot B_{\mathsf{xor}} \cdot B_{\mathsf{P}}$ since one operation on step 3 adds at most one edge. We denote $\mathrm{succ}_{\mathsf{BC}}(s, k)$ the set of successors of the state s in the graph G_{BC} using the key k.

The Graph G_{KS}. It is built by simply going through all the $2^{t_{\mathsf{KS}}}$ possible key schedule internal state input truncated differences, checking which output truncated differences can be obtained through the KS layer and adding edges in G_{KS} accordingly[2]. The time and memory complexities depend on the average branching B_{KS} of the KS layer and amounts to $2^{t_{\mathsf{KS}}} \cdot B_{\mathsf{KS}}$ operations. The number of edges e_{KS} of G_{KS} equals $e_{\mathsf{KS}} = 2^{t_{\mathsf{KS}}} \cdot B_{\mathsf{KS}}$. In the sequel, we denote $\mathrm{succ}_{\mathsf{KS}}(k)$ the set of successors of the key k in graph G_{KS}.

3.3 Online Phase

The online phase finds all the shortest paths in G_r with at most $r \cdot (\frac{v_G}{2} \cdot \log(\frac{v_G}{2}) + e_G)$ computations and memory $r \cdot e_G$, thus linear in the number of rounds r. This is

[2] We assume that the key schedule is simple: given a truncated difference on the input, one can find each reachable truncated output difference in constant time. This assumption is weaker than the one from Footnote 1, and verified by most ciphers since a very complex key schedule would make the whole primitive inefficient anyway.

Algorithm 1 – Search all shortest paths in G_r.

function SEARCH(G_r)
  Copy all nodes of G_r in a new graph G_r^*
  for all $v \in V_0$, $c(v) \leftarrow |v|$
  for all $v \in V_1, \ldots, V_r$, $c(v) \leftarrow \infty$
  SORTLIST(V_0)
  for $i = 1 \rightarrow r$ **do**
    for all $v' \in V_i$, by increasing $c(v')$ **do**
      for all $v \in \text{succ}(v')$ **do**
        $\alpha \leftarrow c(v') + |v|$
        if $c(v) = \infty$ **then**
          $c(v) \leftarrow \alpha$
          Add the edge $v' \rightarrow v$ to G_r^*
        else if $c(v) = \alpha$ **then**
          Add the edge $v' \rightarrow v$ to G_r^*
    SORTLIST(V_i)
  return G_r^*

possible because G_r is a vertex-weighted directed acyclic graph. Since the edges have a constant weight (the number of active S-Boxes, i.e. the weights, are on the nodes and not the edges), the function we want to minimize for each node $v \in V_i$, $i \in [1, r]$ is: $|v| + \min_{v' \in \text{pred}(v)}(c(v'))$, where $\text{pred}(v) \subseteq V_{i-1}$ is the set of all predecessors of v and $c(v')$ represents the cost of the shortest path to v'. In other words, if we know the shortest path costs to all the $v' \in V_{i-1}$, we find the shortest path to any $v \in V_i$ by choosing the predecessor of v with the minimal cost.

This can easily be done by creating a list containing all the nodes $v' \in V_{i-1}$ sorted increasingly according to the cost of their shortest path $c(v')$. Then, starting from the cheapest v' and ending to the most expensive one, we set the shortest path cost of all the successors v of v' to $|v| + c(v')$ if and only if the cost of v was not set yet (see Algorithm 1). This is an improvement over the simple shortest path computation in a directed acyclic graph using a topological order since we can take advantage of the vertex-weighted property. In practice, we iteratively build a simpler vertex-weighted directed acyclic graph G_r^* from G_r (all the nodes are the same, but with less edges), for which each node $v \in V_i$ has a cost equal to the cost of the shortest path to v in G_r, and an edge leading to $v \in V_i$ represents one of the shortest paths to v (see Figure 3). At this point, in the graph G_r^* the costs assigned to all the nodes v in V_r represent the cost of the shortest path to v in G_r. If v_G represents the number of vertices and e_G the number of edges in the graph G, then the complexity of the shortest path search is about $r \cdot (\frac{v_G}{2} \cdot \log(\frac{v_G}{2}) + e_G)$ operations: the $\frac{v_G}{2} \cdot \log(\frac{v_G}{2})$ term comes from the construction of the sorted list of the nodes at each round, and the e_G term is the number of edges visited during each round as we visit all of them. Note that this is an upper bound on the complexity since we do not need to go through all $\frac{v_G}{2}$ nodes every rounds, but only a subset of them, and we may cut some edges among all the e_G ones. The term $e_G = e_{\mathsf{BC}} + e_{\mathsf{KS}}$ dominates the complexity, and since $e_{\mathsf{BC}} >> e_{\mathsf{KS}}$, it can be approximated by the number $e_{\mathsf{BC}} \leq 2^{(n+k)/b} \times 2^{n/b}$ of edges in G_{BC}. Hence, the total time complexity is $O(r \cdot 2^{(2n+k)/b})$ for r rounds.

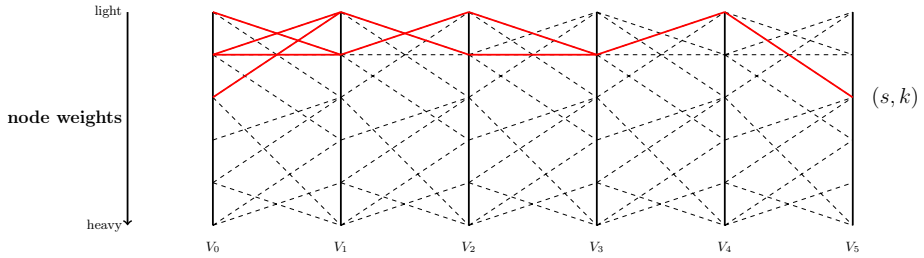


Fig. 3. The dashed edges form an example of a simplified G_5. The thick edges describe paths in the subgraph G_5^* that are shortest paths in G_5 to node (s, k). All the nodes in G_5^* are sorted according to their weight, the top being the cheapest ones.

In order to get *all* the shortest paths in G_r, we need to store at each node $v \in V_i$ not only the first shortest path found to v but all of them (lines 13 and 14 in Algorithm 1). In general, this number is very small and never exceeds the total number of shortest paths anyway. In the worst case where all paths are the shortest, it amounts to the total number of edges $r \cdot e_G$.

As explained previously, in practice we do not use the graph G directly, but two separate graphs G_{BC} and G_{KS}. We can adapt the Algorithm 1 for this setting: in order to build G_r^*, we replace the **for all** loop of line 8 that iterates over all nodes $v' = (s_i, k_i) \in V_i$ by two **for all** loops that describe all $k_{i+1} \in \mathrm{succ}_{\mathsf{KS}}(k_i)$ and all $s_{i+1} \in \mathrm{succ}_{\mathsf{BC}}(s_i, k_{i+1})$.

In [24], Matsui introduces an argument equivalent to the A^* optimization for path-finding or graph traversal algorithms [18] that allows to prune the majority of the edges of G and to avoid the evaluation of many sets of successors. If we know the costs c_k of all k-round characteristics, $1 \leq k \leq n-1$, and we target an n-round characteristic of cost at least c_n, then we can consider only the nodes from V_0 that have a cost at most $c_n - c_{n-1}$, and the ones in V_1 that have a cost at most $c_n - c_{n-2}$. Intuitively, after one round has been passed, we know that we paid *at least* c_1, and since there are $n-1$ remaining rounds to pass, we will need to pay *at least* c_{n-1}. In terms of intervals of costs, for each of the V_i, we only need to consider nodes that have costs in $[c_i, c_n - c_{n-i}]$, $0 \leq i \leq n$ assuming $c_0 = 0$. To take advantage of the A^* heuristic, we sort the sets of successors in both graphs, so that we can perform an extreme pruning of the edges whenever the updated costs exceed the current interval, in an early-abort manner.

Due to space constraints, we cannot detail how to efficiently extend this algorithm to the case of AES-like ciphers, so we continue directly with the consequences of the search for this class of ciphers. However, the full details are given in the extended version of our paper.

4 Applications to SPN and AES-128

4.1 Structural Evaluation of SPN AES-Like Ciphers

We present here the results on the structural evaluation of the AES-like ciphers in regard to the related-key model, which provides an estimation of the security

provided by their key schedule. Namely, we ignore the semantic definition of the S-Box and the MDS matrix, and we are only interested in how they can interact in the related-key settings. The results are measured in terms of number of active S-Boxes as in [22], and presented in Table 1. Lines 2 and 3 of the table provide the minimum number of active S-Boxes (line 2) for any number of rounds when implementing an AES-like cipher, and the number of truncated characteristics that reach that bound (line 3). In these two lines, we count the number of active S-Boxes in both the state and the subkeys, whereas in lines 4 and 5 of Table 1, we consider the case of the hash function setting where the block cipher and its key schedule can be attacked somewhat independently.

Table 1. For the AES-128 cipher on r rounds, this table shows: (1) the minimal number $C_{\mathsf{KS}} + C_{\mathsf{BC}}$ of active S-Boxes in **both** the key schedule C_{KS} and in the block cipher C_{BC} achievable in truncated differential characteristics; and (2), the same figures for the minimal number $\max(C_{\mathsf{BC}}, C_{\mathsf{KS}})$ for the hash function setting. Lines 3 and 5 count the number of distinct truncated characteristics that reach that bound.

Rounds	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$\mathbf{min(}C_{\mathsf{KS}} + C_{\mathsf{BC}}\mathbf{)}$	0	1	3	9	11	13	15	**21**	23	**25**
Trunc. Char. ($\log_2$)	–	4.52	6.58	10.46	5.00	13.26	16.17	21.34	14.90	21.38
$\mathbf{min(max(}C_{\mathsf{BC}}, C_{\mathsf{KS}}\mathbf{))}$	0	1	3	6	7	9	11	14	15	**17**
Trunc. Char. ($\log_2$)	–	4.00	10.00	11.73	10.00	18.92	23.64	>30	>30	>30

Theorem 2. *It is impossible to prove the security of the full* AES-128 *against related-key differential attacks without considering both the differential property of the S-Box and the* P *layer when two keys verify a certain relation. It is impossible to prove the security of the full* AES-128 *in the hash function setting without considering both the differential property of the S-Box and the* P *layer.*

Proof. First, in the case where we consider related-key differential attacks where two keys are related if their difference verify a certain relation (line 2), we remark that for 10 rounds there exists a truncated differential characteristic counting only 25 S-Boxes. As we discussed before, this means that a differential analysis would run in p_{max}^{-25} operations. Consequently, the structure of AES-128 on its own is not enough to prove the resistance to related-key attacks for any ciphers in this class, we at least need to add a criteria on the S-Box via p_{max}.

Secondly, with an S-Box on n bits ($n = 8$ in the AES), the minimal theoretical p_{max} that can be obtained is $2^{-(n-1)}$: consequently, the largest number of rounds that our structural analysis could attack for AES-like ciphers is 7 rounds. Indeed, for 7 rounds, the 15 active S-Boxes give a differential analysis requiring $p_{max}^{-15} \geq 2^{105}$ computations, which might be smaller than 2^{128}. We note that we do not know how to construct an almost-perfect permutation on n bits acting as an S-Box with $p_{max} = 2^{-(n-1)}$. The S-Box chosen in the AES implements a composition of an affine transformation on the inverse mapping, and reach $p_{max} = 2^{-(n-2)}$. Hence, the largest number of rounds that our structural analysis could attack is 8 rounds. Indeed, for 8 rounds, the 21 active S-Boxes

give a differential analysis requiring $p_{max}^{-21} \geq 2^{126}$ computations, which might be smaller than 2^{128}. However, when we instantiate the P layer by the one of the AES-128, we observe that none of the $2^{16.17}$ characteristics found on 7 rounds by our search algorithm nor the $2^{21.34}$ ones for 8 rounds can be instantiated due to linear constraints coming from the key schedule. This means that proving or disproving the security of the AES-128 in the related-key setting needs to consider both the differential properties of the S-Box and the linear equations of the P layer. From an instantiated P layer, we can write a system $\mathcal{Q}$ of linear equations whose solutions are the values of all the truncated differences of the characteristic. Therefore, choosing P such that $\mathcal{Q}$ can be made inconsistent on a small number of rounds brings more security than a random P. Our tool can be used to write this system of linear equations for any truncated differential characteristic.

Finally, for 10 rounds in the hash function setting, there exists characteristics with only 17 active S-Boxes. For the AES-128, in the best case, the differential probability equals $2^{-6 \cdot 17} = 2^{-102}$. In this setting, the adversary is supposed to have full control over the input of both the key schedule and the block cipher, that is why we considered $\max(C_{\mathsf{BC}}, C_{\mathsf{KS}})$ as an objective function to minimize in our search algorithm of Section 3. As the previous structural results, this also means that one cannot prove the security of the full AES-128 against differential cryptanalysis by only analyzing its structure. ■

4.2 Differential Characteristics Results for AES-128

Theorem 3. *After 6 rounds, there is no related-key differential characteristic for* AES-128 *with a probability higher than* 2^{-128}.

Proof. The related-key differential characteristics presented in the previous section are valid only when one deals with truncated differences, and these characteristics give an indication on the structural security provided by the AES-128 key schedule. However, due to the choice of the P layer in AES-128, it turns out that none of them can be instantiated with actual differences, because of inconsistencies in some linear constraints. To overcome this difficulty, and at the cost of a bigger graph G to handle, we first add some more information in the Markov process both on the representation of the key schedule state and the internal permutation state, and we then filter the best characteristics obtained and hope to find one that can be instantiated with actual differences. Our algorithm performs a search fundamentally different from [9], but it finds again and more efficiently the same results.

By a system resolution, we show that from a truncated differential characteristic, we can decide whether it can be instantiated with actual differences, and even find an associated differential characteristic with the greatest probability. As an example, our tool found again the best 17-S-Box truncated differential characteristic on 5 rounds of AES-128, and also found how to achieve the greatest probability 2^{-105} by instantiating the differences. This has to be compared with the upper bound of $2^{-6 \cdot 17} = 2^{-102}$ given in [9] since in the best case, all the AES active S-Boxes have maximal differential probability 2^{-6}. Trying all the

possible differences that instantiate this truncated differential characteristic, we show that we cannot reach that bound, but we can only set 15 out of 17 S-Boxes to the maximal differential probability. The following Table 2 reports the best related-key characteristics found by our tool on AES-128 up to 5 rounds, with their respective highest achievable probabilities. Thus, from 6 rounds, there is no related-key differential characteristic for AES-128 with a probability higher than 2^{-128}. ■

Table 2. After 6 rounds, there is no related-key differential characteristic for AES-128 with a probability higher than 2^{-128}. Our tool retrieved the previous known results but also provides the real differential characteristics with maximum probability.

Rounds	**1**	**2**	**3**	**4**	**5**
$\mathbf{min}(C_{\mathsf{KS}} + C_{\mathsf{BC}})$	0	1	5	13	17
$\mathbf{max}\ \mathbf{log_2}(p)$	0	-6	-31	-81	-105
Appendix reference	–	B	C	D	E

4.3 Distinguishing 9 Rounds of AES-128

As another application of our tool, we describe a 9-round distinguisher for AES-128 in the chosen-key model requiring 2^{55} computations and 2^{32} memory. For an ideal cipher, the same property would be detected after 2^{68} encryption queries. Here, the chosen-key model asks the adversary to find a pair of keys (k, k') satisfying $k \oplus k' = \delta$ with a known difference δ, and a pair of messages conforming to a partially instantiated characteristic in the data part.

We achieve this result by considering the best 5-round related-ley differential characteristic and propagating it backwards to reach 9 rounds. The 5 last rounds hence count 6 active S-Boxes in the key schedule part and 11 in the data part (rounds 5 to 9 in Figure 4 and Table 3). By the backward propagation in the key schedule, we reach a total of 15 active S-Boxes for the key schedule differential characteristic, whose probability equals 2^{-101}. Since we have 2^{128} possible key values, we expect 2^{27} pairs of keys to conform to the differential characteristic in the key schedule. In the block cipher part, we prepend three rounds that we plan to control with an average cost of one computation using the **Super-SBox** technique [14, 16, 23], and one more round at the very beginning that we make as sparse as possible. The entire 9-round differential characteristic is depicted on Figure 4 and in Table 3.

The Distinguishing Algorithm. Once this differential characteristic settled, we find inputs that verify the whole characteristic. We start by finding a pair of keys that conforms to the whole differential characteristic in the key schedule. There are about 2^{27} expected such pairs of keys, and we can generate them at an average cost of one computation by picking random values satisfying all the non-linear transitions and efficiently solve the linear system to retrieve all the subkeys: our implementation of that part confirms about 2^{27} are found. We note that the difference $k \oplus k' = \delta$ is already verified at this stage.

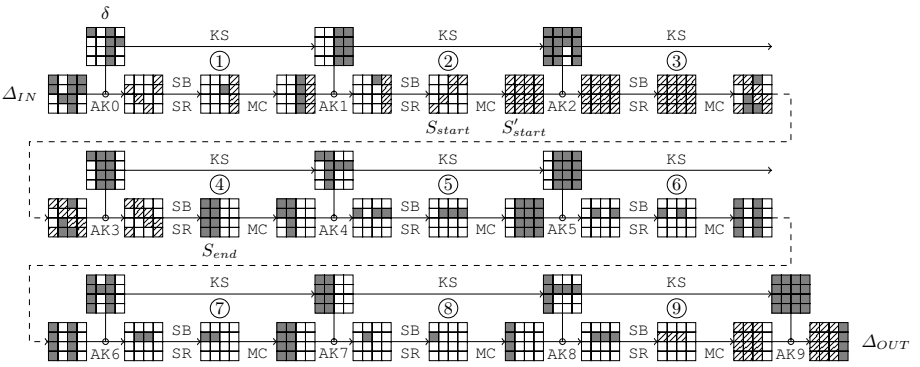


Fig. 4. Differential characteristic of 9-round AES-128 used in the distinguisher. White bytes are inactive, gray bytes have known differences, and hatched bytes are truncated differences.

Table 3. Differential characteristics used in the distinguisher of 9 rounds of AES-128. The known differences are represented by their values from 0x00 to 0xFF, and truncated differences as ??, since their values are unknown, but non-zero. The two lines for state differences are respectively the input difference after key addition and the output difference.

Round	State differences	Key differences
Plaintext	B3??0000 0000??00 28F47A?? ????0000	
1	00??0000 0000??00 000000?? ????0000 00000000 00000000 8EF47A7A ????????	B3000000 00000000 A6F47A7A 008E0000
2	00000000 00000000 28000000 ???????? ???????? ???????? ???????? ????????	00000000 00000000 A6F47A7A A67A7A7A
3	???????? ???????? ???????? ???????? ??0000?? ????7A7A 8E????7A 0000????	8E7A7A7A 8E7A7A7A 288E0000 8EF47A7A
4	??0000?? ????0000 00????00 0000???? 288E0000 8E7A7A7A 00000000 00000000	28000000 A67A7A7A 8EF47A7A 00000000
5	008E0000 00000000 008E0000 008E0000 00000000 8EF47A7A 8EF47A7A 8EF47A7A	28000000 8E7A7A7A 008E0000 008E0000
6	00000000 008E0000 00000000 008E0000 8EF47A7A 00000000 8EF47A7A 00000000	00000000 8E7A7A7A 8EF47A7A 8E7A7A7A
7	00000000 008E0000 008E0000 00000000 8EF47A7A 8EF47A7A 00000000 00000000	8EF47A7A 008E0000 8E7A7A7A 00000000
8	00000000 008E0000 00000000 00000000 8EF47A7A 00000000 00000000 00000000	8EF47A7A 8E7A7A7A 00000000 00000000
9	00000000 008E0000 008E0000 008E0000 ???????? ???????? ???????? 00000000	8EF47A7A 008E0000 008E0000 008E0000
Ciphertext	???????? ???????? ???????? 787A7A7A	78F47A7A 787A7A7A 78F47A7A 787A7A7A

For a pair of keys, we precompute the four arrays T_i containing the paired values of the ith **Super-SBox**: those are four parallel 32-bit non-linear applications obtained by reordering the layers of two rounds of the cipher. To construct the tables T_i, we iterate in parallel over the 2^{32} input values from state S_{end} that corresponds to the ith **Super-SBox** and propagate the values backwards until S'_{start}. We note that the difference in S_{end} is completely determined by our

differential characteristic. We store the pair in T_i indexed by its difference[3], so that this precomputation requires 2^{32} simple operations, a memory complexity of 2^{32}, and depends on the selected pair of keys.

We continue by picking random values for the 5-byte difference after the second non-linear layer in S_{start}, which linearly fixes all the differences in S'_{start}. Note that we can repeat this part about $2^{8 \cdot 5} = 2^{40}$ times. From the precomputed tables T_i, we find on average one pair of messages that verifies the middle rounds from S_{start} to S_{end}. The remaining of the process is probabilistic: backwards, we expect a fraction of 2^{-7} pairs to pass the unique specified S-Box transition in the second round up to Δ_{IN}. Forwards, we expect a fraction of $2^{-6 \times 8} = 2^{-48}$ to verify the 5 last rounds up to Δ_{OUT} (all 8 transitions have been chosen by our tool to be 8 times the same one with maximal probability $p_{max} = 2^{-6}$). Finally, we expect a fraction $2^{-7-48} = 2^{-55}$ of the pairs generated in the middle to propagate correctly forwards and backwards.

By repeating this process for all 2^{40} differences in S_{start} and for 2^{15} distinct pairs of keys, we expect to find a solution for the whole characteristic in $2^{15} \cdot (2^{32} + 2^{40}) \approx 2^{55}$ operations. Note that the freedom degrees left allows to get up to 2^{12} solutions in 2^{67} operations by exhausting the remaining 2^{12} valid pairs of keys.

Ideal Case. For an ideal cipher, the adversary faces a family of random and independent permutations $\{\pi_i, i \in \{0,1\}^{128}\}$. His goal is to find a key k and a pair of messages (m, m') such that: $m \oplus m' \in \Delta_{IN}$ and $\pi_k(m) \oplus \pi_{k \oplus \delta}(m') \in \Delta_{OUT}$, where δ, Δ_{IN} and Δ_{OUT} are specified in Figure 4. Namely, δ is a completely determined 128-bit difference, whereas Δ_{IN} and Δ_{OUT} are two sets of 128-bit differences defined in Figure 4: colored and white bytes are fixed differences, while hatched bytes can take several difference values. On the output, we constrained each of the three independent active bytes after the last non-linear layer of the last round to only 127 reachable difference values (since from a fixed input difference, only 127 output differences can be reached through the AES S-Box), and the MixColumns layer being linear we have $|\Delta_{OUT}| = 127^3 \simeq 2^{21}$. On the input, 4 bytes in Δ_{IN} can take any difference value and 1 byte is constrained to only 127 reachable difference values, thus $|\Delta_{IN}| = 127 \cdot (2^8 - 1)^4 \approx 2^{39}$.

The best known method for the attacker to find (k, m, m') verifying those properties consists in applying the limited birthday algorithm [16]. The additional freedom left in choosing the key bits does not help the attacker to find the actual pair of messages that verifies the required property, since the permutations F_k and $F_{k \oplus \delta}$ have to be chosen beforehand. All in all, the attacker has access to 39 bits of differences at the input and 21 bits in the output, for a pair of permutations on $n = 128$ bits. The limited birthday distinguisher on these permutations finds a solution in time $\max\{\min(2^{IN/2}, 2^{OUT/2}), 2^{IN+OUT-n}\}$, with $IN = n - 39$ and $OUT = n - 21$, which gives a time complexity equivalent to 2^{68} encryption queries.

[3] To simplify, we assume the differences in S'_{start} to be uniformly distributed so that each 32-bit difference appears once. While this simplification is not true in practice, the cost per solution remains one on average, thus it does not change the complexity estimation.

5 Conclusion and Future Works

In this article, we have proposed a simple, efficient and generic algorithm that searches for (truncated) differential characteristics in the single-key, related-key or hash function setting for SPN ciphers. Thanks to this method, we were able to obtain the first non-trivial distinguisher of 9-round AES-128 in the chosen-key model, which has been a long-lasting open problem on this version of AES. We also show that no security proof of AES-128 in the related-key model of the hash function setting can be based only of its structure: one has to take into consideration both the Sbox and the linear layer. We believe this tool will be useful for designers that would like to easily test the security of their own key schedule or message expansion. The research community has still a lot to learn on the security of key schedules and there are many future works possible: extend the 9-round result on AES-128 to the full 10 rounds, find a single-key like security proofs in the related-key model for AES-like ciphers (generic enough to work for any dimension), provide a formal proof of security against differential/linear cryptanalysis for AES in the related-key model, and build simpler, more efficient and more secure key scheduling algorithms.

Acknowledgements. We would like to thank the Martjin Stam, Christian Rechberger and the anonyous referees for their valuable comments on our paper.

References

1. Barreto, P.S.L.M., Rijmen, V.: The Whirlpool Hashing Function. Submitted to NESSIE (September 2000) (revised May 2003), http://www.larc.usp.br/~pbarreto/WhirlpoolPage.html (June 24, 2009)
2. Benadjila, R., Billet, O., Gilbert, H., Macario-Rat, G., Peyrin, T., Robshaw, M., Seurin, Y.: SHA Proposal: ECHO. Submission to NIST (updated) (2009)
3. Biham, E.: New Types of Cryptoanalytic Attacks Using related Keys (Extended Abstract). In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 398–409. Springer, Heidelberg (1994)
4. Biham, E., Dunkelman, O.: The SHAvite-3 Hash Function. Submission to NIST, Round 2 (2009)
5. Biham, E., Dunkelman, O., Keller, N.: A Unified Approach to Related-Key Attacks. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 73–96. Springer, Heidelberg (2008)
6. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 156–171. Springer, Heidelberg (1992)
7. Biryukov, A., Khovratovich, D.: Related-Key Cryptanalysis of the Full AES-192 and AES-256. In: [25] 1–18
8. Biryukov, A., Khovratovich, D., Nikolić, I.: Distinguisher and related-key attack on the full AES-256. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 231–249. Springer, Heidelberg (2009)

9. Biryukov, A., Nikolić, I.: Automatic Search for Related-Key Differential Characteristics in Byte-Oriented Block Ciphers: Application to AES, Camellia, Khazad and Others. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 322–344. Springer, Heidelberg (2010)
10. Biryukov, A., Shamir, A.: Structural Cryptanalysis of SASAS. J. Cryptology 23(4), 505–518 (2010)
11. Bogdanov, A.A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M., Seurin, Y., Vikkelsoe, C.: PRESENT: An Ultra-Lightweight Block Cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007)
12. Bouillaguet, C., Derbez, P., Fouque, P.-A.: Automatic Search of Attacks on Round-Reduced AES and Applications. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 169–187. Springer, Heidelberg (2011)
13. De Cannière, C., Rechberger, C.: Finding SHA-1 Characteristics: General Results and Applications. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 1–20. Springer, Heidelberg (2006)
14. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Springer, Heidelberg (2002)
15. Gauravaram, P., Knudsen, L.R., Matusiewicz, K., Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: Grøstl – a SHA-3 candidate. Submission to NIST, Round 3 (2011)
16. Gilbert, H., Peyrin, T.: Super-Sbox Cryptanalysis: Improved Attacks for AES-Like Permutations. In: Hong, S., Iwata, T. (eds.) FSE 2010. LNCS, vol. 6147, pp. 365–383. Springer, Heidelberg (2010)
17. Guo, J., Peyrin, T., Poschmann, A., Robshaw, M.: The LED Block Cipher. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 326–341. Springer, Heidelberg (2011)
18. Hart, P., Nilsson, N., Raphael, B.: A Formal Basis For The Heuristic Determination of Minimum Cost Paths. IEEE Transactions on Systems, Science, and Cybernetics SSC-4(2), 100–107 (1968)
19. ISO: ISO/IEC 10118-3:2004: Information technology — Security techniques — Hash-functions — Part 3: Dedicated hash-functions (February 2004)
20. Khovratovich, D., Biryukov, A., Nikolic, I.: Speeding up Collision Search for Byte-Oriented Hash Functions. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 164–181. Springer, Heidelberg (2009)
21. Knudsen, L.R.: Truncated and Higher Order Differentials. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 196–211. Springer, Heidelberg (1995)
22. Knudsen, L.R., Rijmen, V.: Known-Key Distinguishers for Some Block Ciphers. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 315–324. Springer, Heidelberg (2007)
23. Lamberger, M., Mendel, F., Rechberger, C., Rijmen, V., Schläffer, M.: Rebound distinguishers: Results on the full whirlpool compression function. In: [25] 126–143
24. Matsui, M.: On Correlation Between the Order of S-boxes and the Strength of DES. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 366–375. Springer, Heidelberg (1995)
25. Matsui, M. (ed.): ASIACRYPT 2009. LNCS, vol. 5912. Springer, Heidelberg (2009)

26. Miles, E., Viola, E.: Substitution-permutation networks, pseudorandom functions, and natural proofs. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 68–85. Springer, Heidelberg (2012)
27. National Institute for Science, Technology (NIST): Advanced Encryption Standard (FIPS PUB 197) (November 2001)
28. Preneel, B., Govaerts, R., Vandewalle, J.: Hash Functions Based on Block Ciphers: A Synthetic Approach. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 368–378. Springer, Heidelberg (1994)
29. Wang, X., Yu, H.: How to Break MD5 and Other Hash Functions. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)

A Best Truncated Differential Characteristics for AES-128

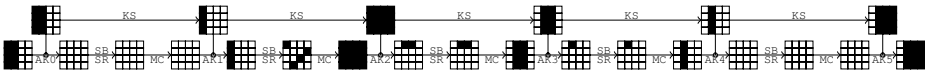


Fig. 5. Best truncated differential characteristics for AES-128 when $r = 5$ rounds with 11 active Sboxes

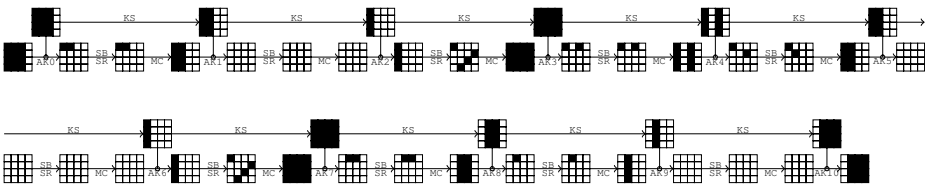


Fig. 6. Best truncated differential characteristics for AES-128 when $r = 10$ rounds with 25 active Sboxes

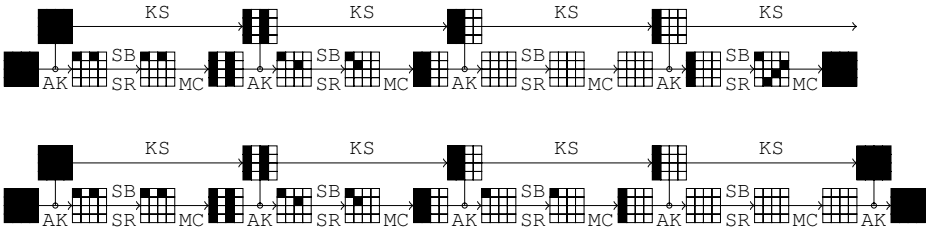


Fig. 7. Best truncated differential characteristics for AES-128 when $r = 8$ rounds with 21 active Sboxes

B Differential Characteristic for 2-Round AES-128

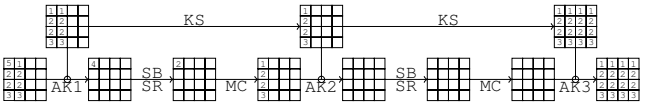


Fig. 8. The best differential characteristic on two rounds of AES-128, which has a probability $p = 2^{-6}$. The vector of differences can take as many as 2^8 values, and for instance: (1,...,5)=(0x1C,0x0E,0x12,0x01,0x1D).

C Differential Characteristic for 3-Round AES-128

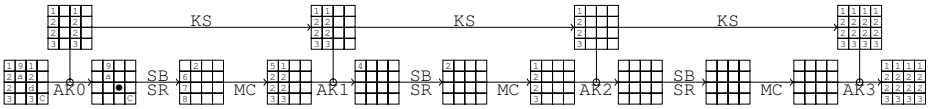


Fig. 9. The best differential characteristic on three rounds of AES-128, which has a probability $p = 2^{-31}$. The vector of differences is (1,...,d)=(0x1C,0x0E,0x12,0x01,0x1D,0x90,0x0D,0x0B,0xB3,0x58, 0x45,0xF7,0x4B).

D Differential Characteristic for 4-Round AES-128

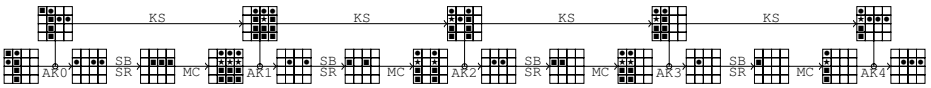


Fig. 10. The first best differential characteristic on four rounds of AES-128, which has a probability $p = 2^{-81}$. Differences are: ■ = 0x7A, ● = 0x8E and ★ = ■ ⊕ ● = 0xF4.

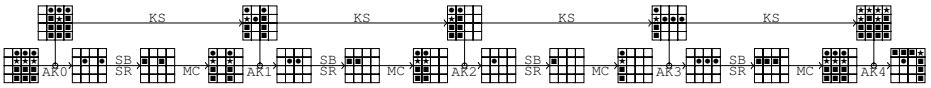


Fig. 11. The second best differential characteristic on four rounds of AES-128, which has a probability $p = 2^{-81}$. Differences are: ■ = 0x7A, ● = 0x8E and ★ = ■ ⊕ ● = 0xF4.

E Differential Characteristic for 5-Round AES-128

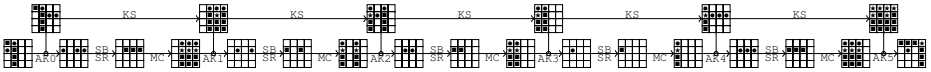


Fig. 12. The best differential characteristic on five rounds of AES-128, which has a probability $p = 2^{-105}$. Differences are: ■ = 0x7A, ● = 0x8E and ★ = ■ ⊕ ● = 0xF4.

Bounds in Shallows and in Miseries*

Céline Blondeau[1], Andrey Bogdanov[2], and Gregor Leander[3]

[1] Aalto University, School of Science, Finland
celine.blondeau@aalto.fi
[2] Technical University of Denmark, Denmark
anbog@dtu.dk
[3] Ruhr University Bochum, Germany
gregor.leander@rub.de

Abstract. Proving bounds on the *expected differential probability* (EDP) of a characteristic over all keys has been a popular technique of arguing security for both block ciphers and hash functions. In fact, to a large extent, it was the clear formulation and elegant deployment of this very principle that helped Rijndael win the AES competition. Moreover, most SHA-3 finalists have come with explicit upper bounds on the EDP of a characteristic as a major part of their design rationale. However, despite the pervasiveness of this design approach, there is no understanding of what such bounds actually mean for the security of a primitive once a key is fixed — an essential security question in practice.

In this paper, we aim to bridge this fundamental gap. Our main result is a quantitative connection between a bound on the EDP of differential characteristics and the highest number of input pairs that actually satisfy a characteristic for a fixed key. This is particularly important for the design of permutation-based hash functions such as sponge functions, where the EDP value itself is not informative for the absence of rekeying. We apply our theoretical result to revisit the security arguments of some prominent recent block ciphers and hash functions. For most of those, we have good news: a characteristic is followed by a small number of pairs only. For Keccak, though, currently much more rounds would be needed for our technique to guarantee any reasonable maximum number of pairs.

Thus, our work — for the first time — sheds light on the fixed-key differential behaviour of block ciphers in general and substitution-permutation networks in particular which has been a long-standing fundamental problem in symmetric-key cryptography.

Keywords: block cipher, hash function, differential cryptanalysis, differential characteristic, expected differential probability, Grøstl.

* *"There is a tide in the affairs of men / Which, taken at the flood, leads on to fortune; / Omitted, all the voyage of their life / **Is bound in shallows and in miseries**. / On such a full sea are we now afloat; / And we must take the current when it serves, / Or lose our ventures"*. The Tragedy of Julius Caesar by William Shakespeare. Act 4, Scene 3.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 204–221, 2013.

1 Introduction

Block Ciphers and Hash Functions. Block ciphers and hash functions are at the very core of cryptography today, being accountable for absolutely most data encryption and authentication occurring in the field. It is not by accident that block ciphers (AES) and hash functions (SHA) are among the few cryptographic algorithms standardized by NIST, the U.S. National Institute of Standards and Technology. The security properties of block ciphers and hash functions are largely interconnected. The traditional way of building a hash function has been to employ a block cipher in a mode of operation, such as Davies-Meyer, Matyas-Meyer-Oseas, Miyaguchi-Preneel or Hirose. Rather lately, it has become popular to build hash functions from permutations which are usually obtained by fixing a key in a well-understood block cipher. While SHA-1 and SHA-2 conform to the former design principle (having the SHACAL block ciphers at their foundation), SHA-3 (Keccak) — building upon the sponge construction [3] — adopts the latter one. In the paper, we will deal with the cryptographic fixed-key properties of block ciphers (or, equivalently, with properties of permutations) — a research field that, rather unduly as we think, has not received much attention recently.

Differential Cryptanalysis, Differential Characteristics, Probabilities. Any sound newly developed block cipher or hash function comes with strong arguments against *differential cryptanalysis*. It was introduced in 1990 by Biham and Shamir [7] for recovering the key of round-reduced DES — the former U.S. Data Encryption Standard. Later they used it to attack the full DES [8]. Differential cryptanalysis was known to the designers of DES (IBM with NSA involvement) back in the 1970s though [15]. Based on the seminal idea of differential cryptanalysis, plenty of extensions have been proposed [5,12,27,28,35]. In fact, it was a variant of differential cryptanalysis that resulted in the first key recovery for the full AES, though in a weak related-key model [9].

Since its publication for the case of DES, differential cryptanalysis has been applied to numerous *iterative block ciphers*, that is, block ciphers whose data transform consists of consecutive application of similar simpler maps (*rounds*).

In the differential cryptanalysis context, given a pair of inputs to the cipher with a certain difference Δ_0, one tracks the propagation of this difference through the r round transforms resulting in intermediate differences Δ_i, $i = 1, \ldots, r-1$, and an output difference Δ_r. The sequence of all these $r+1$ differences $(\Delta_0, \ldots, \Delta_r)$ is called a *differential characteristic*. In a block cipher, once the key K is fixed, the fixed-key *differential characteristic probability* (DP) π_K is the probability for a pair of inputs to follow this differential characteristic. When π_K is averaged over all round keys, one obtains the *expected differential characteristic probability* (EDP) π. The paper at hand contributes to the fundamental understanding of the links between DP and EDP.

Hypothesis of Stochastic Equivalence and Plateau Characteristics. As regards the connection between the fixed-key DP π_K and the EDP π for a characteristic in a cipher, the common *hypothesis of stochastic equivalence* [29] states

that $\pi \approx \pi_K$ for almost all keys. However, there is an essential gap between EDP π and DP π_K, since there can be a significant discrepancy between those values. Probably the most prominent example of a strictly non-equivalent behaviour is actually constituted by the AES and its *plateau characteristics*, that is, differential characteristics that, depending on the key, have a probability of either 0 or 2^{h-n}, where h is the *height* of the plateau characteristic and n is the block size. One presumes that most characteristics over the full AES are plateau of height 1 [21,33] with $n = 128$. At the same time, it is well known that the value of EDP for, say, AES-128 does not exceed 2^{-330}.

Bounds on Differential Characteristic Probability as Security Argument. While the computation of the DP value or even its informative upper-bounding is known to be a difficult problem in symmetric-key cryptography, it turns out that it is possible to compute an upper bound on the value of EDP for a characteristic — at least for some suitable ciphers and under some assumptions. Both block cipher and hash function designers tend to compute such a bound on the EDP whenever possible. This bound is widely accepted as a valid security argument not only for block ciphers [14, 18, 26] but also for hash functions [2, 4, 6, 24, 36].

The starting point of computing an upper bound on EDP is the *Markov cipher* assumption which requires the iterative cipher to be such that the transition probability $\pi_{\Delta_i,\Delta_{i+1}}$ for differences $\Delta_i \to \Delta_{i+1}$ over a round does not depend on the actual input value, where the probability is taken over the keys. Under the Markov cipher assumption and if round keys are independent, it can be shown [29] that the product of all transition probabilities equals the EDP, $\pi = \prod_i \pi_{\Delta_i,\Delta_{i+1}}$.

This approach gives the designers of new block ciphers and hash functions a formal way of arguing the resistance of their primitives towards differential cryptanalysis. Eventually, it took the symmetric-key community several years since the introduction of differential cryptanalysis to come up with the paradigm that finally manifested itself as a winning approach and stipulated the spread of substitution-permutation networks: *the wide trail design strategy* [19] (*decorrelation theory* [34] being another example of a similar but somewhat more general approach). Here, one builds a primitive such that the minimum number of *active S-boxes* (i.e. nonlinear components with nonzero input and output differences in a differential characteristic) over all nontrivial differential characteristics is maximized. Then an upper bound on the difference propagation probability through an S-box is used to compute the EDP for the full characteristic. In fact, to a large extent, it was the clear formulation and elegant deployment of the wide trail design strategy that helped Rijndael win the NIST AES competition.

The application of such approaches is by far not limited to block ciphers though. Also three out of five SHA-3 finalists (Keccak [4], Grøstl [24] and JH [36]) – including the SHA-3 winner – have come with wide-trail type security arguments, providing some bounds on the EDP. However, the exact meaning – quantitative or even qualitative – of these bounds appears to have been unclear.

Criticism of the Expected Differential Characteristic Probability. Indeed, while an upper bound on EDP does contain information on the behaviour of a fixed-key differential characteristic on average, interpreting it can be rather confusing, even for block ciphers. It is not clear what such a bound says when the key is fixed, like it is the case in almost any block cipher based encryption procedure or in permutation based hash functions.

As already mentioned above, an upper bound on EDP can get rather low. For instance, consider the permutation P (or Q) of Grøstl-256. This is a permutation on 512 bits and the designers show that, under the round independency assumption, any differential characteristic has a probability of less than 2^{-972}, by using the wide-trail design strategy. Since 2^{511} (unordered) pairs are possible, this *might seem* to indicate that any given characteristic is fulfilled by zero pairs. But then, it is trivial to find many characteristics that are fulfilled by at least one pair – just take two inputs with some difference Δ_I, apply the permutation P to it and note down the intermediate differences Δ_i after each round.

Of course, this situation is not specific to Grøstl since similar arguments are provided by, among many others, the designers of the SHA-3 winner Keccak, SHA-3 finalist JH as well as lightweight hash functions such as SPONGENT [13] and PHOTON [25]. Here, lower bounds on the number of differentially active S-boxes can only be translated to upper bounds on the *expected* differential characteristic probability (in other words, on the *expected* number of pairs following a characteristic), averaged over *all keys*. At the same time, such designs rely on a permutation which is a substitution-permutation network for a *fixed key*, usually supplied in form of round constants. The assumption that the rounds are independent can only be argued strictly if we have independent round keys. Here this is clearly not the case as *the key (resp. the round constants) is fixed*.

The Motivation. The question of what bounds on expected differential characteristic probabilities actually imply for permutation-based hash functions (like Keccak, Grøstl and JH as well as many more recent hash functions such as SPONGENT and PHOTON) – or even for a block cipher with a fixed key – remains unanswered. Thus, there is a fundamental lack of understanding of what those bounds mean. The significance of this problem is emphasized by the large number of designs that use such bounds without discussing their impact. So, given that there will always be characteristics that are fulfilled by at least one pair, what can we hope for? From a designer's point of view — focusing on differential characteristics — a reasonable goal will be that with high probability, there is no characteristic that is fulfilled by more than one pair.

Now, the critical question is: How small should a bound be on the EDP of a differential characteristic to guarantee this goal above? Concretely, in the example of Grøstl-256, is 2^{-972} enough, too big or already far too small?

Somewhat more specifically, the research problem we aim to address in this work is the following:

Given an upper bound on the expected differential characteristic probability (EDP) π for a block cipher over all keys, what is the probability to have at most B pairs of inputs following a differential characteristic for one fixed key?

Our Main Contribution. In this paper, we answer this question formally and shed some light on what those bounds mean for constructions with a fixed key or a fixed constant. The only assumption we are making in our work is that the number of (unordered) pairs that satisfy a given characteristic follows a binomial distribution over all possible keys (resp. round constants).

Now we formulate our main result. Let B be a bound on the number of pairs of input that fulfill a differential characteristic in a block cipher (or a permutation), once the key is fixed. Let q_B be the probability that all nontrivial characteristics are fulfilled by at most B pairs of inputs. Recall that we denote the block size in bits by n. The main result of our paper is summarized in the next theorem.

Theorem (Main Result). *If π is an upper bound on the expected differential characteristic probability (EDP) for a block cipher (or a permutation) over all keys, the probability q_B that all nontrivial characteristics are fulfilled by at most B input pairs is lower-bounded by:*

$$q_B \geq 1 - \frac{\pi^B}{(B+1)!2^B} 2^{(B+2)n}.$$

Given this result, one can now obtain the greatest sufficient value of an upper bound on EDP to achieve the design goal of a permutation having at most $B = 1$: An upper bound on π of 2^{-3n-7} or lower suffices to attain this goal with probability $q_1 \geq 0.99$. Using this theorem reciprocally for already existing designs, we can state, for instance, that the designers' upper bound of $\pi = 2^{-972}$ on the EDP for the P or Q permutations in Grøstl is sufficient to have at most 3 pairs with a probability of at least $q_3 = 1 - 2^{-363.58}$.

Though we do not consider the EDP and DP of differentials, our work does shed light on a fundamental, previously ignored, problem in the design of (round-based) fixed permutations. The only requirement to apply our result is to provide a bound on the EDP of a characteristic, which is the only indicator of security against differential cryptanalysis that designers are usually able to give.

2 Preliminaries

In this section, we introduce our notation and subsequently the statistical model along with its single assumption. We want our model to deal with all differential characteristics of a cipher. For this purpose, we introduce what we coin as the *differential characteristic spectrum* of a cipher. In a nutshell, this is a list of probabilities of characteristics along with their quantity. Note that, while this spectrum is very suitable to model the differential behaviour of a given cipher, for any real-world cipher it is completely out of reach to compute this spectrum. We therefore will later, cf. Section 3.1, explain how to bypass the need to compute the entire spectrum. In this sense, the introduction of the spectrum is a way of eliminating most of it in a sound manner.

2.1 The Model

Let n denote the bit size of the primitive (permutation size or block size). Let p denote the probability of a differential characteristic and X_p denote the random variable (taken over independent round keys) which corresponds to the number of pairs that fulfill a differential characteristic with probability p. Note that we are always taking the *whole input space* into consideration. In order to simplify the treatment, we furthermore always considered *unordered pairs*. The sole assumption underlying our model, and thus our results, is that X_p follows a binomial distribution. More precisely:

Assumption 1. *X_p follows (over the independent round keys) a binomial distribution with parameters (N, p), i.e.*

$$X_p \sim \mathcal{B}(N, p)$$

where $N = 2^{n-1}$ is the total number of unordered pairs with a fixed difference.

This assumption has been used frequently in the literature, cf. Section 2.2 for more details. The only class of characteristics that clearly do not follow a binomial distribution we are aware of are plateau characteristics, most prominently present in the AES [21,22,33]. We discuss plateau characteristics in Appendix A and explain in which cases our result extends to those characteristics as well.

Now, as outlined above, we do not only deal with a single characteristic, but with all characteristics at the same time. There are usually characteristics with many different probabilities, and conversely many characteristics for a given probability. Following [10] for a similar concept, we capture this information in what we refer to as the *differential spectrum* of a cipher, cf. also the characteristic weight counting function of [20]. For this, denote by $(p_i)_i$ the set of all occurring probabilities and by A_i the number of characteristics with probability p_i. For convenience we assume that $p_i \geq p_{i+1}$, i.e. the probabilities are ordered in descending order. Note that we explicitly exclude the trivial characteristics, i.e. the characteristic with input difference 0.

Definition 1 (Differential Spectrum). *The vector of pairs $\mathcal{S} = ((p_i, A_i))_i$ is called the* differential spectrum *of a cipher.*

The complete list of differential characteristics is modeled, according to Assumption 1, by random variables $X_j^{(p_i)} \sim \mathcal{B}(p_i, 2^{n-1})$, where $1 \leq j \leq A_i$. Clearly, it holds that

$$\sum_i p_i A_i = 2^n - 1 \tag{1}$$

simply as every pair follows some (non-trivial) characteristic. Actually, even more is true, namely

$$\Pr\left(\sum_i \sum_j X_j^{(p_i)} = 2^{n-1}(2^n - 1)\right) = 1.$$

From this perspective it seems reasonable to assume that the vector $(X_j^{(p_i)})_{i,j}$ is multinomial distributed. However, this way one would assume that no other dependency between the individual characteristic exists. This is especially doubtable in the case of characteristics that are identical for a large number of rounds and only diverge in the very last (or first) round.

Let us turn to our main focus, i.e. studying the question of what the maximal number of pairs is that follow a characteristic. In the above model, this corresponds to studying the distribution of the random variable $Z^{\mathcal{S}}$ defined as

$$Z^{\mathcal{S}} = \max_{i,j}\{X_j^{(p_i)}\}. \tag{2}$$

The relevance of $Z^{\mathcal{S}}$ is the following. If, for example, we can argue (within our model) that $\Pr(Z^{\mathcal{S}} \leq B) = 0.99$, we are guaranteed that choosing random round constants (resp. round keys), will result in 99 out of 100 cases in a permutation such that no characteristic is followed by more than B pairs.

In particular, for ensuring that any characteristic is fulfilled by at most one pair, $\Pr(Z^{\mathcal{S}} \leq 1)$ should be close to one.

In the sequel, as in Assumption 1, $N = 2^{n-1}$ will denote the number of unordered pairs with a fixed difference. Furthermore, B will denote the bound on the number of pairs we consider and $q_B = P(Z^{\mathcal{S}} \leq B)$ denotes the probability that no characteristic is fulfilled by more than B pairs. For a given spectra we denote $p = p_0$, i.e. the maximal probability of a characteristic (or an upper bound on it).

2.2 Binomial Distribution

Since differential cryptanalysis is one of the major cryptanalytic techniques in symmetric-key cryptography, the distributions of random variables associated with differential characteristics and differentials have been extensively studied over the past 20 years. In [1], examples with different EDP and fixed-key DP are considered and it is noted that computing the average values does not in general allow one to draw conclusions about the shape of the distributions. In a similar direction, the work [23] develops a model derived from binomial distribution and performs experiments in the case of a random permutation. Moreover, it provides instances of ciphers for which this model holds and does not hold. In [22], it is shown that for only 2 rounds of the AES, this model is not correct due to the existence of plateau characteristics (see the discussion in Appendix A).

In [11], experiments show that — at least for some relevant ciphers — the distribution is actually binomial, that is, as stated in Assumption 1. To establish a clear independent empirical basis for our theoretical study and to support the meaningfulness of the assumption, we conducted experiments of our own on a 16-bit reduced version of PRESENT [14]. The experiments depicted in Figure 1 correspond to 5, 6 and 7 rounds of SmallPresent: As the number of rounds increases and the EDP of the best characteristic decreases, the deviation from the binomial distribution is almost non-existant and clearly indicates that Assumption 1 is realistic in that case.

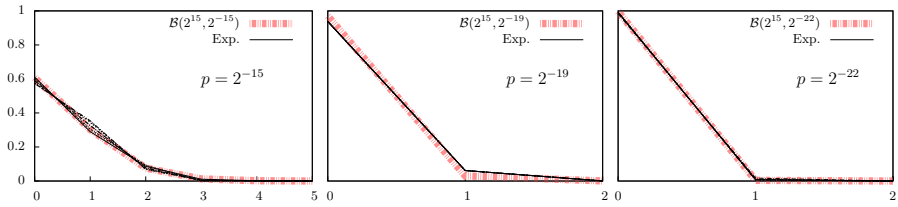


Fig. 1. Distribution for the characteristics of SmallPresent (n=16): Comparison between the distribution of different characteristics with EDP p and the binomial distribution $\mathcal{B}(2^{n-1}, p)$.

3 The Link: Bounding the Bound

In this section, we present and prove the main result of this work. First, we show how to avoid computing the whole spectrum and still be able to make useful statements about the distribution of $Z^{\mathcal{S}}$, as introduced in (2). Second, we provide the proof of our main theorem outlined in the introduction.

3.1 Cutting the Spectrum of a Cipher

As introduced in Section 2, we assume that the number of pairs fulfilling a given characteristic with a certain probability follows a binomial distribution. As we are interested in primitives where no characteristic is satisfied by many pairs, we consider the bound on the EDP which is smaller or equal to 2^{-n}, i.e. $p = p_0 \leq 2^{-n}$ in the sequel.

For a fixed B, we show, in this section, that an underestimate of the $q_B = P(Z^{\mathcal{S}} \leq B)$ can be obtained with partial knowledge of the spectrum. More precisely, we study what we call a *cut* of a given spectrum.

Definition 2. *Given a spectrum* $\mathcal{S} = (p_i, A_i)_{i=0}^{w}$ *we define a* cut spectrum *of order* t $(1 \leq t \leq w)$ *by* $\mathcal{S}_t = (p_i, A'_i)_{i=0}^{t}$ *where*

$$A'_i = A_i \text{ for } 0 \leq i \leq t-1$$

and

$$A'_t = \frac{(2^n - 1) - \sum_{i=0}^{t-1} A_i p_i}{p_t}.$$

The cut spectrum *of order* 0 *is defined by* $\mathcal{S}_0 = (p_0, A'_0)$ *with* $A'_0 = \frac{2^n - 1}{p_0}$.

Thus, by studying a cut of order t of a spectrum, one assumes that all characteristics with probability less than p_{t-1} actually have probability p_t. The definition of A'_t then follows directly from (1).

The following theorem shows that, by studying cuts of a given spectrum, we obtain an underestimate of the cumulative distribution of $Z^{\mathcal{S}}$, cf. (2). While this might be intuitively reasonable, strictly proving it is rather technical, as we will see below.

Note that all results presented here hold independently of any possible correlation between the individual characteristics. Depending of their relations, tightness of the results can differ but the results remain valid.

Theorem 1. *Given a spectrum $\mathcal{S}$ and its cuts $(\mathcal{S}_t)_t$, the following holds for any $B \geq 1$.*

$$q_B = \Pr(Z^{\mathcal{S}} \leq B) \geq \Pr(Z^{\mathcal{S}_t} \leq B) \geq \cdots \geq \Pr(Z^{\mathcal{S}_0} \leq B).$$

In order to prove the statement of the theorem, we first introduce two technical lemmata.

Lemma 1. *Let $2^{-n} \geq p_1 \geq p_2$ be two probabilities. For all, $i \geq 2$ we have*

$$p_1^{i-1}(1-p_1)^{N-i} - p_2^{i-1}(1-p_2)^{N-i} \geq 0.$$

Proof. To prove this inequality, we shall prove that

$$T = \left(\frac{p_1}{p_2}\right)^{i-1} \left(\frac{1-p_1}{1-p_2}\right)^{N-i} \geq 1.$$

Assuming that $p_1 \geq p_2$, we have

$$\frac{p_1 - p_2}{p_1} \leq -\log(1 - \frac{p_1 - p_2}{p_1}) = \log(\frac{p_1}{p_2}).$$

Now the proof follows by a succession of inequalities:

$$\begin{aligned}
& N \leq \frac{1}{2p_1} \leq \frac{1-p_1}{p_1 - p_2}\frac{p_1 - p_2}{p_1} \\
\Rightarrow\ & (N-i) \leq N \leq \frac{1-p_1}{p_1 - p_2}\log(p_1/p_2) \\
\Rightarrow\ & (N-i)\frac{p_1 - p_2}{1-p_1} \leq \log(p_1/p_2) \leq (i-1)\log(p_1/p_2) \\
\Rightarrow\ & (N-i)\log(1 + \frac{p_1 - p_2}{1-p_1}) \leq (N-i)\frac{p_1 - p_2}{1-p_1} \leq (i-1)\log(p_1/p_2) \\
\Rightarrow\ & 0 \leq (i-1)\log(p_1/p_2) - (N-i)\log(\frac{1-p_2}{1-p_1}) \\
\Leftrightarrow\ & 1 \leq exp\left[(i-1)\log(p_1/p_2) + (N-i)\log(\frac{1-p_1}{1-p_2})\right] \\
\Leftrightarrow\ & 1 \leq T.
\end{aligned}$$

□

Lemma 2. *Given two random variables X_1, X_2 with $X_i \sim \mathcal{B}(N, p_i)$, $i = 1$ or 2, and $2^{-n} \geq p_1 > p_2$, for all $B \geq 1$ it holds that*

$$\frac{1}{p_1}\Pr(X_1 > B) - \frac{1}{p_2}\Pr(X_2 > B) \geq 0.$$

Proof. We consider

$$C = \frac{1}{p_1}\Pr(X_1 > B) - \frac{1}{p_2}\Pr(X_2 > B)$$
$$= \sum_{i=B+1}^{N} \binom{N}{i} \left[p_1^{i-1}(1-p_1)^{N-i} - p_2^{i-1}(1-p_2)^{N-i} \right].$$

From Lemma 1, if $B \geq 1$, we have

$$p_1^{i-1}(1-p_1)^{N-i} - p_2^{i-1}(1-p_2)^{N-i} \geq 0.$$

And we conclude that $C \geq 0$. □

Now, using these two lemmata, we can prove that the partial knowledge of the spectrum allows the computation of an underestimate of $q_B = \Pr(Z^{\mathcal{S}} \leq B)$.

Proof (Proof of Theorem 1). To simplify the notation we denote by X_i a random variable that follows a binomial distribution with parameters N and p_i, i.e. $X_i \sim \mathcal{B}(N, p_i)$.

Using the fact the probability of a union of events is smaller than the sum of the probabilities of the different events, independently of the correlation between the different variables X_i, the following holds for any B and any cut spectrum $\mathcal{S}_t$:

$$\Pr(Z^{\mathcal{S}_t} \leq B) = 1 - \Pr(Z^{\mathcal{S}_t} > B) \geq 1 - \sum_{i=0}^{t} \sum_{j=1}^{A'_i} \Pr(X_j^{(p_i)} > B)$$
$$\geq 1 - \sum_{i=0}^{t} A'_i \Pr(X_i > B)$$

For the cut spectra $\mathcal{S}_t = (p_i, A'_i)_{i=0}^{t}$ and $\mathcal{S}_{t-1} = (p_i, \Gamma'_i)_{i=0}^{t-1}$, we first prove that:

$$\forall t > 1 \quad \sum_{i=0}^{t} A'_i \Pr(X_i > B) \leq \sum_{i=0}^{t-1} \Gamma'_i \Pr(X_i > B). \tag{3}$$

Note that, as a consequence of (1), we have

$$A'_t = \frac{(2^n - 1) - \sum_{i=0}^{t-2} A'_i p_i - A'_{t-1} p_{t-1}}{p_t} \text{ and } \quad \Gamma'_{t-1} = \frac{(2^n - 1) - \sum_{i=0}^{t-2} \Gamma'_i p_i}{p_{t-1}}.$$

As for $i < t-1$ we have $A'_i = \Gamma'_i$, we obtain

$$C = \sum_{i=0}^{t} A'_i \Pr(X_i > B) - \sum_{i=0}^{t-1} \Gamma'_i \Pr(X_i > B)$$
$$= A'_{t-1}\Pr(X_{t-1} > B) + A'_t \Pr(X_t > B) - \Gamma'_{t-1}\Pr(X_{t-1} > B)$$
$$= \left[A'_{t-1} - \frac{(2^n - 1) - \sum_{i=0}^{t-2} A'_i p_i}{p_{t-1}} \right] \Pr(X_{t-1} > B) + A'_t \Pr(X_t > B)$$
$$= \left[-\frac{(2^n - 1) - \sum_{i=0}^{t-2} A'_i p_i - A'_{t-1} p_{t-1}}{p_{t-1}} \right] \Pr(X_{t-1} > B) + A'_t \Pr(X_t > B)$$
$$= p_t A'_t \left[\frac{1}{p_t}\Pr(X_t > B) - \frac{1}{p_{t-1}}\Pr(X_{t-1} > B) \right].$$

Given $p_{t-1} > p_t$, from Lemma 2 it follows that $C \leq 0$. The remaining of the proof follows then applying (3) iteratively for all t. □

Example. Figure 2 illustrates Theorem 1 for a reduced variant of PRESENT. For SMALLPRESENT [30] with a 16-bit block size, it is still feasible to compute the spectrum using a branch and bound approach. Clearly, not all characteristics have the same probability, but different probabilities occur with varying frequency.

As predicted by Theorem 1, taking only part of spectrum into account, i.e. studying $Z^{\mathcal{S}_t}$, leads to an underestimate of $q_B = \Pr(Z^{\mathcal{S}} \leq B)$. In this example, studying $\mathcal{S}_7$ already gives a rather tight estimate of the actual value of q_B.

As mentioned above, for real-world ciphers even computing such a cut spectrum is hard. Indeed, for many primitives, designers can often only provide (a bound on) the EDP of the most probable characteristic.

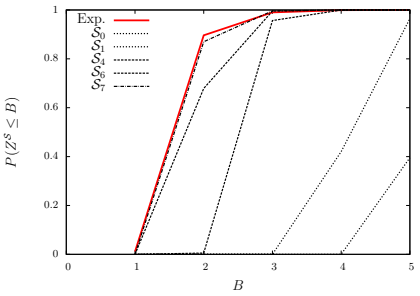


Fig. 2. Influence of cut spectra on $q_B = P(Z^{\mathcal{S}} \leq B)$: Experiments on 6 rounds of SMALLPRESENT with fixed input difference

3.2 At the Limit: Considering the Bound only

In practice, for many primitives, designers are only able to compute (a bound on) the EDP. In that case, the spectrum is cut to its maximum and is defined by $\mathcal{S}_0 = (\pi, A_0)$, where π is an upper bound (or exact value) of the EDP and, following (1), $A_0 = \frac{2^n-1}{\pi}$ is the total number of these potential characteristics. To simplify the notation, in this section the random variable associated to characteristics of this sort is denoted by X.

We recall here the main result of this paper presented in the introduction, now stated formally:

Theorem 2 (Main Result). *Under Assumption 1, if π is an upper bound on the expected differential characteristic probability (EDP) for a block cipher over all keys (or a permutation over all round constants), the probability q_B that all nontrivial characteristics are fulfilled by at most B input pairs is lower-bounded by:*

$$q_B \geq 1 - \frac{\pi^B}{(B+1)!2^B} 2^{(B+2)n}. \tag{4}$$

Proof. From Theorem 1 and (3), we have:

$$\Pr(Z^{\mathcal{S}} \leq B) \geq \Pr(Z^{\mathcal{S}_0} \leq B) \geq 1 - A_0 P(X > B) \geq 1 - \frac{2^n}{\pi} P(X > B).$$

Meaning that when only π is known, the tail of cumulative function of $Z^{\mathcal{S}}$ can be bounded by:

$$\Pr(Z^{\mathcal{S}} \leq B) \geq 1 - \frac{2^n}{\pi} \Pr(X > B). \tag{5}$$

As we have

$$\begin{aligned}\Pr(X > B) &= \sum_{i=B+1}^{N} \binom{N}{i} \pi^i (1-\pi)^{N-i} \leq \sum_{i=B+1}^{N} \binom{N}{i} \pi^i \\ &\leq \sum_{i=B+1}^{N} \frac{N^i}{(B+1)!} \pi^i \leq 2 \frac{(N\pi)^{B+1}}{(B+1)!},\end{aligned}$$

we conclude that

$$\Pr(Z^S \leq B) \geq 1 - \frac{4}{(B+1)!} N^{B+2} \pi^B.$$

□

4 The Impact: Shallows and Miseries?

In this section, we inspect the impact of our main result on the practical constructions, both block ciphers with a fixed key and permutation-based hash functions.

4.1 Sufficient Condition

From our main result stated in Theorem 2, we can deduce that the upper bound on the EDP of a characteristic in a primitive, should be of order of magnitude $\pi \approx 2^{-[(B+2)/B]n}$, in order for our result to guarantee that no characteristic is fulfilled by more than B pairs. More precisely, from Theorem 2, we immediately obtain the following estimate of π:

Corollary 1 (Sufficient Condition). *Let C_B defined by*

$$C_B = \left[(1 - q_B)(B+1)! 2^B\right]^{1/B}.$$

To guarantee with probability $q_B = P(Z^S \leq B)$ that no characteristic is fulfilled by more than B pairs, it suffices to have the maximal probability π of a characteristic such that

$$\pi \leq C_B 2^{-[(B+2)/B)]n}.$$

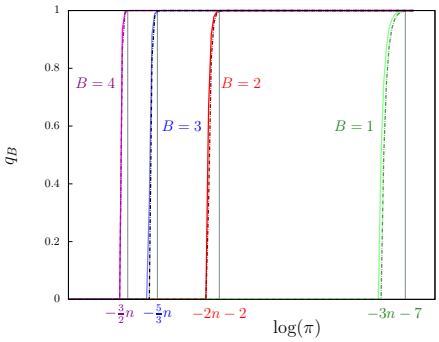


Fig. 3. Bringing it all together: q_B as a function B and EDP bound for characteristics. The dotted curves are computed with (4). The continuous curves are computed using (5) in the case where $n = 64$.

By computing the exact value of C_B introduced in Corollary 1, in order to guarantee that for 99% of the keys or fixed constants, no characteristic is fulfilled by more than one pair, we should consider permutations where the maximal EDP is lower than 2^{-3n-7}. For $B = 2$, the maximum EDP can be up to 2^{-2n-2}. For larger values of B, $C_B \approx 1$ and the same security claims can be achieved if $\pi \leq 2^{-[(B+2)/B]n}$. In Figure 3, we illustrate that the sufficient condition of Corollary 1 is rather tight. Note that though computations have been performed for $n = 64$, results are similar for larger values of n. Most importantly, Figure 3 shows that the distributions are extremely steep at around $2^{-[(B+2)/B)]n}$. Thus, a value of π slightly below this threshold does not guarantee anything any more, while a value slightly larger guarantee the non-existence of characteristics followed by more then B pairs almost certainly.

4.2 Revisiting the Security Arguments of Prominent Primitives

In here, we consider only primitives that come with informative bounds on the EDP of their differential characteristics which excludes ARX-based designs, for instance, where no efficient way of arguing tight bounds on EDP is known. However, we prominently note here that this by no means indicate per se that those designs are stronger or weaker in cryptanalytic terms. This only says that we can state much less about those constructions using the state-of-the-art techniques.

The major findings of this section are presented in Table 1. Not all bounds provided by the respective designers of the primitives mentioned in the table are actually tight. Proving better bounds on the EDP would improve the probabilities q_B for those constructions.

As it usually gets harder to provide strong diffusion with the increase of the block size, bigger variants of primitives often have a higher value of B_0. The notable exceptions are constituted by the lightweight hash functions PHOTON and SPONGENT: For PHOTON, while the maximum number of fulfilling pairs grows first with the size, it gets much lower for its biggest version, which is due

Table 1. Lower bound on the probability to have at most B pairs following a differential characteristic for various primitives. The primitives are either block ciphers with a fixed key (public or secret) or fixed permutations. n: block size. q_B: the probability for all nontrivial characteristics to be fulfilled by at most B input pairs. π: an upper bound on the the expected differential characteristic probability averaged over all keys. − means that Theorem 2 does not provide any informative indication for the parameter set. B_0 is the minimum value of B such that q_B is close to 1.

Primitives	n	π	$q_1 \geq$	$q_2 \geq$	$q_3 \geq$	B_0
AES-128	128	2^{-330}	-	$1-2^{-152.6}$	$1-2^{-357.6}$	2
AES-192	128	2^{-450}	$1-2^{-68}$	$1-2^{-392.6}$	$1-2^{-717.6}$	1
AES-256	128	2^{-480}	$1-2^{-98}$	$1-2^{-452.6}$	$1-2^{-807.6}$	1
Rijndael-192/192	192	2^{-450}	-	$1-2^{-136.6}$	$1-2^{-397.6}$	2
Rijndael-192 /256	192	2^{-480}	-	$1-2^{-196.6}$	$1-2^{-487.6}$	2
Rijndael-256	256	2^{-480}	-	-	$1-2^{-167.6}$	3
PRESENT-80/-128	64	2^{-122}	-	-	$1-2^{-53.6}$	3
LED-64	64	2^{-400}	$1-2^{-210}$	$1-2^{-548.6}$	$1-2^{-887.6}$	1
LED-128	64	2^{-600}	$1-2^{-410}$	$1-2^{-948.6}$	$1-2^{-1487.6}$	1
SPONGENT-88/80/8	88	2^{-176}	-	$1-2^{-4.6}$	$1-2^{-95.6}$	2
SPONGENT-128/128/8	136	2^{-272}	-	$1-2^{-4.6}$	$1-2^{-143.6}$	2
SPONGENT-160/160/16	176	2^{-352}	-	$1-2^{-4.6}$	$1-2^{-183.6}$	2
SPONGENT-224/224/16	240	2^{-480}	-	$1-2^{-4.6}$	$1-2^{-247.6}$	2
SPONGENT-256/256/16	272	2^{-544}	-	$1-2^{-4.6}$	$1-2^{-279.6}$	2
PHOTON-80	100	2^{-216}	-	$1-2^{-36.6}$	$1-2^{-155.6}$	2
PHOTON-128	144	2^{-294}	-	$1-2^{-16.6}$	$1-2^{-169.6}$	2
PHOTON-160	196	2^{-384}	-	-	$1-2^{-179.6}$	3
PHOTON-224	256	2^{-486}	-	-	$1-2^{-185.6}$	3
PHOTON-256	288	2^{-882}	$1-2^{-20}$	$1-2^{-616.6}$	$1-2^{-1213.6}$	1
Grøstl-224/256	512	2^{-972}	-	-	$1-2^{-363.6}$	3
Grøstl-384/512	1024	2^{-1469}	-	-	-	5
JH-224/-256/-384/-512	1024	2^{-1184}	-	-	-	13

to the special case design of the largest permutation [25]. For SPONGENT, the distribution of B_0 is very smooth because of its clearly stated design goal: the EDP bound of 2^{-2n} for an n-bit permutation [13].

SHA-3 finalists certainly deserve special treatment. The standard behavior of B_0 (its increase as n grows) is clearly visible in Grøstl. The EDP bounds are not tight for either version of Grøstl though. With the designers' bounds, one can state that it is good news for Grøstl since not more than 5 pairs can satisfy a characteristic here (and only at most 3 pairs for the smaller variant). The bound of JH is only sufficient to show a maximum of 13 satisfying pairs.

Table 1 does not contain Keccak. The reason is that the best existing bound for the 24 rounds of KECCAK-f[1600] (with a permutation size of $n = 1600$ bits) is actually only 2^{-296} [16]. So, if one aims to attain the goal of having at most two satisfying pairs for a Keccak-type permutation given this bound, 10 times more rounds (240 rounds) would be needed in KECCAK-f[1600]. To achieve $B_0 = 18$, it suffices to take 6 times more rounds (144 rounds). However, if the special case of 1- to 8-symmetric characteristics is considered, a bound of 2^{-1648} can be proven for 18 rounds [4], which leads to $B_0 = 60$ with $q_{60} = 1 - 2^{-18.1}$ for the 18 rounds. Again, we emphasize that this by no means indicates any type of weakness in Keccak. Having high upper bounds on the EDP of a differential characteristic prohibits our model from providing any informative bounds on the number of pairs following a differential characteristic.

5 Conclusion and Future Work

In this paper, we establish the fundamental link of a bound on EDP of a differential characteristic to its fixed-key DP, i.e. the maximum number of input pairs that follow a characteristic. We apply our framework to prominent hash function and block cipher designs. Once the key is fixed (which is almost always the case in practice), our result is the only formal foundation for arguing the crucial differential security properties of symmetric-key primitives available so far.

Having said that, we also clearly state some important open problems which are out of scope of this paper. First, though we constrain ourselves to considering the EDP and DP of differential characteristics here, a much more interesting object of study would be the connection between EDP and DP for *differentials*, which are sets of differential characteristics with certain input and output differences. However, since even bounding EDP for those is notoriously difficult (though not impossible, at least for several rounds of suitable constructions [17]), studying similar questions for differentials seems out of reach given the current techniques (note that [23] considered the differential behaviour of random permutation — a work that technically bears some similarities with ours). Second, though the basic differential cryptanalysis is certainly the most essential differential attack to consider, more advanced techniques such as the *rebound attack* [31] often pose a more critical threat, even in the case where the probabilities of any differential characteristic over the full permutation is very small. However, for rebound attacks, considering differential characteristics over smaller parts of a permutation makes a lot of sense: A good bound over a fraction of rounds will imply that there are only a small number of pairs satisfying a characteristic, so those values are not easy to find. Multiple inbound stages might undermine this reasoning though and it is still an open problem to argue provable security against rebound attacks. We think however that our result opens up the possibility of making at least some basic security arguments for rebound attacks where it has not been feasible so far to come up with a sound bound considerations.

We believe that those are some of the most important fundamental problems in symmetric-key cryptography open today and would like to see the results of this paper as a first step towards their solution.

References

1. Aoki, K.: On maximum non-averaged differential probability. In: Tavares, S., Meijer, H. (eds.) SAC 1998. LNCS, vol. 1556, pp. 118–130. Springer, Heidelberg (1999)
2. Benadjila, R., Billet, O., Gilbert, H., Macario-Rat, G., Peyrin, T., Robshaw, M., Seurin, Y.: SHA-3 Proposal: ECHO (2010) (version 2.0)
3. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Sponge functions. In: Ecrypt Hash Workshop 2007 (2007)
4. Bertoni, G., Daemen, J., Peeters, M., van Assche, G.: The Keccak SHA-3 submission (2011) (version 3)
5. Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 12–23. Springer, Heidelberg (1999)
6. Biham, E., Dunkelman, O.: The SHAvite-3 Hash Function (2009) (tweaked version)
7. Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 2–21. Springer, Heidelberg (1991)
8. Biham, E., Shamir, A.: Differential Cryptanalysis of the Full 16-Round DES. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 487–496. Springer, Heidelberg (1993)
9. Biryukov, A., Khovratovich, D., Nikolić, I.: Distinguisher and Related-Key Attack on the Full AES-256. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 231–249. Springer, Heidelberg (2009)
10. Blondeau, C., Canteaut, A., Charpin, P.: Differential properties of power functions. International Journal of Information and Coding Theory 1(2), 149–170 (2010)
11. Blondeau, C., Gérard, B.: Links Between Theoretical and Effective Differential Probabilities: Experiments on PRESENT. In: Ecrypt Workshop on Tools for Cryptanalysis (June 2010)
12. Blondeau, C., Gérard, B., Nyberg, K.: Multiple Differential Cryptanalysis Using LLR and χ^2 Statistics. In: Visconti, I., De Prisco, R. (eds.) SCN 2012. LNCS, vol. 7485, pp. 343–360. Springer, Heidelberg (2012)
13. Bogdanov, A., Knezevic, M., Leander, G., Toz, D., Varici, K., Verbauwhede, I.: SPONGENT: A Lightweight Hash Function. In: Preneel, Takagi (eds.) [32], pp. 312–325
14. Bogdanov, A.A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M., Seurin, Y., Vikkelsoe, C.: PRESENT: An ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007)
15. Coppersmith, D.: The Data Encryption Standard (DES) and its strength against attacks. IBM Journal of Research and Development 38(3), 243–250 (1994)
16. Daemen, J., Van Assche, G.: Differential Propagation Analysis of Keccak. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 422–441. Springer, Heidelberg (2012)

17. Daemen, J., Lamberger, M., Pramstaller, N., Rijmen, V., Vercauteren, F.: Computational aspects of the expected differential probability of 4-round AES and AES-like ciphers. Computing 85(1-2), 85–104 (2009)
18. Daemen, J., Rijmen, V.: The Block Cipher Rijndael. In: Schneier, B., Quisquater, J.-J. (eds.) CARDIS 1998. LNCS, vol. 1820, pp. 277–284. Springer, Heidelberg (2000)
19. Daemen, J., Rijmen, V.: The Wide Trail Design Strategy. In: Honary, B. (ed.) Cryptography and Coding 2001. LNCS, vol. 2260, pp. 222–238. Springer, Heidelberg (2001)
20. Daemen, J., Rijmen, V.: Probability distributions of Correlation and Differentials in Block Ciphers. IACR Eprint Report 2005/212 (2005)
21. Daemen, J., Rijmen, V.: Understanding Two-Round Differentials in AES. In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 78–94. Springer, Heidelberg (2006)
22. Daemen, J., Rijmen, V.: Plateau characteristics. Iet Information Security 1, 11–17 (2007)
23. Daemen, J., Rijmen, V.: Probability distributions of correlation and differentials in block ciphers. J. Mathematical Cryptology 1(3), 221–242 (2007)
24. Gauravaram, P., Knudsen, L.R., Matusiewicz, K., Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: Grøstl – A SHA-3 candidate (2011)
25. Guo, J., Peyrin, T., Poschmann, A.: The PHOTON Family of Lightweight Hash Functions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 222–239. Springer, Heidelberg (2011)
26. Guo, J., Peyrin, T., Poschmann, A., Robshaw, M.J.B.: The LED Block Cipher. In: Preneel, Takagi (eds.) [32], pp. 326–341
27. Jakimoski, G., Desmedt, Y.: Related-Key Differential Cryptanalysis of 192-bit Key AES Variants. In: Matsui, M., Zuccherato, R.J. (eds.) SAC 2003. LNCS, vol. 3006, pp. 208–221. Springer, Heidelberg (2004)
28. Knudsen, L.R.: Truncated and Higher Order Differentials. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 196–211. Springer, Heidelberg (1995)
29. Lai, X., Massey, J.L., Murphy, S.: Markov Ciphers and Differential Cryptanalysis. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 17–38. Springer, Heidelberg (1991)
30. Leander, G.: Small Scale Variants of The Block Cipher PRESENT. IACR Cryptology ePrint Archive 2010, 143 (2010)
31. Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: The Rebound Attack: Cryptanalysis of Reduced Whirlpool and `Grøstl`. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 260–276. Springer, Heidelberg (2009)
32. Preneel, B., Takagi, T. (eds.): CHES 2011. LNCS, vol. 6917. Springer, Heidelberg (2011)
33. Rijmen, V., Toz, D., Varici, K.: On the Four-Round AES Characteristics. In: Preproceedings of WCC 2013, Bergen, Norway, April 15-19, pp. 315–328 (2013)
34. Vaudenay, S.: Decorrelation: A theory for block cipher security. J. Cryptology 16(4), 249–286 (2003)
35. Wagner, D.: The Boomerang Attack. In: Knudsen, L.R. (ed.) FSE 1999. LNCS, vol. 1636, pp. 156–170. Springer, Heidelberg (1999)
36. Wu, H.: The Hash Function JH (2011)

A Plateau Characteristics and Our Model

Especially for the AES, the existence of so called *plateau characteristics* (introduced in [21, 22]) is a well studied phenomena. Basically, a given characteristic is plateau if there are only two possible values, 0 and 2^{h-1} for some positive integer h, for the number of pairs following this characteristics. In other words, the number of right pairs following a plateau characteristic with EDP p can be modeled by a random variable Y satisfying

$$Y = \begin{cases} 2^{h-1} & \text{with probability } p2^{n-h} \\ 0 & \text{with probability } (1-p)2^{n-h}\,. \end{cases}$$

The value of h is called the height of the characteristic[1].

Plateau characteristics for two rounds of AES are well understood. In particular it has been shown that for two rounds plateau characteristics of height up to 5 exist. Recently, in [33] some four-round plateau characteristics with height greater than one have been presented for AES. We are not aware of any results for more than 4 rounds of AES.

Note that a plateau characteristic of height one for a round-reduced variant of the primitive, trivially extends to any number of rounds simply because the number of right pairs never increases as the number of rounds grows. This is not (in general) the case for plateau characteristic of height greater than one.

Clearly, a plateau characteristic does not follow a binomial distribution. Even more, in the case where the height h is greater than one, the binomial distribution clearly underestimates the probability of the characteristic to be fulfilled by 2^{h-1} or more pairs. Thus, in this case our model does not fit as is.

However, *plateau characteristics of height 1* actually do not pose a problem for our model. In this case, the characteristic is never fulfilled by more than one pair. By assuming a binomial distribution in our model, we therefore *overestimate* the probability of having more than one right pair.

Intuitively, plateau characteristics of height greater than one for all rounds of AES (or similar constructions) seem unlikely. However, until now their existence cannot be excluded an, thus, some doubts on the unrestricted applicability of our model remain.

Finally note that our model can tolerate plateau characteristics of height greater than one as long as they are not too frequent. More precisely, assume the existence of A plateau characteristics of height h with EDP below p. In this case the probability that at least one of them is fulfilled by 2^{h-1} pairs is upper bounded by $Ap2^{n-h}$. Thus is A is sufficiently smaller than $\frac{2^{h-n}}{p}$ with good probability all those plateau characteristics are fulfilled by zero pairs and thus do not affect the validity of our bounds.

[1] The "-1" in 2^{h-1} stems from the fact that we consider unordered pairs, i.e. the total number of pairs is $N = 2^{n-1}$ while [22] considers ordered pairs.

Sieve-in-the-Middle: Improved MITM Attacks*

Anne Canteaut[1], María Naya-Plasencia[1], and Bastien Vayssière[2]

[1] Inria Paris-Rocquencourt, Project-Team SECRET, France
{Anne.Canteaut,Maria.Naya_Plasencia}@inria.fr,
[2] Université de Versailles Saint-Quentin-en-Yvelines, France
bastien.vayssiere.w@gmail.com

Abstract. This paper presents a new generic technique, named sieve-in-the-middle, which improves meet-in-the-middle attacks in the sense that it provides an attack on a higher number of rounds. Instead of selecting the key candidates by searching for a collision in an intermediate state which can be computed forwards and backwards, we look for the existence of valid transitions through some middle sbox. Combining this technique with short bicliques allows to freely add one or two more rounds with the same time complexity. Moreover, when the key size of the cipher is larger than its block size, we show how to build the bicliques by an improved technique which does not require any additional data (on the contrary to previous biclique attacks). These techniques apply to PRESENT, DES, PRINCE and AES, improving the previously known results on these four ciphers. In particular, our attack on PRINCE applies to 8 rounds (out of 12), instead of 6 in the previous cryptanalyses. Some results are also given for theoretically estimating the sieving probability provided by some inputs and outputs of a given sbox.

Keywords: Meet-in-the-middle, bicliques, sbox, matching algorithms.

1 Introduction

Meet-in-the-middle (MITM) attacks are a widely used tool introduced by Diffie and Hellman in 1977. Through the years, they have been applied for analyzing the security of a substantial number of cryptographic primitives, including block ciphers, stream ciphers and hash functions, e.g. [20,5,12,15,14]. They exploit the fact that some internal state in the middle of the cipher can be computed both forwards from the plaintext and backwards from the ciphertext, and that none of these computations requires the knowledge of the whole master key. The attacker then only keeps the (partial) key candidates which lead to a collision in that internal state and discards all the other keys. This generic attack has drawn a lot of attention and raised many improvements, including the partial matching, where the computed internal states are not completely known, the technique of guessing some bits of the internal state [12], the all-subkeys approach [15], splice-and-cut [2,3,13] and bicliques [18]. The most popular application of bicliques is an

* Partially supported by the French Agence Nationale de la Recherche through the BLOC project under Contract ANR-11-INS-011.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 222–240, 2013.

accelerated exhaustive search on the full AES [4]. But, besides this degenerated application where the whole key needs to be guessed, short bicliques usually allow to increase the number of rounds attacked by MITM techniques without increasing the time complexity, but with a higher data complexity. Moreover, following [7], low-data attacks have attracted a lot of attention, motivated in part by the fact that, in many concrete protocols, only a few plaintext-ciphertext pairs can be eavesdropped. MITM attacks belong to this class of attacks in most cases (with a few exceptions like bicliques): usually, 1 or 2 known plaintext-ciphertext pairs are enough for recovering the key.

Our Contribution. This paper first provides a new generic improvement of MITM algorithms, named *sieve-in-the-middle*, which allows to attack a higher number of rounds. Instead of looking for collisions in the middle, we compute some input and output bits of a particular middle sbox S. The merging step of the algorithm then consists in efficiently discarding all key candidates which do not correspond to a valid transition through S. Intuitively, this technique allows to attack more rounds than classical MITM since it also covers the rounds corresponding to the middle sbox S (e.g. two middle rounds if S is a superbox). This new improvement is related to some previous results, including [2] where transitions through an ARX construction are considered; a similar idea was applied in [17] in a differential attack, and in [8] for side-channel attacks. This new generic improvement can be combined with bicliques, since short bicliques also allow to add a few rounds without increasing the time complexity. But, the price to pay is a higher data complexity. Here, we show that this increased data requirement can be avoided by constructing some improved bicliques, if the key size of the cipher is larger than its block size.

These new improvements and techniques are illustrated with four applications which improve previously known attacks. In Section 4, the sieve-in-the-middle algorithm combined with the improved biclique construction is applied to 8 rounds (out of 12) of PRINCE, with 2 known plaintext-ciphertext pairs, while the previous best known attack was on six rounds. Due to the page limitation, the other three applications are presented in the full version of this paper [9] only. In [9], we describe a sieve-in-the-middle attack on 8 rounds of PRESENT, which provides a very illustrative and representative example of our technique. This attack applies up to 8 rounds, while the highest number of rounds reached by classical MITM is only 6. A similar analysis on DES is presented in [9]; our attack achieves 8 rounds, while the best previous MITM attack (starting from the first one) was on 6 rounds. The cores of these two attacks have been implemented, confirming our theoretical analysis. In [9], we also show that we can slightly improve on some platforms the speed-up factor in the accelerated exhaustive search on the full AES performed by bicliques. The time complexity of the sieve-in-the-middle algorithm highly depends on the sieving probability of the middle sbox, i.e., on the proportion of pairs formed by a partial input and a partial output which correspond to a valid transition for S. We then give some results which allow to estimate the sieving probability of a given sbox. In particular, we show that the sieving probability is related to the branch number

of the sbox, and we give a lower bound on the minimal number of known input and output bits which may provide a sieve.

2 The Sieve-in-the-Middle Attack

2.1 Basic Idea

The basic idea of the attack is as follows. The attacker knows one pair of plaintext and ciphertext (P, C) (or several such pairs), and she is able to compute from the plaintext and from a part K_1 of the key candidate an m-bit vector u, which corresponds to a part of an intermediate state x. On the other hand, she is able to compute from the ciphertext and another part K_2 of the key candidate a p-bit vector v, which corresponds to a part of a second intermediate state y. Both intermediate states x and y are related by $y = S(x)$, where S is a known function from $\mathbf{F}_2^n$ into $\mathbf{F}_2^{n'}$, possibly parametrized by a part K_3 of the key. In practice, S can be a classical sbox, a superbox or some more complex function, as long as the attacker is able to precompute and store all possible transitions between the input bits obtained by the forward computation and the output bits obtained backwards (or sometimes, these transitions can even be computed on the fly). In particular, the involved intermediate states x and y usually correspond to partial internal states of the cipher, implying that their sizes n and n' are smaller than the blocksize.

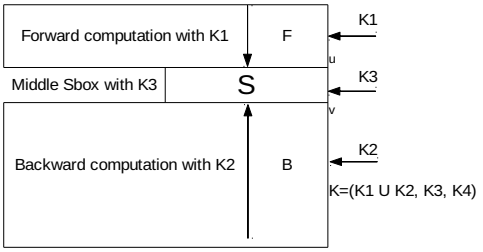


Fig. 1. Generic representation of Sieve-in-the-Middle

Then, the attacker is able to compute some pairs (u, v) in $\mathbf{F}_2^m \times \mathbf{F}_2^p$ and she wants to determine whether those pairs can be some valid parts of a pair $(x, S(x))$ for some $x \in \mathbf{F}_2^n$ (and for some K_3 if S depends on a part of the key). If it appears that no input $x \in \mathbf{F}_2^n$ can lead to a given (u, v), then the keys (K_1, K_2) from which (u, v) has been obtained do not form a valid candidate for the key. In such a case, the $(m + p)$ positions corresponding to (u, v) can be used as a sieve. The sieving probability is then the proportion of pairs (u, v) corresponding to valid parts of $(x, S(x))$. Obviously, in classical MITM attacks, u and v correspond to the same n-bit part of an intermediate state and $S = \mathrm{Id}_n$; the sieving probability is then equal to 2^{-n}. We now define precisely when a pair (I, J) of input and output positions can be used as a sieve.

Definition 1. *Let S be a function from $\mathbf{F}_2^n$ into $\mathbf{F}_2^{n'}$. Let $I \subset \{1, \ldots, n\}$ and $J \subset \{1, \ldots, n'\}$ be two subsets with respective sizes m and p. The sieving probability of (I, J), denoted by $\pi_{I,J}$, is the proportion of all elements in $\mathbf{F}_2^{m+p}$ which can be written as $(x_i, i \in I; S_j(x), j \in J)$ for some $x \in \mathbf{F}_2^n$. The pair (I, J) is called an (m, p)-sieve for S if $\pi_{I,J} < 1$.*

The smaller $\pi_{I,J}$, the better the sieving, because more candidates will be discarded. If S depends on a k_3-bit value key K_3, the definition similarly applies but S must be seen as a function with $(k_3 + n)$ inputs.

When a large number of inputs and outputs of S can be computed by the attacker, they can be used as a sieve, as shown in the following proposition.

Proposition 1. *Any pair (I, J) of sets of size (m, p) with $m + p > n$ is a sieve for S with sieving probability $\pi_{I,J} \leq 2^{n-(m+p)}$.*

Proof. For any given u, there exists exactly 2^{n-m} values of x such that $(x_i, i \in I) = u$. Thus, $(S_j(x), j \in J)$ can take at most 2^{n-m} different values, implying that $\pi_{I,J} \leq 2^{n-(m+p)}$.

However, smaller subsets I and J may provide a sieve even when $m + p \leq n$. This issue will be extensively discussed in Section 5. More generally, u and v may consist of some information bits of x and y, i.e., of some linear combinations of the bits of x and y. We then define two linear functions $L : x \in \mathbf{F}_2^n \mapsto u \in \mathbf{F}_2^m$ and $L' : y \in \mathbf{F}_2^n \mapsto v \in \mathbf{F}_2^p$. The corresponding sieving probability π is now the proportion of (u, v) such that there exists $x \in \mathbf{F}_2^n$ with $L(x) = u$ and $L'(S(x)) = v$. Then, π can be seen as the sieving probability of $I = \{1, \ldots, m\}$ and $J = \{1, \ldots, p\}$ for the function $L' \circ S \circ \widetilde{L}^{-1}$ where $\widetilde{L}$ is any linear permutation of $\mathbf{F}_2^n$ such that $(\widetilde{L}(x)_i, i \in I) = L(x)$.

2.2 Description of the Attack

We now precisely describe the improved MITM attack and provide its complexity. The secret key K is divided into four (possibly non-disjoint) parts, K_1 to K_4. K_1 (resp. K_2) is the part of the key used in the forward (resp. backward) computation, while K_3 is involved in the middle S function only (see Fig. 1). The key bits corresponding to K_4 are not involved in the MITM step. In the following, k_i denotes the length of the key part K_i, while k is the total key length. Moreover, $K_1 \cap K_2$ denotes the bits shared by K_1 and K_2, and κ corresponds to the size of this intersection.

We denote by I (resp. J) the set of input positions of S (resp. output positions) corresponding to u (resp. v). The fact that a pair (u, v) corresponds to a valid pair of inputs and outputs of S is characterized by a Boolean relation $\mathcal{R}$ with $(m + p)$ inputs defined by

$$\mathcal{R}(u, v) = 1 \text{ if and only if } \exists x \in \mathbf{F}_2^n : (x_i, i \in I) = u \text{ and } (S(x)_j, j \in J) = v \ .$$

The attack proceeds as follows.

for all 2^κ values of $K_1 \cap K_2$ **do**
  $\mathcal{L}_f \leftarrow \emptyset$ and $\mathcal{L}_b \leftarrow \emptyset$
  // Forward computation
  for all $2^{k_1-\kappa}$ values of the remaining bits of K_1 **do**
    compute $u = F_{K_1}(P)$ and add u to $\mathcal{L}_f$
  // Backward computation
  for all $2^{k_2-\kappa}$ values of the remaining bits of K_2 **do**
    compute $v = B_{K_2}(C)$ and add v to $\mathcal{L}_b$
  // Merging step
  Merge $\mathcal{L}_f$ and $\mathcal{L}_b$ w.r.t. Relation $\mathcal{R}$ and return the merged list $\mathcal{L}_{sol}$.
// Testing the remaining candidates
for all K with (K_1, K_2) in $\mathcal{L}_{sol}$ **do**
  if $E_K(P) = C$ **then**
    return K

Section 2.3 details some efficient algorithms for merging the two lists $\mathcal{L}_f$ and $\mathcal{L}_b$ (i.e. for recovering all the (u, v) which satisfy $\mathcal{R}(u, v) = 1$) with complexity lower than the product of their sizes.

With a Single Plaintext-Ciphertext Pair. Obviously, the whole secret key can be recovered only if the key length does not exceed the blocksize. Otherwise, 2^{k-b} possible keys will be returned in average where b is the blocksize. The time complexity of the attack is given by:

$$2^\kappa \left(2^{k_1-\kappa} c_F + 2^{k_2-\kappa} c_B + C_{\mathsf{merge}}\right) + \pi 2^k c_E \ ,$$

where π is the sieving probability of (I, J) as defined in Definition 1, c_E is the cost of one encryption, while c_F and c_B correspond to the costs of a partial encryption in the forward and backward directions. In most cases, $c_F \simeq c_B \simeq c_E/2$. C_{merge} is the time complexity of the merging step, and it depends on k_3. Its value is discussed in the following section. The average time complexity of the attack needs to be compared to $2^k c_E$ which is the cost of the exhaustive search. The memory complexity is mainly determined by the memory needed in the merging step. In some cases, it can be improved by storing only one among the two lists $\mathcal{L}_f$ and $\mathcal{L}_b$, when the auxiliary lists used in the merging step remain smaller.

With N Plaintext-Ciphertext Pairs. If N plaintext-ciphertext pairs are available to the attacker, then the average number of keys returned by the attack is 2^{k-Nb}, implying that the whole key will be recovered when $N \geq k/b$. The main modification in the attack concerns the last step where all key candidates in $\mathcal{L}_{sol}$ are tested: before performing an exhaustive search over $(K_1 \cap K_2)$ and K_4 for testing all keys with $(K_1, K_2) \in \mathcal{L}_{sol}$, an additional sieving step is performed in order to reduce the size of $\mathcal{L}_{sol}$. Once a new solution $(K_1, K_2) \in \mathcal{L}_{sol}$ has been found, $(N-1)$ additional pairs (u_i, v_i) generated from the other plaintext-ciphertext pairs are considered, and only the keys for which $\mathcal{R}(u_i, v_i) = 1$ are

kept in $\mathcal{L}_{sol}$ (note that, in some very particular situations, it might be more efficient to directly include in $\mathcal{L}_f$ and $\mathcal{L}_b$ the values u and v generated from several plaintext-ciphertext pairs, and then merge the lists). The average size of $\mathcal{L}_{sol}$ after this additional sieving step is then $\pi^N 2^{k_1+k_2-2\kappa}$. But this formula should be adapted to the case where S depends on a part of the secret key K_3: indeed the merging step determines a candidate for (K_1, K_2, K_3). Then, the sieving probability of the additional sieving step π' differs from π since the value of K_3 is now fixed. π' is then the sieving probability of (I, J) for S_{K_3} averaged over all K_3. Then, in the case of N plaintext-ciphertext pairs, the cost of the forward and backward computations are multiplied by N, while the cost of the testing part decreases:

$$2^\kappa \left(N2^{k_1-\kappa}c_F + N2^{k_2-\kappa}c_B + C_{\mathsf{merge}}\right) + \pi(\pi')^{N-1}2^k c_E \ .$$

2.3 Merging the Two Lists Efficiently

Very often, the middle function S can be decomposed into several smaller sboxes, and the merging step can be performed group-wise. The problem of merging two large lists with respect to a group-wise Boolean relation has been defined and addressed by Naya-Plasencia in [19, Section 2]. Here, we focus on three algorithms proposed in [19], namely instant matching, gradual matching and an improvement of the parallel matching due to [10]. We provide general and precise formulas for the average time and memory complexities of these three algorithms. Actually, in our case, the lists to be merged may be small. Then, the construction of some auxiliary tables, which had a negligible cost in [19] for large lists, must now be taken into account. It might even become the bottleneck of the algorithm. Thus, when the involved lists are small, it is harder to determine a priori which algorithm is the most efficient in a given case. Then, in each application, we need to check thoroughly which algorithm provides the best complexity. The optimal case may even sometimes correspond to the combination of two algorithms.

In the following, we consider two lists, $\mathcal{L}_A$ of size 2^{ℓ_A} and $\mathcal{L}_B$ of size 2^{ℓ_B}, whose roles are interchangeable. The elements of both lists can be decomposed into t groups: the i-th group of $a \in \mathcal{L}_A$ has size m_i, while the i-th group of $b \in \mathcal{L}_B$ has size p_i. The Boolean relation $\mathcal{R}$ can similarly be considered group-wise: $\mathcal{R}(a, b) = 1$ if and only $\mathcal{R}_i(a_i, b_i) = 1$ for all $1 \leq i \leq t$. The sieving probability π associated to $\mathcal{R}$ then corresponds to the product of the sieving probabilities π_i associated to each $\mathcal{R}_i$. Since each $\mathcal{R}_i$ corresponds to an sbox S_i with n_i-bit inputs, a table storing all (a_i, b_i) such that $\mathcal{R}_i(a_i, b_i) = 1$ can be built with time complexity 2^{n_i}, by computing all $(x_i, S_i(x_i)), x_i \in \mathbf{F}_2^{n_i}$. The corresponding memory complexity is proportional to $\pi_i 2^{m_i+p_i}$. This cost won't be included in the cost of the merging algorithm since, in the sieve-in-the-middle process, the tables will be built once for all and not 2^κ times. As we will see, in some situations, these tables can be built "on-the-fly" with much fewer operations.

A complete description of the three matching algorithms is provided in the full version [9]. It is worth noticing that the size of the list $\mathcal{L}_{sol}$ returned by the matching algorithm is not included in the memory complexity since each of its elements can be tested in the attack as soon as it has been found.

Instant Matching. Instant matching successively considers all elements $\mathcal{L}_B$: for each $b \in \mathcal{L}_B$, a list $\mathcal{L}_{aux}$ of all a such that $\mathcal{R}(a,b) = 1$ is built, and each element of $\mathcal{L}_{aux}$ is searched within $\mathcal{L}_A$. Its complexity is

$$\text{Time} = \pi 2^{\ell_B+m} + \pi 2^{\ell_A+\ell_B} \text{ and Memory} = 2^{\ell_A} + 2^{\ell_B} .$$

Gradual Matching. Gradual matching is a recursive procedure: all elements are decomposed into two parts, the first t' groups and the last $(t-t')$, with $t' < t$. For each possible value β of the first t' groups, the sublist $L_B(\beta)$ is built. It consists of all elements in $\mathcal{L}_B$ whose first t' groups take the value β. Now, for each α such that $\mathcal{R}_i(\alpha_i, \beta_i) = 1, 1 \leq i \leq t'$, $L_B(\beta)$ is merged with the sublist $L_A(\alpha)$ which consists of all elements in $\mathcal{L}_A$ whose first t' groups take the value α. Then, we need to merge two smaller lists, of respective sizes $2^{\ell_A - \sum_{i=1}^{t'} m_i}$ and $2^{\ell_B - \sum_{i=1}^{t'} p_i}$.

$$\text{Time} = \left(\prod_{i=1}^{t'} \pi_i\right) 2^{\sum_{i=1}^{t'} m_i + p_i} C_{\mathsf{merge}} \text{ and Memory} = 2^{\ell_A} + 2^{\ell_B} .$$

where C_{merge} is the cost of merging the two remaining sublists.

Parallel Matching without Memory. We give here the first general description of the memoryless version of parallel matching. This algorithm applies an idea from [10] to the parallel matching algorithm from [19]: instead of building a big auxiliary list as in the original parallel matching, we here build small ones which do not need any additional memory. In parallel matching, the elements in both lists are decomposed into three parts: the first t_1 groups, the next t_2 groups, and the remaining $(t - t_1 - t_2)$ groups. Both lists $\mathcal{L}_A$ and $\mathcal{L}_B$ are sorted in lexicographic order. Then, $\mathcal{L}_A$ can be seen as a collection of sublists $\mathcal{L}_A(\alpha)$, where $\mathcal{L}_A(\alpha)$ is composed of all elements in $\mathcal{L}_A$ whose first t groups equal α. Similarly, $\mathcal{L}_B$ is seen as a collection of $\mathcal{L}_B(\beta)$. The matching algorithm then proceeds as follows. For each possible value α for the first t groups, an auxiliary list $\mathcal{L}_{aux}$ is built, corresponding to the union of all $\mathcal{L}_B(\beta)$ where (α, β) satisfies the first t relations $\mathcal{R}_j$. The list $\mathcal{L}_{aux}$ is sorted by its next t_2 groups. Then, for each element in $\mathcal{L}_A(\alpha)$, we check if a match for its next t_2 groups exists in $\mathcal{L}_{aux}$. For each finding, the remaining $(t - t_1 - t_2)$ groups are tested and only the elements which satisfy the remaining $(t - t_1 - t_2)$ relations are returned. Details on the evaluation of the time and memory complexities are given in [9].

$$\text{Time} = \left(\prod_{i=1}^{t_1} \pi_i\right) 2^{\ell_B + \sum_{i=1}^{t_1} m_i} + \left(\prod_{i=t_1+1}^{t_1+t_2} \pi_i\right) 2^{\ell_A + \sum_{i=t_1+1}^{t_1+t_2} p_i} + \left(\prod_{i=1}^{t_1+t_2} \pi_i\right) 2^{\ell_A+\ell_B}$$

$$\text{Memory} = 2^{\ell_A} + 2^{\ell_B} + \left(\prod_{i=1}^{t_1} \pi_i\right) 2^{\ell_B} .$$

3 Combining Sieve-in-the-Middle and Bicliques

Sieve-in-the-middle, as a generic technique, can be combined with other improvements of MITM attacks, in particular with bicliques [4,18]. The general purpose of bicliques is to increase the number of rounds attacked by MITM techniques. Here, we briefly describe how bicliques can increase the number of rounds attacked by the previously described sieve-in-the-middle algorithm. This can be done at no computational cost, but requires a higher data complexity. In order to avoid this drawback, we then present an improvement of bicliques which applies when the key length exceeds the block size of the cipher.

3.1 Sieve-in-the-Middle and Classical Bicliques

The combination of both techniques is depicted on Figure 2: the bottom part is covered by bicliques, while the remaining part is covered by a sieve-in-the-middle algorithm. In the following, $H_{K_8} : X \mapsto C$ denotes the function corresponding to the bottom part of the cipher, and K_8 represents the key bits involved in this part. Then, K_8 is partitioned into three disjoint subsets, K_5, K_6 and K_7. The value taken by K_i with $5 \leq i \leq 7$ will be represented by an integer in $\{0, \ldots, 2^{k_i} - 1\}$. A biclique can be built if the active bits in the computation of $H_{K_8}(X)$ when K_6 varies and the active bits in the computation of $H^{-1}_{K_8}(C)$ when K_5 varies are two disjoint sets. In this case, an exhaustive search over K_7 is performed and a biclique is built for each value h of K_7 as follows. We start from a given ciphertext C^0 and a chosen key $K_8^0 = (0, 0, h)$ formed by the candidate for K_7 and the zero value for K_5 and K_6. We compute $X_h^0 = H^{-1}_{0,0,h}(C^0)$. Next, we compute backwards from C^0 the intermediate state $X_h^i = H^{-1}_{i,0,h}(C^0)$ for each possible value i for K_5. Similarly, we compute forwards from X_h^0 the ciphertext $C_h^j = H_{0,j,h}(X_h^0)$ for each possible value j of K_6. Since the two differential paths are independent, we deduce that $H_{i,j,h}(X_h^i) = C_h^j$ for all values (i, j) of (K_5, K_6).

Then, the sieve-in-the-middle algorithm can be applied for each K_7 and each value for $(K_1 \cap K_2)$. The list $\mathcal{L}_b$ of all output vectors v is computed backwards

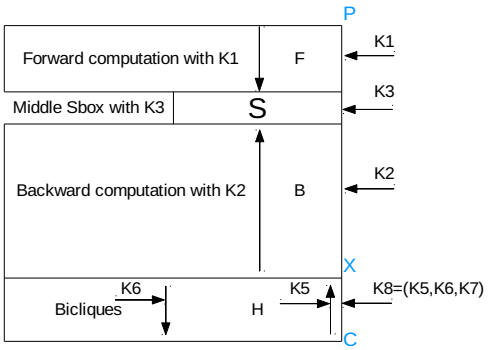


Fig. 2. Generic representation of Sieve-in-the-Middle and bicliques

from X_h^i for each value i of K_5 and each value of $K_2 \setminus (K_1 \cap K_2)$. The list $\mathcal{L}_f$ of all input vectors u is computed forwards from all plaintexts P_h^j corresponding to C_h^j for each value j of K_6 and each value of $K_1 \setminus (K_1 \cap K_2)$. We then merge those two lists of respective sizes $2^{|K_2 \cup K_5|}$ and $2^{|K_1 \cup K_6|}$.

As in classical MITM with bicliques, the decomposition of K_8 should be such that the bits of K_5 do not belong to K_1, the bits of K_6 do not belong to K_2 and the bits of K_7 should lie in $(K_1 \cap K_2)$. The best strategy here seems to choose (K_5, K_6) such that the bits of K_5 belong to $K_2 \setminus (K_1 \cap K_2)$, and the bits of K_6 belong to $K_1 \setminus (K_1 \cap K_2)$. In this case, we have to add to the time complexity of the attack the cost of the construction of the bicliques, i.e., $2^{k_7}(2^{k_5}+2^{k_6})c_H$ (very rarely the bottleneck), where c_H is the cost of the partial encryption or decryption corresponding to the rounds covered by the bicliques. The main change is that the data complexity has increased since the attack now requires the knowledge of all plaintext-ciphertext pairs (P_h^j, C_h^j) corresponding to all possible values (j, h) for (K_6, K_7). The data complexity then would correspond to $2^{k_6+k_7}$ pairs of plaintext-chosen ciphertexts, but it is usually smaller since the ciphertexts C_h^j only differ on a few positions.

3.2 Improved Bicliques for Some Scenarios

Now, we describe a generic idea for improving bicliques in certain scenarios and reducing the data complexity to a single plaintext-ciphertext pair. Our improvement usually applies when the total key size of the cipher is larger than the block size. This occurs for instance when whitening keys are used. A detailed and successful application is demonstrated on PRINCE in Section 4. The main idea of our improvement is to gather some parts of the partial exhaustive search over K_7 into different groups such that, within a group, all obtained ciphertexts C^j are equal to C^0.

We consider a biclique repartition of keys consistent with the sieve-in-the-middle part: we choose $K_5 \subset K_2 \setminus (K_1 \cap K_2)$ as previously, and some set $K_6' \subset K_1$ (this differs from the classical biclique construction where we had $K_6 \subset K_1 \setminus (K_1 \cap K_2)$). Let Δ_6^C be the positions of the bits of C which may be affected by K_6' when computing forward from X, and let Δ_6^X be the positions of the bits of X which may be affected by Δ_6^C and K_6' during the backward computation. In classical bicliques, the path generated in the backward direction by the different K_5 must be independent from the path generated in the forward direction by the different K_6'. Here, we also require this first path generated by K_5 to be independent from the backward path generated when the ciphertext bits in positions Δ_6^C vary. For instance, in the example depicted on Figure 3, H follows the Even-Mansour construction, i.e., it is composed of an unkeyed permutation H' and the addition of two whitening keys K_a and K_b. The positions of K_5 and K_6' are represented in red and blue respectively, and it can be checked that the corresponding paths are independent.

In this situation, an improved biclique without any additional data can be built if the size of Δ_6^X is smaller than k_6'. In our context, the algorithm has to be repeated for each value h for $K_7' = K_8 \setminus (K_5 \cup K_6')$, but the index h will

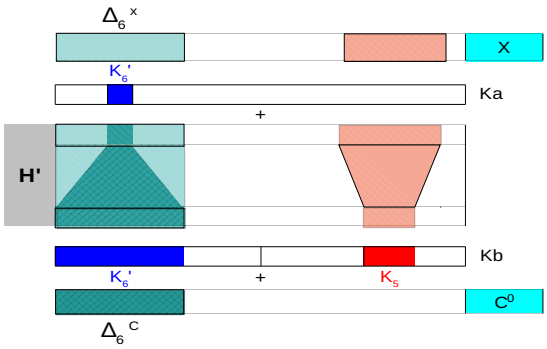
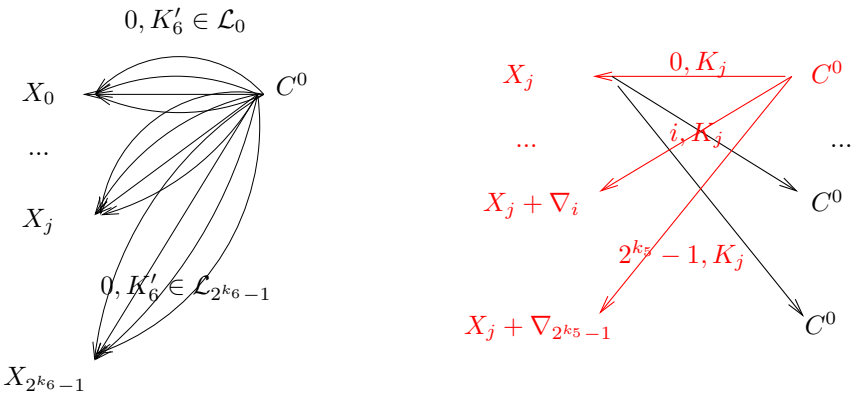


Fig. 3. Example of the improved biclique construction

be omitted in the description. First, we precompute the values obtained from a chosen C^0 when K'_6 takes all possible values. If the number of information bits in Δ_6^X is less than k'_6, all $2^{k'_6}$ transitions can be represented by several lists $\mathcal{L}_j$, each containing the different values of K'_6 which all map C^0 to the same value of the state X, X_j (see Figure 4(a)). For the sake of simplicity, we assume that all these lists have the same size 2^ℓ. In most cases, we have $\ell = k'_6 - |\Delta_6^X|$. For the example depicted on Figure 3, we assume that H' is such that the function obtained by restricting its inputs to the positions in Δ_6^X and its outputs to the positions in Δ_6^C is a permutation. Then, it clearly appears that the number of bits in Δ_6^X is equal to the number of bits of $K'_6 \cap K_b$, and thus strictly smaller than the number of bits of K'_6. More precisely, there are exactly 2^ℓ values of K'_6, with $\ell = |K'_6 \cap K_a|$, which provide the same value of $X = H'^{-1}(C^0 + K_b) + K_a$ when K'_6 varies and all other bits are fixed.



(a) Step 1

(b) Step 2: to be repeated for the $2^{k'_6-\ell}$ values of j

Fig. 4. Improved biclique construction

Now, for each of the $2^{k'_6-\ell}$ values of X_j, all transitions from C^0 to X_j through different values of $K'_6 \in \mathcal{L}_j$ can also be seen as the 2^ℓ biclique transitions from X_j to C^0 through some particular values of the key K'_6 (these transitions are represented in black on Figure 4(b)).

Now, the second step consists in building the bicliques in the other direction: from C^0 for each value of X_j. For each of the $2^{k'_6-\ell}$ values of j, we fix the value of K'_6 to a constant value K_j appearing in $\mathcal{L}_j$. This way, the part of X corresponding to Δ_6^X is the same for all the transitions of the bicliques, and this property holds even when K_5 is modified since both corresponding paths are independent. We then consider the 2^{k_5} possible values i for K_5 and compute the corresponding $X = X_j + \nabla_i$ (see Figure 4(b)). We then deduce the $2^{k_5+k'_6}$ transitions $H(X_j + \nabla_i)_{(i,K'_6)} = C^0$ for all $K'_6 \in \mathcal{L}_j$, from $(2^{k'_6} + 2^{k'_6-\ell+k_5})$ computations of the function. Indeed, the first term in the complexity corresponds to the precomputation phase (Step 1), and the second one to the number of lists $\mathcal{L}_j$, $2^{k'_6-\ell}$, multiplied by the cost for building the bicliques in the other direction. The main advantage of this construction is that it can be combined with the sieve-in-the-middle part as previously described, but it now requires a single plaintext-ciphertext pair, the one formed by (P^0, C^0).

Finally, we assume that the bits of K_5 belong to $K_2 \setminus (K_1 \cap K_2)$, the bits of K'_6 belong to K_1 and the bits of K'_7 are the bits from $(K_1 \cup K_2) \setminus (K_5 \cup K'_6)$, the time complexity of the attack is:

$$2^{k'_7}\left(2^{k'_6} + 2^{k'_6-\ell+k_5}\right) c_H + 2^{k_1} c_F + 2^{k_2} c_B + 2^{\kappa} C_{\mathsf{merge}} + \pi 2^k c_E$$

where C_{merge} is the cost of merging the lists of size $2^{k_1-\kappa}$ and $2^{k_2-\kappa}$ with respect to the sieving conditions.

A similar idea can also be used for choosing an appropriate K_5 which delays the propagation of the unknown bits during the forward computation. This will be shown in the case of Prince.

4 Application to PRINCE

PRINCE is a lightweight block cipher designed by Borghoff *et al.* [6]. Though being very recent, it has already waked the interest of many cryptanalysts [21,16,1]. The best known attacks so far on the proposed cipher, including the security analysis performed by the authors, reach 6 rounds. In particular, MITM with bicliques (without guessing the whole key) is said to reach at most 6 rounds (out of 12). In [16], a reduction of the security by one bit is presented, and in [1] an accelerated exhaustive search using bicliques is presented. Here, we describe how to build sieve-in-the-middle attacks on 8 rounds with data complexity 1 (or 2 if we want to the whole key instead of a set of candidates). In addition to the new sieve-in-the-middle technique, we use the improved method for constructing bicliques presented in Section 3.2.

4.1 Brief Description of PRINCE

PRINCE operates on 64-bit blocks and uses a 128-bit key composed of two 64-bit elements, K_a and K_b. Its structure is depicted on Figure 5. PRINCE is based on the so-called FX-construction: two whitening keys $W_{in} = (K_a + K_b)$ and $W_{out} = (K'_a + K_b)$ are xored respectively to the input and to the output of a 12-round core cipher parametrized by K_b only. The value of K'_a involved in the post-whitening key is derived from K_a by $K'_a = (K_a \ggg 1) \oplus (K_a \gg 63)$.

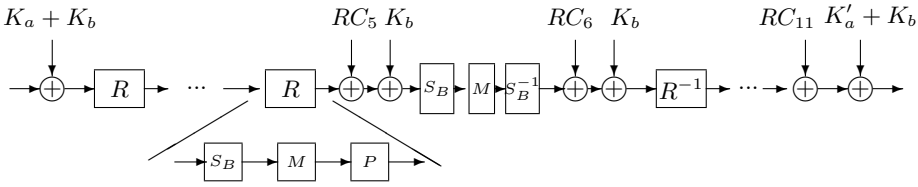


Fig. 5. Structure of PRINCE

The round function is composed of:

- a non-linear layer S_B corresponding to 16 parallel applications of a 4×4 sbox σ.
- a linear layer $P \circ M$, where M is the parallel application of 4 involutive mixcolumns operations on 16 bits each (defined either by $\hat{M}^{(0)}$ or by $\hat{M}^{(1)}$). This transformation is then followed by a permutation P of the 16 nibbles which is the same as the ShiftRows transformation used in the AES.
- the addition of a round constant RC_i and of the subkey K_b.

The first 5 rounds in PRINCE correspond to the previous round permutation R, while the last 5 rounds are defined by the inverse permutation R^{-1}. The two middle rounds correspond to the successive applications of S_B, M and S_B^{-1}.

4.2 Sieve-in-the-Middle and Improved Bicliques on 8 Rounds

Sieve-in-the-Middle on Six Rounds. We first describe the sieve-in-the-middle part of the attack, which covers Rounds 1 to 6 (see Figure 6). The internal state X after Round 6 is supposed to be known, as well as the plaintext. The sieving step is done with respect to a function S which covers Round 3 and the S_B level of Round 4. This middle function S can then be decomposed as four 16×16 superboxes: the colored nibbles in the middle of Figure 6 represent the nibbles belonging to the same superbox.

The 128 keybits in PRINCE are then decomposed as depicted on Figure 7:

- K_1, i.e. the keybits known in the forward direction, are represented in white and in blue in K_b and the first whitening key W_{in}. They correspond to all bits K_b and W_{in} except the 11 leftmost bits of the third 16-bit group in K_b.

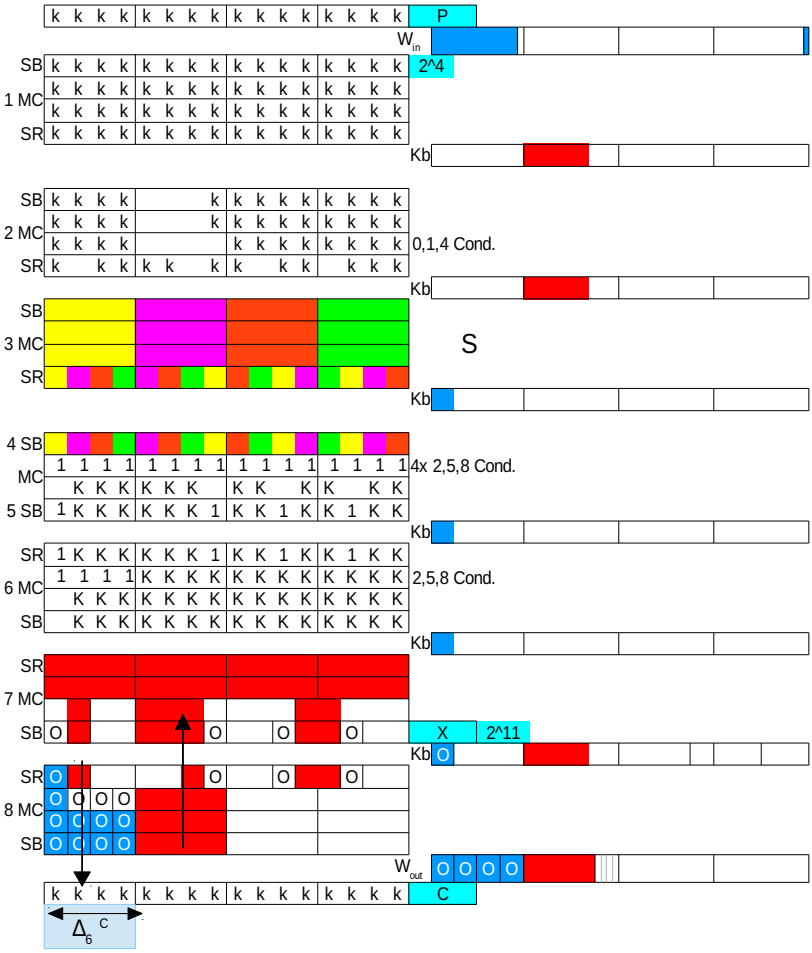


Fig. 6. Sieve-in-the-middle attack on 8 rounds of PRINCE with data complexity of 1

– K_2, i.e. the keybits known in the backward direction, are represented in white and in red in K_b and W_{in}. They correspond to all bits of K_b and W_{in} except the leftmost nibble of K_b and the 16 bits at positions 0 and 49 to 63 in W_{in}.

It follows that the intersection $(K_1 \cap K_2)$ consists of $\kappa = 97$ information bits of (K_a, K_b): the 49 white bits in K_b and the 48 white bits in W_{in}.

The algorithm is described on Figure 6, where each nibble which contains 'K' is known in the backward computation, each nibble which contains 'k' is known in the forward computation and '1' means that there is a known bit in the nibble. The right part of the figure represents the key. We will exploit the fact that, for each 16×16 mixcolumns operation, there exist 4 output bits (one

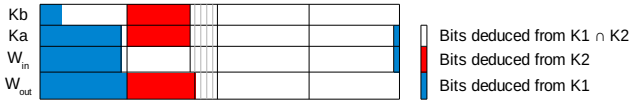


Fig. 7. Decomposition of the key in the attack on 8 rounds of PRINCE. $W_{in} = K_a \oplus K_b$ and $W_{out} = (K_a \ggg 1) \oplus (K_a \gg 63) \oplus K_b$.

per nibble), as well as 8 information bits of the output, which do not depend on a given input nibble. Each of these 8 information bits corresponds to the sum of two output bits (see [9, Sect. 6.2] for details). In the backward computation, from State X and K_2, we can compute 3 nibbles of each input of the mixcolumns operations at Round 5. Then, we deduce one bit in each nibble of the output of the middle function S, as well as 32 information bits which involve the outputs of two different superboxes. When considering $s < 4$ superboxes together, the number of information bits known is reduced to 8 if $s = 2$, and to 20 if $s = 3$.

In the forward computation, from the plaintext P and K_1, we compute three input nibbles of each superbox. From the mixcolumns operation in Round 2 whose input is partially known, we can also have 4 additional information bits on the input of the middle function S. When considering $s < 4$ superboxes together, the number of information bits known is reduced to 0 if $s = 2$ and to 1 if $s = 3$.

Then, we need to merge the two lists $\mathcal{L}_f$ and $\mathcal{L}_b$ of respective sizes 2^4 and 2^{11}. Since $m = 4 \times 12 + 4 = 52$ input bits and $p = 4 \times 4 + 32 = 48$ output bits are known, the total sieving probability π is at most $2^{64-(52+48)} = 2^{-36}$. In the following, the tables T_j providing all transitions for the four superboxes S_j are supposed to be known[1].

We are going to first apply the instant matching on the first two blocks (orange and green), i.e., instant matching as described in [9, Algo. 1] with parameters $n_1 = n_2 = 16$ and $m_1 = m_2 = 12$ and $p_1 + p_2 = 8 + 8 = 16$. The sieving probability of these two superboxes together is then $\pi_{1,2} = 2^{32-(24+16)} = 2^{-8}$. We consider $\mathcal{L}_A = \mathcal{L}_b$ and $\mathcal{L}_B = \mathcal{L}_f$. From the corresponding formula in Section 2.3, we get that the time complexity of this step is $2^{-8}2^{4+16} + 2^{-8}2^{15} \approx 2^{12}$. With this complexity we have found $2^{15}\pi_{1,2} = 2^7$ input-output pairs of S which are valid for the first two superboxes. We can now check whether each of these pairs is also valid for the two remaining superboxes. Now, the sieving probability for the remaining part is at most $2^{-36} \times 2^{+8} = 2^{-28}$ as the total sieving probability is at most 2^{-36}.

Therefore, at the end of the merging step, for each guess of the $\kappa = 113$ bits of $(K_1 \cap K_2)$, we have a probability of $2^{7-28} = 2^{-21}$ of finding a correct configuration for the 15 remaining bits of (K_1, K_2). This means that the testing step will consider $2^{113-21} = 2^{92}$ keys, and it will recover 2^{64} possible candidates

[1] The orange and green superboxes that involve common key bits only can be computed on the fly and will be used first for the instant matching. For each pair we obtain, the whole key is already known, so we can repeat the on-the-fly procedure.

for the whole key. If two plaintext-ciphertext pairs are available, the testing step will consider $2^{92-36} = 2^{56}$ keys instead of 2^{92}, leading to performing a test over 2^{56} candidates for recovering the correct key.

Improved Bicliques Part. Our attack combines the previous sieve-in-the-middle algorithm with bicliques built as described in Section 3.2, without increasing the data complexity. We define K'_6 as the five nibbles corresponding to the union of the leftmost nibble of K_b and the four leftmost nibbles of the whitening key $W_{out} = (K'_a + K_b)$. Then, Δ_6^C is represented on Figure 6 by the four 'O' symbols in the line before C. Also, Δ_6^X then corresponds to the 'O' symbols in X. Then, $|\Delta_6^C| = 16$ and $|\Delta_6^X| = 16$. The remaining 'O' show the path from Δ_6^C to Δ_6^X. All 2^{20} transitions obtained when K'_6 varies correspond, for each one of the 2^{16} possible values of j, to 2^4 biclique transitions from X_j to C. Then, K_5 is defined as the 11 leftmost bits of the third 16-bit group of K_b, implying that K_5 is equal to $K_2 \setminus (K_1 \cap K_2)$. The path generated in the backward direction, represented in red, is then independent from the blue path generated by K'_6, and also from the path with 'O' symbols from Δ_6^C to Δ_6^X.

The complete algorithm then consists in performing an exhaustive search over the $\kappa = 97$ common bits corresponding to the white bits of K_b and W_{in} in Figure 7. The previously described bicliques determine 2^{16} states X_j, and 2^4 transitions from each X_j to C. Then, for each X_j, we examine the corresponding 2^4 values of K'_6. For those K'_6, we compute forwards from the plaintext P the list of all 2^4 vectors u. It is worth noticing that even if the red bits of K_a and K_b are unknown in the forward direction, their sum is known (see Fig. 7). Similarly, the list $\mathcal{L}_b$ of all vectors v is computed backwards from the 2^{11} X^i and their associated value i for K_5. From the formula given in Section 3.2, we deduce that, for one plaintext-ciphertext pair, the time complexity is

$$\text{Time} = 2^{97}\left(2^{20}+2^{16+11}\right)c_H + 2^{117}c_F + 2^{113}c_B + 2^{97}\times 2^{12} + 2^{-36}\times 2^{128}c_E \simeq 2^{124}c_H \ .$$

We have then gained more than four bits over the exhaustive search ($2^{128}c_E$). The memory complexity is of 2^{20}, corresponding to the precomputed table in the construction of the improved bicliques, since the transition tables for the superboxes can be computed on the fly.

5 Sieving Probability and Related Properties of the Sbox

5.1 General Properties

In this section, we focus on the general problem of theoretically estimating the sieving property provided by two subsets $I \subset \{1, \ldots, n\}$ and $J \subset \{1, \ldots, n'\}$, with respective sizes m and p, for a given function S from $\mathbf{F}_2^n$ into $\mathbf{F}_2^{n'}$. In particular, we provide some results on the minimal value of $(m + p)$ for which a sieve exists. In the following, S_J denotes the function from $\mathbf{F}_2^n$ into $\mathbf{F}_2^p$ corresponding to the p coordinates of S defined by J. Also, for any affine subspace W, $S_{|W}$ denotes the restriction of S to W, i.e., the function defined on W by $S_{|W}(x) = S(x)$. Obviously, $S_{|W}$ can be identified with a function of $\dim W$ input variables.

For a given input set I, V denotes the linear subspace $V = \{x \in \mathbf{F}_2^n : x_i = 0, i \in I\}$. Then, the sieving probability of (I, J) can be expressed in terms of the sizes of all $\mathsf{Range}(S_J)_{|u+V}$ when u varies (see Prop 2 in [9]). Most notably, we deduce:

Corollary 1. *The sieving probability of (I, J) satisfies $\pi_{I,J} \geq 2^{-p}$, with equality if and only if S_J does not depend on its inputs at positions in $\{1, \ldots, n\} \setminus I$.*

Link with the branch number of S. We associate to S the (nonlinear) code $\mathcal{C}_S$ of length $(n + n')$ and of size 2^n defined by $\mathcal{C}_S = \{(x, S(x)), x \in \mathbf{F}_2^n\}$. The minimum distance of $\mathcal{C}_S$ is the lowest value of $wt(x + y) + wt(S(x) + S(y))$ for distinct x, y. It corresponds to the *branch number* of S. Obviously, when $m+p > n$, the sieving probability of any (I, J) of size (m, p) is at most $2^{n-(m+p)}$ (see Prop 1). Now, the following proposition shows that this upper bound is tight when $(m + p)$ exceeds some bound depending on the branch number of S.

Proposition 2. *Let m and p be two integers with $m + p \geq n$. Then, all (m, p)-sieves have probability $2^{n-(m+p)}$ if and only if $m + p > n + n' - d_{\min}$ where $d_{\min}$ is the branch number of S (i.e., the minimal distance of $\mathcal{C}_S$).*

For instance, the branch number of the 4×4 PRESENT sbox is equal to 3. It follows that any (m, p) sieve with $m + p \geq 6$ has probability $2^{n-(m+p)}$.

Lower Bound on the Minimal Value of $(m + p)$. Even if the code $\mathcal{C}_S$ is a nonlinear code, its dual distance can be defined as follows (if $\mathcal{C}_S$ is linear, this definition coincides with the minimum distance of the dual code $\mathcal{C}_S^{\perp}$).

Definition 2. *Let $\mathcal{C}$ be a code of length N and size M over $\mathbf{F}_q$ and $A = (A_0, \ldots, A_N)$ be its distance distribution, i.e., $A_i = \frac{1}{M}\#\{(x, y) \in \mathcal{C} \times \mathcal{C} : d_H(x, y) = i\}$.*

Let $A' = (A'_0, \ldots, A'_N)$ be the image of A under the MacWilliams transform, $A'(X, Y) = A(X + (q - 1)Y, X - Y)$ where $A(X, Y) = \sum_{i=0}^{N} A_i X^{N-i} Y^i$ and $A'(X, Y) = \sum_{i=0}^{N} A'_i X^{N-i} Y^i$. The dual distance *of $\mathcal{C}$ is the smallest nonzero index i such that $A'_i \neq 0$.*

The dual distance of $\mathcal{C}_S$ is a lower bound on the lowest $(m + p)$ for which an (m, p)-sieve exists.

Theorem 1. *Let $d^{\perp}$ be the dual distance of the code $\mathcal{C}_S$. Then, for any (m, p) such that $m + p < d^{\perp}$, there is no (m, p)-sieve for S. Moreover, there exists no (m, p)-sieve for S with $m+p \leq n$ if and only if $\mathcal{C}_S$ is an MDS code, which cannot occur if S is defined over $\mathbf{F}_2$.*

In some scenarios, S is defined over a larger alphabet, and I and J may be defined as two sets of byte (or nibble) positions. Then, the previous theorem proves that, if the corresponding code $\mathcal{C}_S$ is an MDS code, there is no (m, p)-sieve for $m + p \leq n$, and we deduce also from Proposition 2 that all (m, p)-sieve with $m + p > n$ have probability $2^{n-(m+p)}$.

5.2 Sieving Probability for Some Particular Values of (m, p)

$(m, 1)$-Sieves and Nonlinearity. When $p = 1$, a pair $(I, \{j\})$ of size $(m, 1)$ is a sieve if and only if S_j is constant on some coset $u + V$. Therefore, if $(I, \{j\})$ is a sieve, then S_j is $(n - m)$-normal, i.e. constant on an affine subspace of dimension $(n - m)$. In particular, it can be approximated by an affine function with a probability at least $\frac{1}{2}(1 + 2^{-m})$ [11]. It follows that, if S provides the best resistance to linear cryptanalysis for even n, then it has no sieve $(I, \{j\})$ with $|I| < \frac{n}{2} - 1$. As an example, the AES Sbox does not have any $(2, 1)$-sieve.

$(n - 1, p)$-Sieves. When $m = n - 1$, the sieving probability can be easily determined by the difference table of S.

Proposition 3. *Let $I = \{1, \ldots, n\} \setminus \{\ell\}$ and let $J \subset \{1, \ldots, n'\}$ with $|J| = p$. Then,*

$$\pi_{I,J} = 2^{-(p-1)} - 2^{-(p+n)} \sum_{\beta \in \mathbf{F}_2^{n'-p}} \delta(e_\ell, (0_J, \beta)) \ ,$$

where $\delta(a, b) = |\{x \in \mathbf{F}_2^n : S(x + a) + S(x) = b\}|$ is the element of index (a, b) in the difference table of S, and e_ℓ is the input vector with a 1 at position ℓ. Thus, $(I, \{j\})$ is a sieve except if S_j is linear in x_ℓ.

Since the branch number of the PRESENT sbox is 3, Prop. 2 implies that (m, p)-sieves with $m + p = 5$ exist for this sbox. Indeed, by considering its difference table, we get that all (I, J) of size $(3, 2)$ correspond to a sieving probability $\pi_{I,J} \in \{\frac{1}{2}, \frac{1}{2} - \frac{1}{32}, \frac{1}{2} - \frac{1}{16}\}$. For instance, the sieve used in the attack in [9], $I = \{0, 1, 2\}$ and $J = \{0, 1\}$ has probability $\frac{1}{2}$. We also derive from Prop. 3 the exact sieving probability involved in the attack on the DES in [9].

6 Conclusions

The main contributions of this paper are a generic improvement of MITM attacks, the sieve-in-the-middle technique, which allows to attack more rounds, and an improved biclique construction which avoids the need of additional data. These two methods have been applied to PRESENT, DES, AES and PRINCE. Moreover, some general results on the sieving probability of an sbox are given, which allow to theoretically estimate the complexity of the attack.

A future possible line of work is to investigate some possible combinations with other existing MITM improvements: with the guess of intermediate state bits [12], or with the all-subkeys approach [15]. A promising direction would be to try to make a first selection within each of the two lists before the merging step, by keeping only the input values (resp. output values) which have the lowest probability of corresponding to a valid transition. This introduces some non-detection probability, since some correct candidates would be discarded, but the sieving would be improved. Such an approach does not seem easy, but it would surely be a big step forward for further improving MITM attacks.

Acknowledgements. We thank Dmitry Khovratovich for his valuable comments, and all CryptoExperts members for their kindness and hospitality.

References

1. Abed, F., List, E., Lucks, S.: On the Security of the Core of PRINCE Against Biclique and Differential Cryptanalysis. Cryptology ePrint Archive, Report 2012/712 (2012), http://eprint.iacr.org/2012/712
2. Aoki, K., Sasaki, Y.: Preimage Attacks on One-Block MD4, 63-Step MD5 and More. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 103–119. Springer, Heidelberg (2009)
3. Aoki, K., Sasaki, Y.: Meet-in-the-Middle Preimage Attacks Against Reduced SHA-0 and SHA-1. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 70–89. Springer, Heidelberg (2009)
4. Bogdanov, A., Khovratovich, D., Rechberger, C.: Biclique Cryptanalysis of the Full AES. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 344–371. Springer, Heidelberg (2011)
5. Bogdanov, A., Rechberger, C.: A 3-Subset Meet-in-the-Middle Attack: Cryptanalysis of the Lightweight Block Cipher KTANTAN. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 229–240. Springer, Heidelberg (2011)
6. Borghoff, J., Canteaut, A., Güneysu, T., Kavun, E.B., Knezevic, M., Knudsen, L.R., Leander, G., Nikov, V., Paar, C., Rechberger, C., Rombouts, P., Thomsen, S.S., Yalçın, T.: PRINCE – A Low-Latency Block Cipher for Pervasive Computing Applications. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 208–225. Springer, Heidelberg (2012)
7. Bouillaguet, C., Derbez, P., Fouque, P.-A.: Automatic Search of Attacks on Round-Reduced AES and Applications. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 169–187. Springer, Heidelberg (2011)
8. Brumley, B.B., Hakala, R.M., Nyberg, K., Sovio, S.: Consecutive S-box Lookups: A Timing Attack on SNOW 3G. In: Soriano, M., Qing, S., López, J. (eds.) ICICS 2010. LNCS, vol. 6476, pp. 171–185. Springer, Heidelberg (2010)
9. Canteaut, A., Naya-Plasencia, M., Vayssière, B.: Sieve-in-the-Middle: Improved MITM Attacks (Full Version). Cryptology ePrint Archive, Report 2013/324 (2013), http://eprint.iacr.org/2013/324
10. Dinur, I., Dunkelman, O., Keller, N., Shamir, A.: Efficient Dissection of Composite Problems, with Applications to Cryptanalysis, Knapsacks, and Combinatorial Search Problems. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 719–740. Springer, Heidelberg (2012)
11. Dobbertin, H.: Construction of Bent Functions and Balanced Boolean Functions with High Nonlinearity. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 61–74. Springer, Heidelberg (1995)
12. Dunkelman, O., Sekar, G., Preneel, B.: Improved Meet-in-the-Middle Attacks on Reduced-Round DES. In: Srinathan, K., Rangan, C.P., Yung, M. (eds.) INDOCRYPT 2007. LNCS, vol. 4859, pp. 86–100. Springer, Heidelberg (2007)
13. Guo, J., Ling, S., Rechberger, C., Wang, H.: Advanced Meet-in-the-Middle Preimage Attacks: First Results on Full Tiger, and Improved Results on MD4 and SHA-2. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 56–75. Springer, Heidelberg (2010)

14. Isobe, T.: A Single-Key Attack on the Full GOST Block Cipher. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 290–305. Springer, Heidelberg (2011)
15. Isobe, T., Shibutani, K.: All Subkeys Recovery Attack on Block Ciphers: Extending Meet-in-the-Middle Approach. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 202–221. Springer, Heidelberg (2013)
16. Jean, J., Nikolic, I., Peyrin, T., Wang, L., Wu, S.: Security Analysis of PRINCE. In: FSE 2013. LNCS. Springer (to appear, 2013)
17. Khovratovich, D., Naya-Plasencia, M., Röck, A., Schläffer, M.: Cryptanalysis of *Luffa* v2 components. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 388–409. Springer, Heidelberg (2011)
18. Khovratovich, D., Rechberger, C., Savelieva, A.: Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 Family. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 244–263. Springer, Heidelberg (2012)
19. Naya-Plasencia, M.: How to Improve Rebound Attacks. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 188–205. Springer, Heidelberg (2011)
20. Sasaki, Y.: Meet-in-the-Middle Preimage Attacks on AES Hashing Modes and an Application to Whirlpool. IEICE Transactions 96-A(1), 121–130 (2013)
21. Soleimany, H., Blondeau, C., Yu, X., Wu, W., Nyberg, K., Zhang, H., Zhang, L., Wang, Y.: Reflection Cryptanalysis of PRINCE-like Ciphers. In: FSE 2013. LNCS. Springer (to appear, 2013)

Construction of Differential Characteristics in ARX Designs Application to Skein*

Gaëtan Leurent

UCL Crypto Group
Gaetan.Leurent@uclouvain.be

Abstract. In this paper, we study differential attacks against ARX schemes. We build upon the generalized characteristics of De Cannière and Rechberger and the multi-bit constraints of Leurent.

Our main result is an algorithm to build complex non-linear differential characteristics for ARX constructions, that we applied to reduced versions of the hash function Skein. We present several characteristics for use in various attack scenarios: on the one hand we show attacks with a relatively low complexity, in relatively strong settings; and on the other hand weaker distinguishers reaching more rounds. Our most notable results are practical free-start and semi-free-start collision attacks for 20 rounds and 12 rounds of Skein-256, respectively. Since the full version of Skein-256 has 72 rounds, this result confirms the large security margin of the design.

These results are some of the first examples of complex differential trails built for pure ARX designs. We believe this is an important work to assess the security those functions against differential cryptanalysis. Our tools are publicly available from the ARXtools webpage.

Keywords: Symmetric ciphers, Hash functions, ARX, Generalized characteristics, Differential attacks, Skein.

1 Introduction

ARX is a popular alternative to S-Box based designs for the design of symmetric key cryptographic primitives. ARX designs use only Additions ($a \boxplus b$), Rotations ($a \ggg i$), and Xors ($a \oplus b$). These operations are very simple and can be implemented efficiently in software or in hardware, but when mixed together, they interact in complex and non-linear ways. ARX designs have been quite popular recently; in particular, two of the SHA-3 finalists, BLAKE and Skein, follow this design strategy. This stategy as also been used for stream ciphers such as Salsa20 and ChaCha, and block ciphers, such as TEA, XTEA or HIGHT (RC5 uses additions and data-dependant rotations, but we only consider construction with fixed rotations). Recently, a dedicated short-input PRF, SipHash [1], has been

* Part of this work was done when the author was at the University of Luxembourg.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 241–258, 2013.

built following the ARX design. We note that Salsa20 is in the eStream portfolio, while SipHash is already deployed as the default hash table implementation of the Perl and Ruby languages. More generally, functions of the MD/SHA family are built using Additions, Rotations, Xors, but also bitwise Boolean functions, and logical shifts; they are sometimes also referred to as ARX.

The ARX design philosophy is opposed to S-Box based designs such as the AES. Analysis of S-Box based designs usually happen at the word-level; differential characteristics are relatively easy to build, but efficient attacks often need novel techniques, such as the rebound attack against hash functions [20]. For ARX designs, the analysis is done on a bit-level; finding good differential characteristics remains an important challenge. In particular, the seminal attacks on the MD/SHA-familiy by the team of X. Wang are based on differential characteristics built by hand [28,30,29], and a significant effort has been dedicated to building tools to construct automatically such characteristics [6,24,10,17,25,18,16]. This effort has been quite successful for functions of the MD/SHA family, and it has allowed new attacks based on specially designed characteristics: attacks against HMAC [11], the construction of a rogue MD5 CA certificate [26], and attacks against combiners [19].

However, this body of work is mainly focused on MD/SHA designs, as opposed to pure ARX designs such as Skein, BLAKE or Salsa20. In MD/SHA-like functions, the Boolean functions play an important role, and the possibility to absorb differences gives a lot of freedom for the construction of differential characteristics. In pure ARX designs, the addition is the only source of non-linearity (over $\mathbb{F}_2$), and the freedom in the carry expansions is much harder to use than the absorption property of Boolean functions.

To this effect, Leurent introduced multi-bit constraints [14] involving several consecutive bits of a variable (*i.e.* $x^{[i]}$ and $x^{[i-1]}$), instead of considering bits one by one. He describes reduced sets of 1.5-bit and 2.5-bit constraints, and explains how to propagate these constraints using S-systems and automata. This set of constraints is well suited to study ARX designs because it can extract a lot of information about the carry extensions in modular additions. A set of tools to propagate these constraints is given in [14], and the main result is a negative result (for the cryptanalyst) showing that several previous attacks are invalid.

1.1 Our Results

In this paper, we study the problem of constructing differential characteristics for ARX schemes. This work is heavily inspired by the framework of generalized characteristics from De Cannière and Rechberger [6], and the multi-bit constraints of [14]. As opposed to the results of [14], we give positive results for cryptanalysts.

We first recall how to describe a differential characteristic, and the main ideas for constraint propagation in Section 2. Then, we describe a differential characteristic search algorithm in Section 3 using a constraint propagation tool, and

we present our results on Skein in Section 4. Finally, we describe our technical improvements over the previous constraint propagation tools in the full version of this paper.

Construction of Differential Characteristics. We use a propagation tool to construct differential characteristics automatically. Using an efficient constraint propagation tool and some simple heuristics, we show that we can actually build complex non-linear characteristics. We obtain some of the first complex differential trails for ARX designs and we believe that this automated approach is an important step to assess the security of ARX designs against differential cryptanalysis.

Application to Skein. We apply this technique to reduced versions of the Skein hash function, where we build rebound-like characteristics by connecting two high-probability trails.

We compare our results with previous works in Table 1. Most previous works on Skein are either weak distinguishers (such as boomerang properties or free-tweak free-start partial-collisions), or attack with marginal improvement over brute-force (such as some biclique-based results). In this work, we present attacks in relatively strong settings (collisions and free-start collisions) with a relatively low complexity (several attacks are practical, and all our attack gain at least a factor 2^8).

Constraint Propagation. Finally, we describe an alternative way to perform the constraint propagation for multi-bit constraints. Our approach is significantly more efficient that the technique of [14], and uses the full set of 2^{32} constraints instead of a reduced set of 16 carefully chosen constraints. The reduced set is sufficient in most situations, but we show that the full set extracts some more information. This improvement was crucial to allow the characteristic search to work in practice.

In addition, our approach can also deal with larger systems that the previous technique with a reasonable complexity. In particular, we can deal with the 3-input modular sums, and 3-input Boolean functions used in functions of the MD/SHA family. We can also propagate 4 simultaneous trails in a boomerang configuration through an addition or an xor, with full 2-bit constraints.

1.2 Related Work

A recent result by Yu *et al.* achieves a similar result as our free-start free-tweak partial-collision on 32 rounds, and is also based on a complex non-linear trail for Skein-256. This work has been available on ePrint since April 2011 [31], but the characteristic given in that version of the paper was flawed [14]. This has motivated our work on building such characteristics automatically.

More recently, they managed to build a valid characteristic and their work has been presented at FSE [33]; this result was achieved simultaneously and independently from our work. Building such a trail by hand is impressive, but

Table 1. Comparison of attacks on reduced versions of Skein-256 (we omit attack on previous versions, and weak distinguishers). The full Skein-256 has 72 rounds. In order to compare various attack settings, we count the number of extra degrees of freedom used by the attack.

Extra Degrees of freedom		Rounds	Time	Generic	Ref, notes
Collision	0	4	2^{96}	2^{128}	[13], biclique
		8	2^{120}		
		9	2^{124}		
		12	$2^{126.5}$		
Free-start collision	8	$22^{\dagger}$	$2^{253.8\dagger}$	2^{256}	[15], biclique
		$37^{\dagger}$	$2^{255.7\dagger}$		
Related-tweak‡ partial q-multicol	10	20	$q \cdot 2^{97}$	$2^{\frac{q-1}{q+1} \cdot 130}$	[27], 126 bits
Free-tweak partial q-multicol	12	32	$q \cdot 2^{85}$	$2^{\frac{q-1}{q+1} \cdot 205}$	[33], 51 bits
Collision	0	12	$\approx 2^{100\star}$	2^{128}	4.4
Semi-free-start collision	4	12	$\approx 2^{40}$	2^{128}	4.4
Free-start collision	8	20	$\approx 2^{40}$	2^{128}	4.5
Free-start near-collision	8	24	$\approx 2^{40}$	$2^{88.4}$	4.5, 15 bits
Related-tweak‡ near-collision	10	24	$\approx 2^{40}$	$2^{117.3}$	4.6, 3 bits
Related-tweak‡ partial q-multicol	10	32	$\approx q \cdot 2^{119\star}$	$2^{\frac{q-1}{q+1} \cdot 205}$	4.6, 51 bits
Free-tweak partial q-multicol	12	32	$q \cdot 2^{105}$	$2^{\frac{q-1}{q+1} \cdot 205}$	4.6, 51 bits
Block cipher attacks					
Key recovery (Threefish-512)		32	2^{181}	2^{512}	[32], Boomerang
		33	2^{305}		
		34	2^{424}		

† Attacks on Skein-512. For Skein-256, fewer round will be attacked, with a complexity slightly below 2^{128}.
‡ Using freedom degrees in the tweak *difference*, but the tweak *value* can be arbitrary.
* Using heuristic assumptions about the search for a large number of characteristics.

this kind of result it is very challenging to replicate or to apply to another primitive. We hope that our automatic approach will be easier to adapt to new settings.

2 Analysis of Differential Characteristics

The first step for working with differential characteristics (or trails) is to choose a way to represent a characteristic, and to evaluate its probability. The main idea of differential cryptanalysis is to consider the computation of the function for a pair of inputs X, X', and to specify the difference between x and x' for every internal state variable x. The difference can be the xor difference, the modular difference, or more generally, use any group operation. However, this approach is not efficient for ARX design, because both the modular difference and the xor

difference play an important role. Several works have proposed better way to represent a differential characteristic for ARX designs.

Signed Bitwise Difference. The groundbreaking results of Wang *et al.* [28,30,29] are based on a bitwise signed difference. For each bit of the state, they specify whether the bit is inactive ($x = x'$), active with a positive sign ($x = 0, x' = 1$), or active with a negative sign ($x = 1, x' = 0$). This information express both the xor difference and the modular difference.

Generalized Characteristics. This was later generalized by De Cannière and Rechberger [6]: for each bit of the state, they look at all possible values of the pair (x, x'), and they specify which values are allowed. The constraints `-`, `u` and `n` correspond to the bitwise signed difference of Wang. De Cannière and Rechberger also describe an algorithm to build differential characteristics using this set of constraints.

Multi-bit Constraints. Recently, Leurent studied differential characteristics for ARX designs, and introduced multi-bit constraints [14]. These constraints are applied to the values of consecutive bits of a state variable (*e.g.* $x^{[i]}$ and $x^{[i-1]}$) instead of being purely bitwise. Multi-bit constraints are quite efficient to study ARX designs because they can capture the behaviour of carries in the modular addition. Two set of constraints are introduced in [14]:

- a set of 16 constraints involving $(x^{[i]}, x'^{[i]}, x^{[i-1]})$ called 1.5-bit constraints;
- a set of 16 constraints involving $(x^{[i]}, x'^{[i]}, x^{[i-1]}, x'^{[i-1]}, x^{[i-2]})$ called 2.5-bit constraints.

The full sets of 2^8 1.5-bit constraints and 2^{32} 2.5-bit constraint are not used because the propagation method of [14] becomes impractical with such large sets.

2.1 Constraint Propagation and Probability Computation

In [14], the constraints are studied using the theory of S-functions introduced in [22]. We use the following definitions:

T-function. A T-function on n-bit words with k inputs and l outputs is a function from $(\{0,1\}^n)^k$ to $(\{0,1\}^n)^l$ with the following property:

> *For all t, the t least significant bits of the outputs can be computed from the t least significant bits of the inputs.*

S-function. An S-function on n-bit words is a function from $(\{0,1\}^n)^k$ to $(\{0,1\}^n)^l$, for which we can define a small set of *states* $\mathcal{S}$, and an initial state $S[-1] \in \mathcal{S}$ with the following property:

> *For all t, bit t of the outputs and the state $S[t] \in \mathcal{S}$ can be computed from bit t of the inputs, and the state $S[t-1]$.*

For instance, the modular addition is an S-function, with a 1-bit state corresponding to the carry. An S-function can also include bitwise functions, shifts to the left by a fixed number of bits, or multiplications by constants. A system of equation that can be written as a S-function is called an S-system.

2.2 Differential Characteristics

In order to describe a differential characteristics with this framework, we specify a difference for each internal variable of a cipher, and we consider the operations that connect the variables. For a series a constraints Δ, we write $\delta x = \Delta$ to denote that the pair (x, x') follows the difference pattern Δ. For instance, $\delta x =$ `x--0` is equivalent to $x \oplus x' =$ `1000` and $x^{[0]} = 0$.

For each operation $\odot$, we can write a system:

$$\delta x = \Delta_x \qquad \delta y = \Delta_y \qquad \delta z = \Delta_z \qquad z = x \odot y \qquad z' = x' \odot y', \tag{1}$$

where x, y, z, x', y', z' are unknowns, and $\Delta_x, \Delta_y, \Delta_z$ are parameters. In an ARX design, all the operations except the rotations are S-function, and the difference operation δ can be written with bitwise operations and left-shifts; therefore system (1) is an S-system. Using tools to analyze this S-system, we can verify if the specified input and output patterns for each operation are compatible. We deal with the rotations $y = x \ggg i$ by just rotating the constraint pattern: if $\delta x = \Delta_x$ then we use $\delta y = \Delta_x \ggg i$.

We can also find new constraints that must be satisfied for any solution to the system. This allows to propagate constraints between the inputs and outputs of the operation $\odot$. When we consider a characteristic for a cipher, this process will be iterated for each operation, until no new constraints are found.

Moreover, we can compute the probability to reach the specified output pattern by counting the number of solutions. Assuming that the probabilities of each operations are independent, we can compute the probability of the full characteristic by multiplying the probabilities of each operations.

2.3 Tools for S-Systems

In [14], a set of constraints is represented by an S-system, and an automaton is built to compute the probability of each operation. To perform constraints propagation, each constraint is split into two disjoint subsets; if one of the subsets results in an incompatible system, the constraint can be restricted to the other subset without reducing the number of solutions.

This approach allows to achieve a good efficiency when the automaton is fully determinized: one can test whether a system is compatible with only n table accesses. However, the table becomes impractically large if the set of constraints is too large, or if the operation is too complex. In [14], the automaton is fully determinized for 1.5-bit constraints, but could not be determinized for 2.5-bit constraints; this results in a quite inefficient propagation algorithm for 2.5-bit constraints.

In this work, we explore a different option using non-deterministic automata. This allows to deal with large set of constraints and more complex operations.

We need to perform many operations to verify whether a system is compatible, but the automata are very sparse and can be represented by tables small enough to fit in the cache (the tables of [14] take hundreds of megabytes for an addition); this gives better results in practice. In addition, we show special properties of the automata allowing an efficient propagation algorithm without splitting the constraints into subsets. Due to space constraints, the technical details of our new approach are given in the full version of this paper.

2.4 Comparison

We show a comparison with previous methods in Table 2. We use the same settings as [14]:

1. A reduced Skein with two rounds and 4 words of 4 bits each; In this setting the full 2.5-bit constraints offer a little advantage over the reduced set of 2.5-bit constraints.
2. A reduced Skein with three rounds and 4 words of 6 bits each. We only use sparse differences (less than 4 active bits in the input and output), because the full space is too large to be exhausted in practice. In this setting, the full 2.5-bit constraints give a significant improvement over the reduced set of 2.5-bit constraints.

These experiments show that using the full set of 2.5-bit constraints gives better results than using the reduced set of [14]. We also give timing informations[1]: our new approach for constraint propagation is one order of magnitude faster that the previous method with a reduced set of 2.5-bit constraints, and somewhat slower than the previous method with 1.5-bit constraints.

Table 2. Experiments with toy versions of Skein. We give the number of input/output differences accepted by each technique, and the ratio of false positive.

	2 rounds / 4 bits		3 rounds / 6 bits (sparse*)		
Method	Accepted	F pos.	Accepted	F pos.	Time†
Exhaustive search	$2^{25.1}$ (35960536)	–	$2^{18.7}$ (427667)	–	
2.5-bit full set	$2^{25.3}$ (40597936)	0.13	$2^{19.2}$ (619492)	0.4	2.5 ms
2.5-bit reduced set [14]	$2^{25.3}$ (40820032)	0.14	$2^{19.5}$ (746742)	0.7	50 ms
1.5-bit reduced set [14]	$2^{25.3}$ (40820032)	0.14	$2^{20.4}$ (1372774)	2.2	0.5 ms
1-bit constraints [6]	$2^{25.4}$ (43564288)	0.21	$2^{20.7}$ (1762857)	3.1	0.5 ms
Check adds independently	$2^{25.8}$ (56484732)	0.57			

* Weight 4 differences. The number of input/output differences is $\left(\sum_{i=0}^{4} \binom{24}{i}\right)^2 \approx 2^{26.7}$

† Average time to verify one input/output difference (over the false positives of the 1.5-bit reduced set).

[1] The comparison is done with similar implementations.

3 Automatic Construction of Differential Characteristics

In order to mount a differential attack for a hash function or a block cipher, an important task is to build a differential characteristic. For the analysis of ARX primitives (and MD/SHA-like designs), the characteristic is usually designed at the bit level. This turns out to be a very challenging task because of the complex interactions between the operations, and the large number of state elements to consider.

This problem has been heavily studied for attacks on the MD/SHA family of hash functions: a series of attacks by X. Wang and her team are based on differential characteristics built by hand [28,30,29,33], while later works gave algorithms to build such characteristics automatically [6,24,10,17,25]. Unfortunately, most of those tools are not publicly available.

In this section, we show that the multi-bit constraints can be used to design a successful algorithm for this task on pure ARX designs. Our algorithm is heavily inspired by the pioneer work of De Cannière and Rechberger [6], and the more detailed explanation given in [23] and [21].

3.1 Types of Trails

Differential trails can be classified in two categories: iterative and non-iterative. An iterative characteristic exploits the round-based nature of many cryptographic constructions: if a trail can be built over a few rounds with the same input and output difference Δ, then this characteristic can be repeated to reach a larger number of rounds. In practice very few iterative characteristics have been found for ARX constructions, because many designs use different rotation amounts or Boolean functions over the rounds, or a non-iterative key-schedule. Notable exceptions include the attacks of den Boer and Bosselaers against MD5 [7], and the recent work of Dunkelman and Khovratovich on BLAKE [8]. In this work, we focus on non-iterative trails.

The main way to build non-iterated trails is to connect two simple and high-probability trails using a complex and low-probability section in between. The choice of the high-probability trails will depend on the attack setting, and should be done by the cryptanalyst using specific properties of the design, while the complex section will be build automatically by an algorithm (or by hand). When the characteristic is used in a hash-function attack, the cost of the low-probability section can usually be avoided.

For instance, the characteristics used for the attacks on SHA-1 use a linear section built using local collisions [4,29], and a non-linear section to connect a given input difference to the linear characteristic. This general idea is also the core of the rebound attack [20]: it combines two high-probability trails using a low-probability transition through an S-box layer.

In our applications, we will use a rebound-like approach to connect two high-probability trails with a complex low-probability section. Using rebound-like differential trails for ARX designs has also been proposed in [33].

3.2 Algorithm

Our algorithm takes as input a characteristic representing two high-probability trails $\Delta_1 \to \Delta_2$ and $\Delta_3 \to \Delta_4$. The middle section is initially unconstrained, *i.e.* filled with `?`. The main part of the algorithm is a search phase which tries to fill the middle part with a valid characteristic. We follow the general idea of the algorithm of De Cannière and Rechberger, by repeating the following operations:

Propagation: deduce more information from the current characteristic by running the propagation algorithm on each operation.

Guessing: select an unconstrained state bit (*i.e.* a `?` constraint), and reduce the set of allowed values (*e.g.* to a `-` or `x` constraint).

When a contradiction is found, we go back to the last guess, and make the opposite choice, leading to a backtracking algorithm. However, we abort after some number of trials and restart from scratch because mistakes in the early guesses would never be corrected.

Our algorithm is built from the idea that the constraint propagation is relatively efficient to check if a transition $\Delta \to \Delta'$ is possible. Therefore to connect the differences Δ_2 and Δ_3 from the high-probability trails, we essentially guess the middle difference Δ' and we check whether the transitions $\Delta_2 \to \Delta'$ and $\Delta' \to \Delta_3$ are possible.

This leads to the following difference with the algorithm of De Cannière and Rechberger:

- We only use signed differences, *i.e.* we use the constraints `-`, `u`, and `n`.
- We specify in advance which words of the state will be restricted in the guessing phase, using state words in the middle of the unspecified section.
- We guess from the low bits to the high bits, and we can compare incomplete characteristics by counting how many bits have been guessed before aborting the search.
- Every time the backtracking process is aborted, we remember which guess was best and the random guesses of the next run are strongly biased toward this choice.

Thanks to this approach, we can use the best path of the previous run as an input for the search algorithm, and explore solutions with few differences in the guesses. Finally, we use a simulated annealing algorithm in order to find better characteristics.

3.3 Finding Pairs

The hardest part of our attacks in to build the differential trails. Finding conforming pairs for the middle section is relatively easy using the propagation algorithm: one just has to make random choices for the unconstrained bits in the middle and run the propagation algorithm after each choice. In practice the paths we found leave very few choices to make, and most of them lead to valid pairs. The degrees of freedom in the key can then be used to build many different

pairs. This can be compared to the rebound attack on AES-like designs [20]: in this attack the trails are easy to build, and finding pairs for the inbound phase has a small amortized cost.

4 Application to Skein-256

In this section, we apply our algorithm to build characteristics for several attack scenarios on Skein-256.

4.1 Short Description of Threefish and Skein

The compression function of Skein is based on the block cipher Threefish. In this paper we only study Threefish-256, which uses a 256-bit key (as 4 64-bit values), a 128-bit tweak (as 2 64-bit values), and a 256-bit state (as 4 64-bit values). The full version of Skein has 72 rounds. We denote the ith word of the state after r rounds as $e_{r,i}$. There is a key addition layer every 4 rounds:

$$e_{r,i} = \begin{cases} v_{r,i} + k_{r/4,i} & \text{if } r \bmod 4 = 0 \\ v_{r,i} & \text{otherwise} \end{cases}$$

where $k_{r/4,i}$ is the ith word of the round key at round $r/4$. The state $v_{r+1,i}$ (for $i = 0, 1, .., n_w$) after round $r+1$ is obtained from $e_{r,i}$ by applying a $\texttt{MIX}$ transformation and a permutation of 4 words as following:

$$\begin{array}{lll} (f_{r,2j}, f_{r,2j+1}) & := \texttt{MIX}_{r,j}(e_{r,2j}, e_{r,2j+1}) & \text{for } j = 0, 1, .., n_w/2 \\ v_{r+1,i} & := f_{r,\sigma(i)} & \text{for } i = 0, 1, .., n_w \end{array}$$

where σ is the permutation (0 3 2 1) (specified in [9]) and $(c, d) = \texttt{MIX}_{r,j}(a, b)$ is defined as:

$$\begin{aligned} c &= a \boxplus b \\ d &= (b \lll R_{r \bmod 8,j}) \oplus c \end{aligned}$$

The rotations $R_{r \bmod 8,j}$ are specified in [9]. The key scheduling algorithm of Threefish produces the round keys from a tweak (t_0, t_1) and a key as following:

$$\begin{aligned} k_{l,0} &= k_{(l \quad) \bmod 5} & k_{l,1} &= k_{(l+1) \bmod 5} + t_{l \bmod 3} \\ k_{l,2} &= k_{(l+2) \bmod 5} + t_{(l+1) \bmod 3} & k_{l,3} &= k_{(l+3) \bmod 5} + l, \end{aligned}$$

where $k_4 = C_{240} \oplus \bigoplus_{i=0}^{4} k_i$ with C_{240} a constant specified in [9], and $t_2 = t_0 \oplus t_1$. The compression function F for Skein is given as $F(M, H, T) = E_{H,T}(M) \oplus M$, where H is the chaining value, M is the message, and T is a block counter. This follows the Matyas-Meyer-Oseas construction for the compression function, and the Haifa construction for the iteration.

In this work, we only consider attacks on multiples of four rounds, because the structure of Skein is built with 4-round blocks with key additions in between.

We describe attacks in three different settings in Sections 4.4, 4.5, and 4.6. The attacks are based on different kinds of trails shown in Figures 2, 3, and 4. Due to space constraints, we do not include differential characteristics, but they are given in the full version of this paper. All the characteristics have been verified by building a conforming pair, and we give example of colliding pairs in Appendix A.

4.2 Building Characteristics

To describe a differential characteristic for Skein with our framework, we write constraints for each $e_{r,i}$ value, and for the $v_{r,i}$ values before a key addition (*i.e.* when $r \bmod 4 = 0$). For each round, we have 4 equations and 2 rotations, corresponding to two `MIX` functions. We also write the full key schedule as a system of equations.

We note that the variables $e_{r,2j}$ with $r \bmod 4 = 0$ are only involved in modular additions: $f_{r,2j} = e_{r,2j} \boxplus e_{r,2j+1}$ and $e_{r,2j} = v_{r,2j} \boxplus k_{r/4,2j}$. Therefore, we could remove these variables, and write $f_{r,2j} = v_{r,2j} \boxplus k_{r/4,2j} \boxplus e_{r,2j+1}$ using a three-input modular addition. In practice, the propagation algorithm for three-input modular addition takes significantly longer, so we keep the variables, but we try to avoid constraining them since the multi-bit constraints can propagate the modular difference.

Choosing the High-Probability Characteristics. In attack setting with differences in the key, we build the high-probability trails starting from a non-active state, with a low-weight key difference. When we go through the key addition, a difference is introduced in the state, and we propagate the difference by linearizing the function. If we have no difference in the key, we start with a single active bit in the state and we propagate the difference for a few rounds by linearizing the function. Most of our trails use the most significant bit as the active bit in order to increase their probabilities.

4.3 General Results

For the algorithm to work successfully, we need to find a delicate balance in the initial characteristic. If the unconstrained section is too short, there will not be enough degrees of freedom to connect the high-probability parts. On the other hand, if the unconstrained section is too long, the propagation algorithm will not filter bad characteristics efficiently.

In practice, we can only build characteristics when we have a key addition layer in the unconstrained part of the characteristic. This way, the algorithm can use degrees of freedom from the key to connect the initial characteristics. In general it seems hard to find enough degrees of freedom to build a valid trail without using degrees of freedom from the key: for a random function f and arbitrary differences Δ_2 and Δ_3, we expect on average a single pair satisfying $f(x + \Delta_2) = f(x) + \Delta_3$. We can consider the intermediate differences for one such pair as a differential characteristics but a differential characteristic with a single valid pair is not very useful for a differential attack.

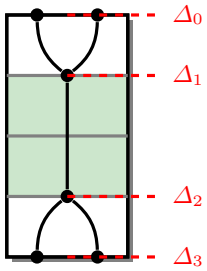


Fig. 1. Previous trails: rel-key, rel-tweak

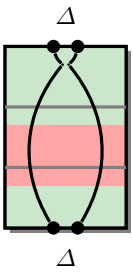


Fig. 2. Collision trails: fixed key

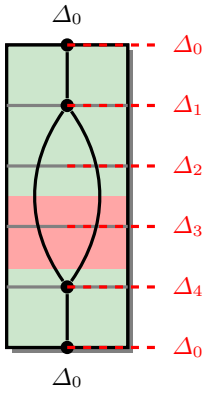


Fig. 3. Collision trails: related-key

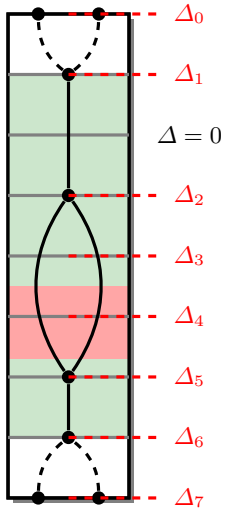


Fig. 4. (Near-) Collision trails: rel-key, rel-tweak

In order to let the algorithm use the degrees of freedom in the key efficiently, we use the registers before and after a key addition as guessing points: $v_{r,0}, v_{r,1}, v_{r,2}, v_{r,3}, e_{r,1}, e_{r,3}$ with $r \bmod 4 = 0$ (as discussed above we do not constrain $e_{r,0}$ and $e_{r,2}$).

We find that the characteristics built by the algorithm are rather dense, and use many degrees of freedom in the state, and many degrees of freedom in the key. This is not a problem for attacks on the compression function, but the characteristics are harder to use in attacks against the full hash function, where fewer degrees of freedom are available to the attacker. We note that this problem is less acute for attack against functions of the MD/SHA family, where the message block is much larger than the state.

On the other hand, the trail built by hand by Yu *et al.* [33] is somewhat sparser, and leaves more degrees of freedom for the key and the middle state.

4.4 Collision Attacks

We first study attacks with no difference in the chaining value so that they can be applied to the full hash function. Since Skein uses the MMO mode, the chaining value of the hash function is the key to the block cipher. We try to build characteristics for a collision attack, therefore we use the same difference in the initial state and in the final state so that they can cancel out in the feed-forward[2]. We start with a low-weight difference in one of the first rounds

[2] We could build characteristics for 20 rounds if we consider near-collisions, but this would not work on the full hash function because of the finalization step.

and we propagate by linearization through rounds 0–4 and backward through round 11.

We give an example of a colliding pair for the compression function of Skein-256 reduced to 12 rounds in Table 3.

Full Collision Attack. To build a collision attack on the full hash function, we have to deal with the fact that the characteristic is only valid for a small fraction of the keys, *i.e.* a small fraction of the chaining values. We use a large number of characteristics, and a large number of random chaining values, in a meet-in-the-middle fashion.

More precisely, the characteristics given by the algorithm have many constraints of the key, which define a set of valid keys, and the number of conforming pairs estimated by looking at the probability of each step is even lower. We assume that each solution will correspond to a different key, and the number of solutions of the characteristics gives the number of key (*i.e.* chaining values) for which we can actually build a collision. Our experiments indicate that we can expect to build characteristics with more than 2^{106} solutions for a cost of 2^{50}. If we extrapolate this experimental result, we expect that it is possible to build many such characteristics. Let's assume that we can build N characteristics for a cost of $N \times 2^{50}$; where each characteristic has 2^{106} solutions out of 2^{200} valid keys. In a second phase, we will hash M random message blocks and test if they can give a collision using one of the characteristics. Out of the M chaining values generated, we expect that $M \times N \times 2^{200-256}$ will be valid for one characteristic, and $M \times N \times 2^{106-256}$ values will actually lead to a collision after verification. An important step of the attack will be to find for which characteristic a given chaining value can be valid, but this can be done efficiently using a hash table indexed by the bits of the chaining value which are imposed by the characteristics.

The optimal complexity is achieved with $N = 2^{50}$ and $M = 2^{100}$. With these parameters we only have to verify 2^{94} valid chaining values, so the verification step is negligible. This gives a collision attack on 12-round Skein-256 with a time complexity of 2^{100}, using memory to store 2^{50} characteristics[3].

The assumption that we can build so many good characteristics is a strong assumption, and it is hard to verify. However, we believe that this estimation is a safe upper bound, and that better characteristics would be found by running the search algorithm for longer times. In our experiments, we tested a few different high probability trails as input to the algorithms, and we spend an effort equivalent to about 2^{50} hash computations on our best candidate.We ran our algorithm 128 times with different initialization of the PRNG, and we report the best paths found by each run after 120 CPU hours in Figure 5. In addition, we ran a few parallel experiments for 120 hours with 32 cores. All these experiments generated more than 400.000 different characteristics; the best characteristic al-

[3] To store a characteristic, we just need to store masks defining the valid keys, and one state in the middle (if is not necessary to store all the intermediate constraints). Then, we can test a chaining value candidate by just computing all the intermediate states and checking if we reach a collision. This would take about 4×256 bits.

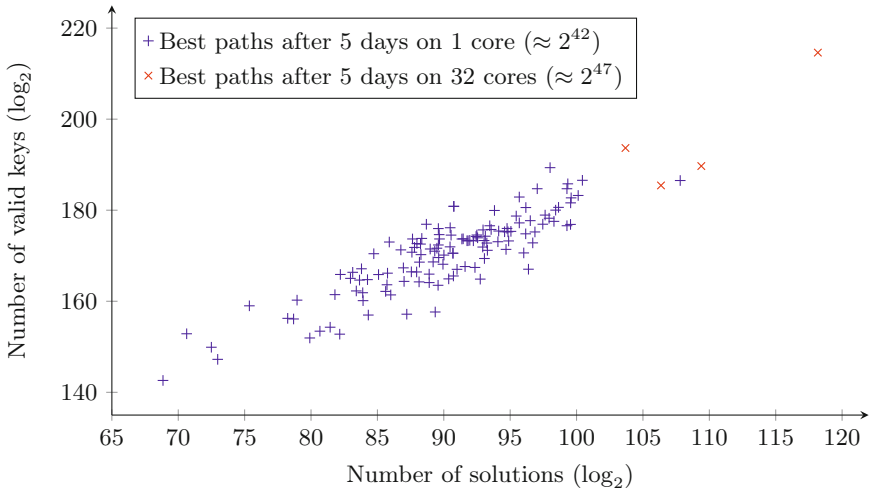


Fig. 5. Best characteristic found after each run. The experiments were run on Intel Xeon L5420 CPUs.

low 2^{118} solutions, and 30.000 of these allow more than 2^{106} solutions (only the best characteristic of each run is shown in Figure 5). We note that in order to build a large number of characteristics, we would also use several different starting points for the linear part.

4.5 Free-Start Collision Attack

For a collision attack on the compression function, *i.e.* a free-start attack on the hash function, we can use a difference in the chaining value (*i.e.* the key). We note that the key schedule of Skein-256 repeats itself every 20 rounds when there is no tweak difference. Therefore, we build trails with two inactive blocks as shown in Figure 3: the difference introduced in the initial state by k_0 cancels out with the difference introduced in the final state by k_5.

We give a collision pair built using this strategy in Table 4.

We can also extend this path to a free-start near-collision attack against 24-round Skein, if we extend the trail to 4 more rounds at the end. A linearized trail gives near-collisions with 15 active bits, and the cost of finding a conforming pair is negligible before the cost of finding the trail.

4.6 Free-Tweak Free-Start Near-Collision Attack

Finally, we can use degrees of freedom in the tweak to reach the maximum number of rounds possible. Previous works have shown that the key schedule allows to have one round without any active key words if we use a difference in the tweak in order to cancel a difference in the key. Using this property we can

build a 24-round trail, and extend it to 32 round by propagating the external difference for four extra rounds in each direction, as shown in Figure 4. This is the approach used in [31].

We give a characteristic built using this idea in the full version of this paper. This results in a low weight difference for the input and output, with many zero bits in predetermined position. Moreover, we can follow the approach of [31] and also specify a fixed characteristic for round 0 to 4 and 28 to 32. It costs about 2^{40} to build a characteristic that allows 2^{20} solutions, so we can estimate that the amortized costs of building a valid pair for rounds 4 to 28 is about 2^{20}. Using the analysis of [31], we would build a conforming pair for rounds 0 to 32 for a cost of $2^{20+43+43} = 2^{119}$, assuming that we can find 2^{66} different characteristics.

Alternatively, if we can choose the value of the tweak, then we only need a single characteristic, and we follow the same attack as [33] with the same complexity.

Note that the complexity of these attack is higher than the generic complexity of a partial-collision attack on $256 - 51$ pre-specified bits, $2^{102.5}$. However, the generic complexity to reach the fixed 256-bit difference with 51 pre-sepcified active bits is still 2^{128}. Alternatively, this attack can be considered as a q-multicollision attack [2].

Conclusion

In this paper we describe an algorithm to build differential characteristics for ARX designs, and we apply the algorithm to find characteristics for various attack scenarios on Skein. Our attacks do not threaten the security of Skein, but we achieve good results when compared to previous attacks; our main results are low-complexity attacks in relatively strong settings. In particular, we show practical free-start and semi-free-start collision attacks for 20 rounds and 12 rounds of Skein-256, respectively.

We obtain some of the first complex differential trails for pure ARX functions (as opposed to MD/SHA-like functions with Boolean functions). Since our approach is rather generic, we expect that our technique can be applied to other ARX designs, and will be used to evaluate the security of these designs against differential cryptanalysis.

Our improvements to the tools of [14], and the code to build differential characteristics for Skein are publicly available from the ARXtools webpage: `http://www.di.ens.fr/~leurent/arxtools.html`. We hope that this will promote cooperation between researchers, and avoid a situation where several teams have to develop their own implementation.

Acknowledgement. We would like to thank Pierre-Alain Fouque and Thomas Peyrin for fruitful discussions about differential characteristics and propagation of constraints.

The author is supported by the ERC project CRASH. Part of this work was done while the author was at the university of Luxembourg, supported by the AFR grant PDR-10-022 of the FNR.

Computational resources have been provided by HPC facility of the University of Luxembourg, as well as by the supercomputing facilities of the Université catholique de Louvain (CISM/UCL) and the Consortium des Équipements de Calcul Intensif en Fédération Wallonie Bruxelles (CÉCI) funded by the Fond de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under convention 2.5020.11.

References

1. Aumasson, J.-P., Bernstein, D.J.: SipHash: A fast short-input PRF. In: Galbraith, S., Nandi, M. (eds.) INDOCRYPT 2012. LNCS, vol. 7668, pp. 489–508. Springer, Heidelberg (2012)
2. Biryukov, A., Khovratovich, D., Nikolic, I.: Distinguisher and Related-Key Attack on the Full AES-256. In: [12], pp. 231–249
3. Canteaut, A. (ed.): FSE 2012. LNCS, vol. 7549. Springer, Heidelberg (2012)
4. Chabaud, F., Joux, A.: Differential Collisions in SHA-0. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 56–71. Springer, Heidelberg (1998)
5. Cramer, R. (ed.): EUROCRYPT 2005. LNCS, vol. 3494. Springer, Heidelberg (2005)
6. De Cannière, C., Rechberger, C.: Finding SHA-1 Characteristics: General Results and Applications. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 1–20. Springer, Heidelberg (2006)
7. den Boer, B., Bosselaers, A.: Collisions for the Compression Function of MD-5. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 293–304. Springer, Heidelberg (1994)
8. Dunkelman, O., Khovratovich, D.: Iterative differentials, symmetries, and message modification in BLAKE-256. In: ECRYPT2 Hash Workshop (2011)
9. Ferguson, N., Lucks, S., Schneier, B., Whiting, D., Bellare, M., Kohno, T., Callas, J., Walker, J.: The Skein hash function family. Submission to NIST (2008/2010)
10. Fouque, P.A., Leurent, G., Nguyen, P.: Automatic Search of Differential Path in MD4. In: ECRYPT Hash Workshop (2007), `http://eprint.iacr.org/2007/206`
11. Fouque, P.-A., Leurent, G., Nguyen, P.Q.: Full Key-Recovery Attacks on HMAC/NMAC-MD4 and NMAC-MD5. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 13–30. Springer, Heidelberg (2007)
12. Halevi, S. (ed.): CRYPTO 2009. LNCS, vol. 5677. Springer, Heidelberg (2009)
13. Khovratovich, D., Rechberger, C., Savelieva, A.: Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 Family. In: [3], pp. 244–263
14. Leurent, G.: Analysis of Differential Attacks in ARX Constructions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 226–243. Springer, Heidelberg (2012)
15. Li, J., Isobe, T., Shibutani, K.: Converting meet-in-the-middle preimage attack into pseudo collision attack: Application to SHA-2. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 264–286. Springer, Heidelberg (2012)
16. Mendel, F., Nad, T., Schläffer, M.: Finding collisions for round-reduced SM3. In: Dawson, E. (ed.) CT-RSA 2013. LNCS, vol. 7779, pp. 174–188. Springer, Heidelberg (2013)
17. Mendel, F., Nad, T., Schläffer, M.: Finding SHA-2 Characteristics: Searching through a Minefield of Contradictions. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 288–307. Springer, Heidelberg (2011)

18. Mendel, F., Nad, T., Schläffer, M.: Collision Attacks on the Reduced Dual-Stream Hash Function RIPEMD-128. In: [3], pp. 226–243
19. Mendel, F., Rechberger, C., Schläffer, M.: MD5 Is Weaker Than Weak: Attacks on Concatenated Combiners. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 144–161. Springer, Heidelberg (2009)
20. Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: The Rebound Attack: Cryptanalysis of Reduced Whirlpool and `Grøstl`. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 260–276. Springer, Heidelberg (2009)
21. Mouha, N., De Cannière, C., Indesteege, S., Preneel, B.: Finding Collisions for a 45-Step Simplified HAS-V. In: Youm, H.Y., Yung, M. (eds.) WISA 2009. LNCS, vol. 5932, pp. 206–225. Springer, Heidelberg (2009)
22. Mouha, N., Velichkov, V., De Cannière, C., Preneel, B.: The Differential Analysis of S-Functions. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 36–56. Springer, Heidelberg (2011)
23. Peyrin, T.: Analyse de fonctions de hachage cryptographiques. PhD thesis, University of Versailles (2008)
24. Schläffer, M., Oswald, E.: Searching for Differential Paths in MD4. In: Robshaw, M. (ed.) FSE 2006. LNCS, vol. 4047, pp. 242–261. Springer, Heidelberg (2006)
25. Stevens, M., Lenstra, A.K., de Weger, B.: Chosen-Prefix Collisions for MD5 and Colliding X.509 Certificates for Different Identities. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 1–22. Springer, Heidelberg (2007)
26. Stevens, M., Sotirov, A., Appelbaum, J., Lenstra, A.K., Molnar, D., Osvik, D.A., de Weger, B.: Short Chosen-Prefix Collisions for MD5 and the Creation of a Rogue CA Certificate. In: [12], pp. 55–69
27. Su, B., Wu, W., Wu, S., Dong, L.: Near-Collisions on the Reduced-Round Compression Functions of Skein and BLAKE. In: Heng, S.-H., Wright, R.N., Goi, B.-M. (eds.) CANS 2010. LNCS, vol. 6467, pp. 124–139. Springer, Heidelberg (2010)
28. Wang, X., Lai, X., Feng, D., Chen, H., Yu, X.: Cryptanalysis of the Hash Functions MD4 and RIPEMD. In: [5], pp. 1–18
29. Wang, X., Yin, Y.L., Yu, H.: Finding Collisions in the Full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer, Heidelberg (2005)
30. Wang, X., Yu, H.: How to Break MD5 and Other Hash Functions. In: [5], pp. 19–35
31. Yu, H., Chen, J., Jia, K., Wang, X.: Near-Collision Attack on the Step-Reduced Compression Function of Skein-256. IACR Cryptology ePrint Archive, Report 2011/148 (2011)
32. Yu, H., Chen, J., Wang, X.: The Boomerang Attacks on the Round-Reduced Skein-512. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 287–303. Springer, Heidelberg (2013)
33. Yu, H., Chen, J., Wang, X.: Partial-Collision Attack on the Round-Reduced Compression Function of Skein-256. In: Moriai, S. (ed.) FSE. LNCS, Springer (2013)

A Collision Pairs for Reduced Compression Functions of Skein-256

In this section we give some examples of colliding pair for reduced versions of the compression function of Skein-256. The differential characteristics used to build those pair are given in the full version of the paper.

The inputs are $k = cv$ and $v_0 = m$. The output is $h = E_k(m) \oplus m$. The examples are given for $T = 0$.

Table 3. Semi-free-start collision for 12-round Skein-256 (rounds 0–12)

	Input 1	Input 2		Output 1	Output 2
k_0	968cb2e66b0fb527	968cb2e66b0fb527	$e_{12,0}$	2798a30c07459007	2398930c07459007
k_1	37fce3361809b06a	37fce3361809b06a	$e_{12,1}$	2410f135e024aace	2410e135e024aace
k_2	4bb032fb1894a60b	4bb032fb1894a60b	$e_{12,2}$	60490bbd9ddcb933	60490bbd9ddcb933
k_3	d917aa4640682db6	d917aa4640682db6	$e_{12,3}$	7fd51384c7b528f3	7fd51384c7b528f3
m_0	e7395021238d7d18	e3396021238d7d18	h_0	c0a1f32d24c8ed1f	c0a1f32d24c8ed1f
m_1	7229b06628958c1a	7229a06628958c1a	h_1	56394153c8b126d4	56394153c8b126d4
m_2	3ea410b0b8f1b533	3ea410b0b8f1b533	h_2	5eed1b0d252d0c00	5eed1b0d252d0c00
m_3	fc0aa7147201f560	fc0aa7147201f560	h_3	83dfb490b5b4dd93	83dfb490b5b4dd93

Table 4. Free-start collision for 20-round Skein-256 (rounds 0–20)

	Input 1	Input 2		Output 1	Output 2
k_0	5f977cfdd64d2f57	5f977cfdd64d2f57	$e_{20,0}$	6627a3d5c18e2057	6627a3d5c18e2057
k_1	35839193022be6f4	b5839193022be6f4	$e_{20,1}$	7a1eeeee92b2202d	fa1eeeee92b2202d
k_2	05e168930700458f	85e168930700458f	$e_{20,2}$	2bf3a5067fac9218	abf3a5067fac9218
k_3	6f47d57f8b6f9b78	6f47d57f8b6f9b78	$e_{20,3}$	b0ccc2f709dc2e35	b0ccc2f709dc2e35
m_0	627f37f95152438c	627f37f95152438c	h_0	0458942c90dc63db	0458942c90dc63db
m_1	0532b3fdf499d0d7	8532b3fdf499d0d7	h_1	7f2c5d13662bf0fa	7f2c5d13662bf0fa
m_2	91c792ab31ba535c	11c792ab31ba535c	h_2	ba3437ad4e16c144	ba3437ad4e16c144
m_3	72e80ac1aaee8118	72e80ac1aaee8118	h_3	c224c836a332af2d	c224c836a332af2d

On Fair Exchange, Fair Coins and Fair Sampling*

Shashank Agrawal and Manoj Prabhakaran

University of Illinois, Urbana-Champaign
{sagrawl2,mmp}@illinois.edu

Abstract. We study various classical secure computation problems in the context of fairness, and relate them with each other. We also systematically study fair sampling problems (i.e., inputless functionalities) and discover three levels of complexity for them.

Our results include the following:

- Fair exchange cannot be securely reduced to the problem of fair coin-tossing by an r-round protocol, except with an error that is $\Omega(\frac{1}{r})$.
- Finite fair *sampling* problems with rational probabilities can all be reduced to fair coin-tossing and unfair 2-party computation (or equivalently, under computational assumptions). Thus, for this class of functionalities, fair coin-tossing is complete.
- Only sampling problems which have fair protocols without any fair setup are the trivial ones in which the two parties can sample their outputs independently. Others all have an $\Omega(\frac{1}{r})$ error, roughly matching an upperbound for fair sampling from [21].
- We study communication-less protocols for sampling, given another sampling problem as setup, since such protocols are inherently fair. We use spectral graph theoretic tools to show that it is impossible to reduce a sampling problem with *common information* (like fair coin-tossing) to a sampling problem without (like "noisy" coin-tossing, which has a small probability of disagreement).

The last result above is a slightly sharper version of a classical result by Witsenhausen from 1975. Our proof reveals the connection between the tool used by Witsenhausen, namely "maximal correlation," and spectral graph theoretic tools like Cheeger inequality.

1 Introduction

Despite wide interest in the problem of fairness, our understanding of some of the most fundamental questions about it is greatly lacking. In this work, we study fair-exchange, fair coin-flipping, and more generally fair sampling, to understand the relation between these primitives. In the process, we also obtain a sharper version of a classical information theory result from the 70's on common information of correlated random variables.

* Research supported in part by NSF grants 1228856 and 0747027.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 259–276, 2013.

Fair coin-flipping and fair-exchange are two classical problems in cryptography, with a long history of results, both positive and negative. The most influential, and perhaps the most important negative result, dates back to the work of Cleve [8], in which a deceptively simple argument was used to prove a result of great consequence: irrespective of what computational assumptions are used, any 2-party coin-flipping protocol is vulnerable to a simple attack which can produce a bias that is inversely proportional to the number of rounds of the protocol (rather than being negligible, as one would have preferred).

Our first result relates fair coin-flipping to fair-exchange. It is easy to see that a fair exchange functionality can be directly used to obtain a fair coin-flipping protocol (and thus Cleve's impossibility for fair coin-flipping implies impossibility of fair-exchange as well). We ask if fair coin-flipping and fair-exchange are *equivalent*, possibly under some computational assumption. That is, *given access to a fair coin-flipping functionality, can we implement a fair-exchange protocol?*

The answer turns out to be negative: we show that an efficient attack can break the security of any fair-exchange protocol that has access to fair coin-flipping, in the same way Cleve's attack could break the security of any fair coin-flipping protocol. Our attack, like Cleve's, is a simple fail-stop attack, and does not rely on any computation other than running the steps of the protocol itself. However, it differs from Cleve's in some essential ways, in order to handle the presence of the fair coin-flipping functionality. (In particular, one of our attacks requires the adversary to run a particular round of the protocol twice, to "look-ahead" before actually accessing the coin-flipping functionality.)

Our other results relate to the problem of *fair sampling*. This is a generalization of the fair coin-flipping problem, in which it is not necessary that Alice and Bob output the same bit, but instead they are required to produce outputs that are correlated in a specified manner. While somewhat more subtle than the problem of fair coin-flipping, one can use a natural (standalone) simulation based security definition to get the right definition of fair and secure sampling. Surprisingly, we show that fair coin-flipping is at least as "complex" as generating correlated outputs from various distributions like noisy coin-flipping (where each party gets an unbiased coin, but with probability say, 0.1, their coins do not agree), random-OT (where Alice gets two random bits (x_0, x_1) and Bob gets (b, x_b) for a random b). That is, all these fair sampling problems can be solved with access to a fair coin-flipping functionality (under standard computational assumptions, or alternately, with access to *unfair* 2-party computation functionalities). On the other hand, we believe the converse does not hold in general. We give results (including one of independent interest) that show that somewhat restricted protocols cannot give fair coin-flipping from fair sampling functionalities if the distribution does not provide any *common information* (as formalized by [11]) to the two parties.

Two points are worth highlighting here. In standard (unfair) secure 2-party computation, the "complexity" of coin-flipping and that of say, noisy coin-flipping are inverted. Indeed, noisy coin-flipping is a *complete* functionality for unfair 2-party computation in the information-theoretic setting, whereas coin-flipping is

not (see for e.g. [18]). The second point is that, noisy coin-flipping and random OT, though possibly strictly simpler than coin-flipping itself, still turn out to be impossible to fairly and securely implement, irrespective of any computational assumptions or setups. We emphasize that these are sampling problems, and should not be confused with (automatically fair) functionalities like OT (with inputs). We generalize the proof of Cleve to show that unless a 2-party distribution is trivial (i.e., the outputs for the two parties are independent of each other), it does not have a fair protocol. In fact, our proof leads to a slight simplification of Cleve's argument, but without yielding any quantitative improvements.

Finally, an important contribution of our work on fair sampling is a deeper understanding of *common information*, a concept introduced by Gács and Körner [11], and since then widely studied in the information theory literature. Roughly speaking, common information of a 2-party distribution is a piece of information two parties can agree on, after they obtain a sample from the 2-party distribution (with each party obtaining only its part of the output). Distributions like noisy coin-flipping, and random OT have no common information. Gács and Körner showed that, even if a large number of samples from such a distribution are given to the two parties, if they must agree on a common output without further communication, then the *entropy rate* of their outputs must be zero. Our interest in this setting, where the parties have to agree on an output without any communication, is because such a protocol is inherently fair (provided the access to the samples are fair). The original proof of Gács and Körner used tools from ergodic theory to show that the number of independent random bits that Alice and Bob can agree on is $o(n)$ if they access n samples from a distribution with zero common information. Witsenhausen used maximal correlation [12,22] to show that they cannot agree on an output with any positive entropy (not entropy rate) except by suffering a constant probability of disagreement [23]. We reprove this result using tools from spectral graph theory. Technically, our proof is quite similar to that in [19] who refined Witsenhausen's proof that used maximal correlation. However, by identifying the connection with spectral graph theory, we are able to obtain a slightly sharper result, afforded to us by Cheeger's inequality [5].

Our Results. We provide a collection of results on fair 2-party computation (all of which also extend to the case of multi-party computation without honest majority). Our focus is on studying certain important and representative tasks, rather than attempting to exhaustively characterize fairness of all the tasks. The following three canonical problems can be used to explain our main results.

- Fair exchange $\mathcal{F}_{\text{EXCH}}$. Alice and Bob exchange a single bit.
- Fair coin-tossing $\mathcal{F}_{\text{COIN}}$. Alice and Bob obtain a common coin.
- Fair sampling of a random instance of oblivious transfer $\mathcal{F}_{\text{ROT}}$: Alice gets a pair of random bits (x_0, x_1) and Bob gets (b, x_b) where b is a random bit.

We show that these three functionalities have decreasing complexity in the context of fairness:

1. $\mathcal{F}_{\text{EXCH}} > \mathcal{F}_{\text{COIN}}$: In Section 3, we show that $\mathcal{F}_{\text{EXCH}}$ cannot be reduced to $\mathcal{F}_{\text{COIN}}$, irrespective of what computational assumptions are made. We show that for any r-round protocol for $\mathcal{F}_{\text{EXCH}}$ using $\mathcal{F}_{\text{COIN}}$, there is an efficient fail-stop adversary for which the simulation error is $\Omega(\frac{1}{r})$. On the other hand, it is well-known that $\mathcal{F}_{\text{COIN}}$ can be reduced to $\mathcal{F}_{\text{EXCH}}$.
2. $\mathcal{F}_{\text{COIN}} \geq \mathcal{F}_{\text{ROT}}$: In Section 4 we show that $\mathcal{F}_{\text{ROT}}$ can be reduced to $\mathcal{F}_{\text{COIN}}$ (and an unfair 2-party computation problem). This protocol involves no communication between the two parties, except for them both accessing an unfair sampling functionality, and $\mathcal{F}_{\text{COIN}}$. This protocols extends beyond $\mathcal{F}_{\text{ROT}}$, and shows that $\mathcal{F}_{\text{COIN}}$ is complete with respect to fair and secure reductions at least for a class of "nice" sampling tasks (including $\mathcal{F}_{\text{ROT}}$).
3. $\mathcal{F}_{\text{COIN}} > \mathcal{F}_{\text{ROT}}$? We do not completely rule out a reduction of $\mathcal{F}_{\text{COIN}}$ to $\mathcal{F}_{\text{ROT}}$. However, we present important partial negative results in Section 5. In particular, we show that there is no *logarithmic round reduction* from coin flipping to a distribution with zero common information. (Also see below.)
4. $\mathcal{F}_{\text{ROT}}$ non-trivial: Though $\mathcal{F}_{\text{ROT}}$ is at the bottom of this list, in Section 6 we show that it cannot be fairly sampled either (irrespective of the computational assumptions used). Here we have a tight characterization: only distributions that can be fairly sampled are the ones in which there is no correlation between Alice's and Bob's outputs. Our result is also tight in that the bias we obtain closely matches a positive result from [21].

In Section 5 we investigate a sub-class of protocols for fair sampling, in which the two parties access samples from a setup functionality, and then, without any communication, produce their outputs. Such protocols are inherently fair. The question of when such protocols are possible presents interesting combinatorial and information-theoretic questions.

Using tools from spectral graph theory to analyze an appropriately defined graph product (or rather, bipartite graph product), we show that even with an unbounded number of samples from $\mathcal{F}_{\text{ROT}}$, any such protocol for $\mathcal{F}_{\text{COIN}}$ will have a *constant amount* of error. Specifically, we give a tight bound on the second eigenvalue of the normalized Laplacian of $G_1 \boxtimes G_2$ in terms of that of G_1 and G_2, where $\boxtimes$ is a natural bipartite graph product that we define. Our result sharpens a classical result on "common information" from information theory, originally proven by Gács and Körner [11] using techniques from ergodic theory and subsequently improved by Witsenhausen [23] using the "maximal correlation" measure [12,22]. As it turns out, our spectral graph theoretic proof is very similar to the one using maximal correlation as reformulated in [19], but we can make a slightly sharper statement relating the error when using one sample from $\mathcal{F}_{\text{ROT}}$ versus using an unbounded number of samples, thanks to Cheeger inequality.

This result also goes beyond $\mathcal{F}_{\text{ROT}}$, and in fact gives a tight characterization of which sampling problems allow $\mathcal{F}_{\text{COIN}}$ to be reduced to them, and which ones do not: any 2-party distribution (i.e., a pair of correlated random variables) which has non-zero "common information" can be used to implement $\mathcal{F}_{\text{COIN}}$, where common information is as was defined in the seminal work of [11].

We believe the explicit connection between spectral graph theory and tools in information theory is of independent interest, and holds promise for other problems.

An Emerging Picture. While the focus on this work has been to study specific functionalities, our results suggest a certain hierarchy of "complexity" of functionalities. Firstly, in general fair functions with input (like XOR) can be strictly more complex than fair sampling problems. We leave it open to study distinctions within functions with input (e.g., both parties having input vs. only one party having input). Our other results have explored variations among fair sampling problems. There are three apparent classes here: trivial problems (which can be sampled trivially, by both parties independently generating their outputs), non-trivial problems with zero common information (which includes $\mathcal{F}_{\text{ROT}}$ and noisy coin-flip), and problems with non-zero common information (which includes $\mathcal{F}_{\text{COIN}}$). Indeed, the last class is complete for all sampling problems with rational probabilities. The qualitative separation between problems with and without common information is formalized in the setting of protocols without communication.

Related Work. The problem of fairness in multi-party computation goes back to the work of Even and Yacobi [10] where exchange of digital signatures is informally proved to be impossible. The first rigorous proof of the impossibility of fairly computing a functionality comes from Cleve's work [8]. He showed that a very basic functionality, that of tossing a coin, cannot be realized fairly. Subsequent works like that of [9] which considered stronger attacks, relied on computationally unbounded adversaries.

A recent series of results has renewed interest in the area of fairness. Starting with the work of Gordon et al. [14], where they show that several functionalities of interest can be realized with complete fairness, there has been a series of results in this area. In [21], Moran et al. solve a long standing open problem in fairness. They show that Cleve's lower bound on the bias of coin tossing protocols can be achieved (up to a factor of 2) by a protocol. Beimel et al. [3] extend their results to the multi-party model when less that 2/3 of the parties are corrupt. In [15], Gordon et al. study the question of reductions among fair functionalities. They show that no *short* primitive is complete for fairness. They also establish a fairness hierarchy for simultaneous broadcast. Further in [13], a definition of partial fairness is proposed, and it is shown that any two-party functionality, at least one of whose domains or ranges is polynomial in size, can be realized fairly under this definition. Beimel et al. [2] study partial fairness in the multi-party setting. Asharov et al. [1] provide a complete characterization of functions that imply fair coin tossing, and hence cannot be computed fairly due to Cleve's impossibility result. The negative results in this work relied on computationally unbounded adversaries.

Separations of the kind we consider (impossiblity of reducing XOR to coin-flipping) was also considered in the context of security with abort, but in the

computationally unbounded setting [20]. We remark that such a result does not hold in the computationally bounded setting.

The notion of common information was introduced by [11], and further developed in [24,16,23] and many later works. The problem of obtaining isoperimetric inequalities of graph products has been studied, but for notions of graph products different from the bipartite product we study (e.g. [7], also see [6]).

2 Preliminaries

2.1 Secure Two-Party Computation with Complete Fairness

We are interested in (possibly randomized) two-party secure function evaluation with complete fairness (in contrast to security with abort). The functionalities we consider are all finite and their domains and ranges remain constant, irrespective of the security parameter. All entities considered are probabilistic polynomial time (PPT). They are given the security parameter n as an auxiliary input, and their total running time is polynomial in n.

Fairness is modeled by specifying the ideal functionality $\mathcal{F}$ to be fair: it delivers the output to both parties together. A corrupt party controlled by the adversary may explicitly instruct the ideal functionality to abort (or, provide $\perp$ as its input) without receiving any information from the functionality; but if both parties provide valid inputs, then the functionality will evaluate a specified function of the inputs and provide the results to the parties. Let $\text{IDEAL}_{\mathcal{F},\mathcal{S}}(n) = (\text{VIEW}_{\mathcal{F},\mathcal{S}}(n), \text{OUT}_{\mathcal{F},\mathcal{S}}(n))$ be the random variable that denotes the output of the adversary and the output of the honest party in the ideal world.

In the real world, instead of outsourcing the computation, parties run a protocol π which enables them to compute $\mathcal{F}$. While the honest party follows the instructions of π, the corrupt party controlled by the adversary may deviate arbitrarily. Let $\text{REAL}_{\pi,\mathcal{A}}(n) = (\text{VIEW}_{\pi,\mathcal{A}}(n), \text{OUT}_{\pi,\mathcal{A}}(n))$ be the random variable that denotes the view of the adversary and the output of the honest party.

A more detailed description of the ideal and real executions can be found in the full version of the paper. In proving our negative results, we use the following weak simulation based security definition.

Definition 1 (Weak Security). *A protocol π is said to be a weak ϵ-secure realization of a two party functionality $\mathcal{F}$ if for every PPT adversary $\mathcal{A}$ in the real world, there exists a PPT adversary $\mathcal{S}$ in the ideal world such that*

$$\Delta\left(\text{IDEAL}_{\mathcal{F},\mathcal{S}}(n), \text{REAL}_{\pi,\mathcal{A}}(n)\right) \leq \epsilon(n).$$

We say that π is a weak secure realization of $\mathcal{F}$, if it is a weak ϵ-secure realization of $\mathcal{F}$ for a negligible function $\epsilon(n)$.

Our definition is similar to the one given in [13], except that we do not require security to hold in the presence of auxiliary information, which makes our definition weaker. Note that using a weaker security definition only strengthens the impossibility results. On the other hand, we remark that our positive results, i.e., constructions, are in fact UC secure [4].

2.2 Normal Form of a Protocol

We shall use the following normal form for a 2-party protocol π between Alice and Bob. The number of rounds of the protocol will be denoted by $r(n)$, where n is the security parameter. In this protocol, parties may also have access to a *setup functionality* $\mathcal{G}$. We shall often refer to such a setup functionality as an *oracle*. Without loss of generality, we assume that the i^{th} round in π consists of the following steps, for $1 \leq i \leq r(n)$:

- Alice sends a message to Bob; if Alice aborts without sending this message, Bob produces an output, denoted by the random variable Y_{i-1}.
- the functionality $\mathcal{G}$ is invoked; if this invocation is aborted, Alice and Bob would produce outputs.
- then Bob sends a message to Alice; if Bob aborts without sending this message, Alice produces an output, denoted by the random variable X_i.
- $\mathcal{G}$ is invoked once again; again, if this invocation is aborted, Alice and Bob would produce outputs.

In all our results, the functionality $\mathcal{G}$ will be an inputless function, and the particular attacks we use do not involve aborting its invocation. So we have not given any names for the random variables corresponding to the outputs if $\mathcal{G}$'s invocation is aborted. If multiple setups, say $\mathcal{G}_1$ and $\mathcal{G}_2$, are available, they will be invoked one after the other in every round.

We remark that what makes proving our impossibility results harder is that the protocol π can access $\mathcal{F}_{\text{COIN}}$ throughout its execution, rather than only in a pre-processing phase. Indeed, it has been observed before by Ishai et al. [17] that the impossibility results for fair deterministic function evaluation in the plain model continue to hold in a pre-processing model.[1]

3 Fair Exchange from Fair Coin-Flipping

In this section, our goal is to show that two parties cannot exchange their bits fairly, even when given access to fair coin-flipping functionality. The $\mathcal{F}_{\text{COIN}}$ functionality does not take any input and provides a bit uniformly distributed in $\{0,1\}$ to the two parties. The $\mathcal{F}_{\text{EXCH}}$ functionality is also simple to state: if $x, y \in \{0,1\}$, then $\mathcal{F}_{\text{EXCH}}(x,y) = (y,x)$; but if one of the parties aborts or sends an invalid input to it, the functionality substitutes its input by a default value, say 0.[2] Recall our convention that this is a fair functionality, so the adversary cannot prevent the delivery of output to the honest party.

[1] The observation in [17] considers not just deterministic function evaluation. However in the general case, the impossibility of fairness there holds only under a stricter requirement, that the correctness of the protocol should hold conditioned on the randomness of the pre-processing phase. In particular, $\mathcal{F}_{\text{COIN}}$ does not reduce to $\mathcal{F}_{\text{COIN}}$ in such a pre-processing model. Our results are not restricted to the pre-processing model, nor depend on such a security requirement.

[2] An alternate formulation would be that if $(x,y) \notin \{0,1\}^2$, then $\mathcal{F}_{\text{EXCH}}(x,y) = (\perp, \perp)$ where $\perp$ is a special symbol indicating abort. It can be easily seen that these formulations are "isomorphic" to each other (see following text).

We define another functionality $\mathcal{F}_{\text{XOR}}$ which takes inputs x and y from the two parties. If $x, y \in \{0,1\}$, then $\mathcal{F}_{\text{XOR}}(x,y) = (x \oplus y, x \oplus y)$; but if one of the parties aborts or sends an invalid input, the functionality substitutes its input by a default value, say 0 (similar to what $\mathcal{F}_{\text{EXCH}}$ does above). The functionality $\mathcal{F}_{\text{XOR}}$ is "isomorphic" to $\mathcal{F}_{\text{EXCH}}$: that is, each functionality can be (UC) securely reduced to the other using a protocol that involves no other communication other than a single invocation of the latter functionality. Then it is easy to see that the fair functionality $\mathcal{F}_{\text{EXCH}}$ can be (weakly) securely realized (using any set up) if and only if the fair functionality $\mathcal{F}_{\text{XOR}}$ can be (weakly) securely realized (using the same set up). This allows us to prove that $\mathcal{F}_{\text{EXCH}}$ cannot be reduced to $\mathcal{F}_{\text{COIN}}$ by showing instead that $\mathcal{F}_{\text{XOR}}$ cannot be reduced to $\mathcal{F}_{\text{COIN}}$.

The result of this section follows. A formal proof can be found in the full version of the paper. Here we provide a sketch which describes the main ideas involved in the proof. We point out that the result is tight up to a constant, since [13] shows that $\mathcal{F}_{\text{XOR}}$ can be computed ϵ-securely in $O(1/\epsilon(n))$ rounds even without access to $\mathcal{F}_{\text{COIN}}$.

Theorem 1. *For any weakly ϵ-secure protocol $\pi^{\mathcal{F}_{\text{COIN}}}$ that realizes the functionality $\mathcal{F}_{\text{XOR}}$ and runs in $r(n)$ rounds, $r(n) \in \Omega(\frac{1}{\epsilon(n)})$.*

Proof sketch: Similar to Cleve's approach [8], we shall consider a collection of fail-stop adversaries that corrupt either Alice or Bob. We shall also consider the case when neither party is corrupt. We seek to argue that at least for one of these adversaries, the outcome in the real experiment cannot be simulated within a $\Omega(\frac{1}{r})$ error by any simulator in the ideal world, where r is the number of rounds in $\pi^{\mathcal{F}_{\text{COIN}}}$. ($r$ is a function of the security parameter n, but for the sake of readability, we write r instead of $r(n)$.)

We start off along the same lines as Cleve: we note that at the end of the protocol, the parties will agree on their outcome (except with at most ϵ probability). On the other hand, in the beginning of the protocol, the variables Y_0 and X_1 are independent of each other; also, by considering an adversary who forces an abort right at the beginning, each of Y_0 and X_1 should be close to uniformly random. So Y_0 and X_1 are equal with probability only about half. Thus there must be a round i such that $\mathsf{Pr}[X_i = Y_i] - \mathsf{Pr}[X_i = Y_{i-1}] = \Omega(\frac{1}{r})$ (or $\mathsf{Pr}[X_{i+1} = Y_i] - \mathsf{Pr}[X_i = Y_i] = \Omega(\frac{1}{r})$; w.l.o.g, we can consider the former to be the case).

In Cleve's case, where the protocol he considers does not have access to any setup, we can consider two adversaries that corrupt Alice, and selectively abort at round i as follows. (See Section 2 for the numbering of the rounds.) The first adversary forces Bob to output Y_{i-1} if $X_i = 0$ and otherwise forces him to output Y_i; the second adversary does the same for $X_i = 1$. To ensure that Bob's output is unbiased under these two attacks requires that

$$\mathsf{Pr}[Y_{i-1} = 0 \wedge X_i = 0] \approx \mathsf{Pr}[Y_i = 0 \wedge X_i = 0],$$
$$\mathsf{Pr}[Y_{i-1} = 1 \wedge X_i = 1] \approx \mathsf{Pr}[Y_i = 1 \wedge X_i = 1].$$

This contradicts the assumption that $\mathsf{Pr}[X_i = Y_i] - \mathsf{Pr}[X_i = Y_{i-1}] = \Omega(\frac{1}{r})$. A crucial element in this proof is that Alice can compute X_i first and then selectively force Bob to output Y_{i-1}.

Unfortunately, but not surprisingly, this breaks down when the parties have access to the $\mathcal{F}_{\text{COIN}}$ oracle as in our case. To compute X_i, Alice must obtain the first coin in round i. But after that she cannot force Bob to output Y_{i-1}: he will output only Y_i (note that aborting an access to $\mathcal{F}_{\text{COIN}}$ cannot help, because w.l.o.g, the protocol can instruct a party to substitute it with a coin it generates). Indeed, we cannot expect Cleve's argument to go through when an $\mathcal{F}_{\text{COIN}}$ oracle is present, because fair coin-flipping is trivially possible given access to $\mathcal{F}_{\text{COIN}}$.

The reason we can expect to have an attack nevertheless, has to do with the fact that there is an additional correctness requirement in the case of $\mathcal{F}_{\text{XOR}}$ that is not present in the case of coin-flipping. For instance, if the parties were to output a coin they obtain in round i as their final output, while none of the attacks can bias this outcome, when the execution is carried out without any corruption, the output will be different from the XOR of the input with probability $\frac{1}{2}$.

We leverage this fact in a somewhat non-obvious manner. Suppose we want to run Cleve's attacks as well as we can. The two adversaries described above can proceed right up to the point before accessing $\mathcal{F}_{\text{COIN}}$ in round i. Then, without invoking $\mathcal{F}_{\text{COIN}}$, the attacker can check what the value of X_i would be for each of the two possible outcomes from $\mathcal{F}_{\text{COIN}}$ (by feeding one value of the coin to the honest protocol execution, then rewinding it, and feeding the other value of the coin). If in both cases the outcome is the same, then the adversary manages to find X_i without invoking $\mathcal{F}_{\text{COIN}}$ at all. Let E_i^A denote this event that in an honest execution of the protocol, at the point before invoking the first access to $\mathcal{F}_{\text{COIN}}$ in round i, the value of X_i already gets determined.

But what happens if the complement event $\overline{E}_i^A$ occurs? In this case, X_i is a 0 with probability $\frac{1}{2}$ and 1 with probability $\frac{1}{2}$. Further, this happens independently of Y_{i-1}. Note that Y_i could very well be correlated with X_i, since it is influenced by the same coin that decides X_i as well as messages sent by Alice after determining X_i. On the other hand, the final output of Bob, Y_r must be (almost) independent of X_i, since it must (mostly) equal the XOR of the inputs, which is fixed well before the coin from $\mathcal{F}_{\text{COIN}}$ is accessed. Thus, X_i is correlated almost the same way (i.e., uncorrelated) with both Y_{i-1} and Y_r.

This gives us a way to emulate the effect of forcing the outcome to be Y_{i-1} when X_i comes out a particular way, provided $\overline{E}_i^A$ occurs: instead of trying to force Bob to output Y_{i-1} (for which it is too late), let the protocol run to completion and force his outcome to be Y_r.

Somewhat surprisingly, this intuition can be turned into a concrete argument. We employ adversaries for each round which check if the event E_i^A occurs, and adopt one of the above strategies. Note that the adversary can efficiently determine if the event E_i^A occurs (without accessing the corresponding instance of $\mathcal{F}_{\text{COIN}}$). □

In the full version, we actually prove a generalization of Theorem 1. We show that no δ-balanced function [1] can be securely reduced to $\mathcal{F}_{\text{COIN}}$. The XOR

function, which is δ-balanced with $\delta = 1/2$, is therefore not reducible to $\mathcal{F}_{\text{COIN}}$ either.

Remark. The proof and the result readily extends to the case when the protocol has access to other *unfair* non-reactive functionalities as well as $\mathcal{F}_{\text{COIN}}$, since in that case Alice can determine whether the event E_i^A occurs (using an unfair access to the functionalities) and act accordingly. Also, a corollary of the generalization of Theorem 1 is that access to any fair functionality that can be securely realized using access to (polynomially many invocations of) a δ-balanced function is not sufficient to obtain a secure fair XOR protocol.

4 Fair Sampling from Fair Coin-Flipping

We shall say that a functionality $\mathcal{F}$ is complete for fair function evaluation if for any fair function evaluation task there is an (information theoretically) secure protocol that uses $\mathcal{F}$ *and optionally, some unfair functionality* $\mathcal{G}$. Allowing access to an unfair functionality eliminates the need to base the completeness result on computational assumptions. (Equivalently, one could define it in terms of a reduction to $\mathcal{F}$ that is secure in the probabilistic polynomial time setting, and assume the existence of oblivious transfer protocol.)

In [15] it was shown that no finite functionality is complete for fair computation, even restricted to finite functionalities. We pose the same question, but restricted to finite sampling functionalities (i.e., functionalities without input).

Surprisingly, we show that fair coin-flipping functionality $\mathcal{F}_{\text{COIN}}$ is in fact complete for this class of problems. We mention a caveat in our result: our protocol for fair sampling requires that the probability values in the target distribution are rational numbers. Note that since the functionalities are finite, there is only a finite constant set of probabilities in question, independent of the security parameter. We say that such distributions are "nice." If the target distribution involves probabilities that are not rational, then even though one could approximate them to negligible error using rational numbers, an initial unfair secure computation phase in our protocol would involve exponentially large outputs. However, even in this case, the number of accesses to $\mathcal{F}_{\text{COIN}}$ is still only polynomial.

Let p_{XY} denote a joint distribution over two random variables X and Y which take values in the finite domains $\mathcal{X}$ and $\mathcal{Y}$ respectively. Our goal is to construct an information-theoretic, UC secure protocol for the functionality which takes no input, but samples (X, Y) according to p_{XY} and gives X to Alice and Y to Bob. This functionality is modeled as a fair functionality as described in Section 2. The protocol has access to an arbitrary unfair 2-party computation problem (in fact, a 2-party sampling problem suffices) and the fair coin tossing function $\mathcal{F}_{\text{COIN}}$.

The basic idea of the protocol is fairly simple: first use an unfair secure computation phase to generate two lists A and B such that if a uniformly random i is picked then (A_i, B_i) will be distributed according to the target distribution. This computation will give the list A to Alice and B to Bob. Then, use (many

accesses to) fair coins to sample an index i; if either party aborts mid-way, the other party simply tosses the remaining coins on its own. Alice's output will be A_i and Bob's output will be B_i. The list A could contain many indices i such that $A_i = x$ for some character x, such that not all of these indices have the same value of B_i. For security of this protocol, it is important that if Alice receives i such that $A_i = x$, she learns nothing more about Bob's output B_i, than what x reveals. This is ensured by randomly permuting the lists A and B.

It remains to describe how the lists A and B can satisfy the above requirements. For this, first, for each $(x, y) \in \mathcal{X} \times \mathcal{Y}$, express the probability $p(x, y)$ as rational number $P_{x,y}/Q$, where Q is the same for all (x, y). Note that this is where we assume that the distribution p_{XY} is nice. Then, for each pair (x, y), add $P_{x,y}$ copies of (x, y) to a list L. The size of this list will be Q. Then randomly permute L to obtain a list $((a_1, b_1), \cdots (a_Q, b_Q))$. A is defined to be the list $(a_1, \cdots, a_Q)$ and B the list $(b_1, \cdots, b_Q)$.

Simulation to prove the security of this protocol is straightforward and omitted.

Despite the restriction to nice distributions, we note that the consequences of this protocol are already quite powerful. The sampling problems mentioned in the introduction — noisy coin-flipping with noise probability 0.1 or random oblivious transfer distribution (in which one of 8 different possibilities occur with the probability 1/8 each) — are covered by this protocol.

We also point out another feature of this protocol: Alice and Bob do not interact with each other, except by accessing two sampling functions: the first one which produces the lists A and B (unfairly) and the second one which gives fair coins.

5 Impossibility of Fair Coin Flip from Fair Sampling

In this section we ask if it is possible to have a fair coin flipping protocol given access to a setup for fairly sampling from a 2-party distribution. As we shall see, this depends on whether the setup distribution gives *non-zero common information* to the two parties. Our definition of common information of a 2-party distribution, adapted from Gács and Körner [11], is best understood in terms of the characteristic bipartite graph representation of a 2-party distribution.

Characteristic Bipartite Graph. Consider a distribution which samples a pair of symbols $(u, v) \in U \times V$ with probability $p(u, v)$ and gives u to Alice and v to Bob. The characteristic bipartite graph (or simply the graph of a distribution) of this distribution is a weighted graph $G = (U, V, w)$ with U and V as the two partite sets, and with weight of the edge between $u \in U$ and $v \in V$ defined to be $w(u, v) = p(u, v)$. Edges with weight 0 are considered absent, and only nodes with at least one edge incident on them are retained in G.

Common Information. In the above setting, consider a function C which maps a sample (u, v) to the index of the connected component in G that contains the edge (u, v) (after removing 0-weight edges). We define the common information of a 2-party distribution as the entropy of the random variable $C(u, v)$ when

(u, v) is sampled from the distribution. In particular, the distribution has zero common information iff G has a single connected component (after removing 0-weight edges and isolated nodes).

For example, 2-party coin flipping has 1 bit of common information, whereas a noisy coin flipping which gives an unbiased coin each to Alice and Bob which are equal only with probability say 0.9, has zero common information.

Conjecture 1. For the class of finite 2-party distributions, the ones that are complete with respect to fair and secure reductions are exactly the ones that have positive common information.

We do not completely resolve this conjecture, but we provide the following results in its evidence:

1. In the positive direction, the conjecture is equivalent to stating that coin flipping is complete (since, as can be easily seen, any distribution with positive common information can be used to obtain fair coins). Our result in Section 4 proves this, restricted to the class of "nice" distributions.
2. In the negative direction, we show that there is no *logarithmic round reduction* from coin flipping to a distribution with zero common information. We present this proof in the full version of the paper.
3. We show that there is no reduction from coin flipping to a distribution with zero common information using a protocol that has (an unbounded number of) rounds which access the setup, followed by a polynomial number of communication rounds. (Our proof does not apply if the accesses to the setup are interspersed with communication.) This can be shown using Theorem 2 below, which deals with the special case when the protocol involves no communication at all.

Theorem 2. *[23] Let p_{UV} be a 2-party distribution with zero common information. Then for every constant $\delta > 0$ there is a constant $\epsilon > 0$ (depending on p_{UV}) such that for any 2-party protocol in which the parties are given an arbitrary number of samples from p_{UV}, but they do not exchange any messages and the entropy of the output of at least one of the parties is at least δ, then with probability at least ϵ the output of the two parties will be different.*

In Appendix A we give a new proof for this result, originally due to [23]. Note that the error probability ϵ does not decrease with the number of coin samples the protocol is allowed access to. Further, Lemma 1 in Appendix A, implies that the error ϵ_0 achievable using *a single sample* from p_{UV} is (for the same δ) $O(\sqrt{\epsilon})$. That is, using more than one sample can decrease the error probability at most quadratically. To the best of our knowledge, this final result was not known previously.

6 Secure Sampling

In this section we consider the task of sampling from a joint distribution (X, Y), where X and Y are distributed over finite domains $\mathcal{X}$ and $\mathcal{Y}$ respectively. Two

parties Alice and Bob wish to sample from this distribution such that Alice learns only the value of X and Bob learns only the value of Y. The functionality for secure sampling $\mathcal{F}_{\text{ss}}$ is very simple: it does not take any input and produces a sample from the distribution (X, Y).

Let $X \times Y$ denote the product distribution of X and Y, i.e., $\mathsf{Pr}[X \times Y = (x, y)] = \mathsf{Pr}[X = x] \cdot \mathsf{Pr}[Y = y]$ for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. We show:

Theorem 3. *For any weakly ϵ-secure protocol π that realizes the functionality $\mathcal{F}_{\text{ss}}$ and runs in $r(n)$ rounds,*

$$r(n) \geq \frac{\Delta((X, Y), X \times Y) - 3\epsilon(n)}{2(|\mathcal{X}| + |\mathcal{Y}|)\epsilon(n)}, \tag{1}$$

$$r(n) \geq \frac{\alpha_{XY} - 3\epsilon(n)}{4\epsilon(n)}, \tag{2}$$

where $\alpha_{XY} = \mathsf{max}_{(x,y)\in(\mathcal{X},\mathcal{Y})} |\mathsf{Pr}[(X, Y) = (x, y)] - \mathsf{Pr}[X \times Y = (x, y)]|$.

In general, the two bounds are incomparable: the nature of joint distribution decides which one is stronger (for examples see the full version). Our first bound closely matches an upper bound from [21], who give an ϵ-secure sampling protocol with $\frac{\Delta((X,Y),X\times Y)}{2\epsilon(n)} + c$ rounds, where c is a positive constant. We prove this bound in the full version of the paper. Here we prove the second bound in the above theorem. Our proof is a natural generalization (and perhaps a slight simplification/clarification) of Cleve's proof for fair coin-tossing.

Proof of (2)**:** Consider a weakly ϵ-secure protocol π for secure sampling that runs in $r(n)$ rounds. In a single round, Alice sends a message to Bob followed by Bob sending a message to Alice. Recall the definitions of X_i and Y_{i-1} for $1 \leq i \leq r(n) + 1$ from Section 2. Since we are working in the plain model here (without any oracle set-up), Alice (resp. Bob) can compute the value of X_i (resp. Y_i) before sending the message for round i.

For simplicity in the following, we omit the security parameter n. We also assume that Alice and Bob always output a value from $\mathcal{X}$ and $\mathcal{Y}$ respectively when the other party aborts (for more discussion, see the full version). Fix a pair $(x, y) \in \mathcal{X} \times \mathcal{Y}$, and define four adversaries as shown in Figure 1 for each $1 \leq i \leq r$.

Let us first consider the probability that Bob outputs y when Alice is corrupted by $\mathcal{A}_{i,x}$ in the real world.

$$\begin{aligned}\mathsf{Pr}[\text{OUT}_{\pi,\mathcal{A}_{i,x}} = y] &= \mathsf{Pr}[X_i = x \wedge Y_i = y] + \mathsf{Pr}[X_i \neq x \wedge Y_{i-1} = y] \\ &= \mathsf{Pr}[X_i = x \wedge Y_i = y] - \mathsf{Pr}[X_i = x \wedge Y_{i-1} = y] + \mathsf{Pr}[Y_{i-1} = y].\end{aligned} \tag{3}$$

When $\mathcal{A}_i$ corrupts Alice, $\mathsf{Pr}[\text{OUT}_{\pi,\mathcal{A}_i} = y]$ is simply $\mathsf{Pr}[Y_{i-1} = y]$.

On the other hand, since the sampling functionality is inputless, no matter what strategy the adversary adopts in the ideal world, the output of Bob is distributed according to the marginal distribution Y. Therefore, for all $\mathcal{S}_{\mathcal{A}_{i,x}}, \mathcal{S}_{\mathcal{A}_i}$,

Adversary $\mathcal{A}_{i,x}$:	Adversary $\mathcal{B}_{i,y}$

Simulate Alice for $i-1$ rounds
if $X_i = x$ **then**
abort at round $i+1$
else
abort at round i
end if
	Simulate Bob for $i-1$ rounds
	if $Y_i = y$ **then**
	abort at round $i+1$
	else
	abort at round i
	end if

Adversary $\mathcal{A}_i$	Adversary $\mathcal{B}_i$
Simulate Alice for $i-1$ rounds Abort	Simulate Bob for $i-1$ rounds Abort

Fig. 1. Adversaries for round i, where $1 \leq i \leq r$.

we have

$$\mathsf{Pr}[\mathrm{OUT}_{\mathcal{F}_{\mathrm{SS}},\mathcal{S}_{\mathcal{A}_{i,x}}} = y] = \mathsf{Pr}[\mathrm{OUT}_{\mathcal{F}_{\mathrm{SS}},\mathcal{S}_{\mathcal{A}_i}} = y] = \mathsf{Pr}[Y = y], \tag{4}$$

where $\mathcal{S}_{\mathcal{A}}$ denotes the ideal world counterpart of a real world adversary $\mathcal{A}$. Hence,

$$\Delta\Big(\mathrm{OUT}_{\pi,\mathcal{A}_{i,x}}, \mathrm{OUT}_{\mathcal{F}_{\mathrm{SS}},\mathcal{S}_{\mathcal{A}_{i,x}}}\Big) + \Delta\big(\mathrm{OUT}_{\pi,\mathcal{A}_i}, \mathrm{OUT}_{\mathcal{F}_{\mathrm{SS}},\mathcal{S}_{\mathcal{A}_i}}\big) \geq \left|\mathsf{Pr}[X_i = x \wedge Y_i = y] - \mathsf{Pr}[X_i = x \wedge Y_{i-1} = y]\right|. \tag{5}$$

Similarly, by considering the real and ideal world outputs of Alice when Bob is corrupted by adversaries $\mathcal{B}_{i,y}$ and $\mathcal{B}_i$, for all $\mathcal{S}_{\mathcal{B}_i}, \mathcal{S}_{\mathcal{B}_{i,y}}$, we have

$$\Delta\Big(\mathrm{OUT}_{\pi,\mathcal{B}_{i,y}}, \mathrm{OUT}_{\mathcal{F}_{\mathrm{SS}},\mathcal{S}_{\mathcal{B}_{i,y}}}\Big) + \Delta\big(\mathrm{OUT}_{\pi,\mathcal{B}_i}, \mathrm{OUT}_{\mathcal{F}_{\mathrm{SS}},\mathcal{S}_{\mathcal{B}_i}}\big) \geq \left|\mathsf{Pr}[Y_i = y \wedge X_{i+1} = x] - \mathsf{Pr}[Y_i = y \wedge X_i = x]\right|. \tag{6}$$

Adding (5) and (6) for all $1 \leq i \leq r$, we have that the sum of $4r$ statistical difference terms is at least $|\mathsf{Pr}[X_{r+1} = x \wedge Y_r = y] - \mathsf{Pr}[X_1 = x \wedge Y_0 = y]|$. Hence, there exists an adversary $\mathcal{A}$ (among the $4r$ adversaries) such that for any ideal world adversary $\mathcal{S}$,

$$\Delta(\mathrm{OUT}_{\pi,\mathcal{A}}, \mathrm{OUT}_{\mathcal{F}_{\mathrm{SS}},\mathcal{S}}) \geq \frac{|\mathsf{Pr}[X_{r+1} = x \wedge Y_r = y] - \mathsf{Pr}[X_1 = x \wedge Y_0 = y]|}{4r} \tag{7}$$

We want to lower bound the above quantity in terms of X and Y. To this end, observe that when neither party is corrupt, the joint distribution of Alice and Bob's outputs in the ideal world is given by (X, Y), and in the real world it is given by (X_{r+1}, Y_r). Then, the ϵ-security of π implies that

$$|\mathsf{Pr}[(X, Y) = (x, y)] - \mathsf{Pr}[(X_{r+1}, Y_r) = (x, y)]| \leq \epsilon. \tag{8}$$

We can also obtain the following from the ϵ-security of π:

$$|\mathsf{Pr}[(X_1, Y_0) = (x, y)] - \mathsf{Pr}[X \times Y = (x, y)]| \leq 2\epsilon \tag{9}$$

(see the full version for a proof). Combining (7) with (8) and (9), we get:

$$\Delta(\mathrm{OUT}_{\pi,\mathcal{A}}, \mathrm{OUT}_{\mathcal{F}_{\mathrm{SS}},\mathcal{S}}) \geq \frac{|\mathsf{Pr}[(X, Y) = (x, y)] - \mathsf{Pr}[X \times Y = (x, y)]| - 3\epsilon}{4r}. \tag{10}$$

However, since π is an ϵ-secure protocol, the above quantity can be at most ϵ. Choosing a pair (x, y) that maximizes $|\mathsf{Pr}[(X, Y) = (x, y)] - \mathsf{Pr}[X \times Y = (x, y)]|$, we obtain the desired bound:

$$r \geq \frac{\alpha_{XY} - 3\epsilon}{4\epsilon}. \tag{11}$$

Acknowledgments. We thank Vinod Prabhakaran for pointing out that Theorem 2 was proven in [23], and for pointing us to [19]. We also thank the anonymous referees for helpful suggestions and pointers.

References

1. Asharov, G., Lindell, Y., Rabin, T.: A full characterization of functions that imply fair coin tossing and ramifications to fairness. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 243–262. Springer, Heidelberg (2013)
2. Beimel, A., Lindell, Y., Omri, E., Orlov, I.: $1/p$-secure multiparty computation without honest majority and the best of both worlds. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 277–296. Springer, Heidelberg (2011)
3. Beimel, A., Omri, E., Orlov, I.: Protocols for multiparty coin toss with dishonest majority. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 538–557. Springer, Heidelberg (2010)
4. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067 (2005) (revised version of [?])
5. Cheeger, J.: A lower bound for the smallest eigenvalue of the laplacian. Problems in analysis 625, 195–199 (1970)
6. Chung, F.R.K.: Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92). American Mathematical Society (1996)
7. Chung, F.R.K., Tetali, P.: Isoperimetric inequalities for cartesian products of graphs. Combinatorics, Probability & Computing 7(2), 141–148 (1998)
8. Cleve, R.: Limits on the security of coin flips when half the processors are faulty (extended abstract). In: Hartmanis, J. (ed.) STOC, pp. 364–369. ACM (1986)
9. Cleve, R., Impagliazzo, R.: Martingales, collective coin flipping and discrete control processes (1993) (manuscript), http://www.cpsc.ucalgary.ca/~cleve/pubs/martingales.ps
10. Even, S., Yacobi, Y.: Relations among public key signature systems. Technical Report 175, Technion, Haifa, Israel (1980)
11. Gács, P., Körner, J.: Common information is far less than mutual information. Problems of Control and Information Theory 2(2), 149–162 (1973)

12. Gebelein, H.: Das statistische problem der korrelation als variations und eigenwertproblem und sein zusammenhang mit der ausgleichungsrechnung. Zeitschrift für angew. Math. und Mech. 21, 364–379 (1941)
13. Gordon, S.D., Katz, J.: Partial fairness in secure two-party computation. Journal of Cryptology 25(1), 14–40 (2012); Preliminary version in Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 157–176. Springer, Heidelberg (2010)
14. Gordon, S.D., Hazay, C., Katz, J., Lindell, Y.: Complete fairness in secure two-party computation. J. ACM 58(6), 24:1–24:37 (2011); Preliminary version in STOC 2008
15. Gordon, D., Ishai, Y., Moran, T., Ostrovsky, R., Sahai, A.: On complete primitives for fairness. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 91–108. Springer, Heidelberg (2010)
16. Gray, R.M., Wyner, A.D.: Source coding for a simple network. Bell System Technical Journal 53, 1681–1721 (1974)
17. Ishai, Y., Ostrovsky, R., Seyalioglu, H.: Identifying cheaters without an honest majority. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 21–38. Springer, Heidelberg (2012)
18. Kilian, J.: More general completeness theorems for secure two-party computation. In: Yao, F.F., Luks, E.M. (eds.) STOC, pp. 316–324. ACM (2000)
19. Kumar, G.: Binary renyi correlation: A simpler proof of Witsenhausen's result and a tighter upper bound (2010) (manuscript), http://www.stanford.edu/~gowthamr/research/binary_renyi_correlation.pdf
20. Maji, H.K., Ouppaphan, P., Prabhakaran, M., Rosulek, M.: Exploring the limits of common coins using frontier analysis of protocols. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 486–503. Springer, Heidelberg (2011)
21. Moran, T., Naor, M., Segev, G.: An optimally fair coin toss. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 1–18. Springer, Heidelberg (2009)
22. Rényi, A.: On measures of dependence. Acta Math. Hung. 10, 441–451 (1959)
23. Witsenhausen, H.S.: On sequences of pairs of dependent random variables. SIAM Journal of Applied Mathematics 28, 100–113 (1975)
24. Wyner, A.D.: The common information of two dependent random variables. IEEE Transactions on Information Theory 21(2), 163–179 (1975)

A A Spectral Graph Theoretic Proof of Theorem 2

Firstly, we define a natural notion of bipartite graph product, to capture the bipartite characteristic graph resulting from multiple independent samples from a 2-party distribution.

Definition 2. *If $G_1 = (U_1, V_1, w_1)$ and $G_2 = (U_2, V_2, w_2)$ are two weighted bipartite graphs, we define their* bipartite tensor product $G_1 \boxtimes G_2 = (U, V, w)$ *as a weighted bipartite graph with $U = U_1 \times U_2$, $V = V_1 \times V_2$ and $w((u_1, u_2), (v_1, v_2)) = w_1(u_1, v_1) \cdot w_2(u_2, v_2)$.*

Also, for all positive integers k we define $G^{\boxtimes k} = G^{\boxtimes k-1} \boxtimes G$, where $G^{\boxtimes 0} = K_{1,1}$ (a single edge with weight 1).

To prove Theorem 2, we shall see that it suffices to lowerbound the "Cheeger constant" of $G^{\boxtimes k}$ (for all $k \in \mathbb{N}$). Before defining the Cheeger constant, we note

that we can consider a weighted bipartite graph $G = (U, V, w)$ as a general (not necessarily bipartite) weighted graph $G' = (T, w)$, where $T = U \cup V$, by extending its weight function (originally defined over $U \times V$) to cover all pairs of nodes in the graph, in a natural way: for $(v, u) \in V \times U$, $w(v, u) = w(u, v)$; for $(x, x') \in U^2 \cup V^2$, $w(x, x') = 0$. Also, as a matter of convenient notation, for every node $x \in T$, we define $w(x) = \sum_{y \in T} w(x, y)$. Also, for $S \subseteq T$, let $w(S) = \sum_{x \in S} w(x)$ and $w(S, \overline{S}) = \sum_{(x,y) \in S \times \overline{S}} w(x, y)$.

Definition 3 (Cheeger Constant). *For a weighted graph $G = (T, w)$, the Cheeger constant $h(G)$ is*

$$h(G) = \min_{S \subseteq T} \frac{w(S, \overline{S})}{\min(w(S), w(\overline{S}))}. \tag{12}$$

Lemma 1. *Given a weighted bipartite graph G, for all non-negative integers k, t, $h(G^{\boxtimes k} \boxtimes {K_{2,2}}^{\boxtimes t}) \geq \frac{1}{2} h^2(G)$, where $K_{2,2}$ denotes the complete bipartite graph with weight $\frac{1}{4}$ on all four edges.*

While this does not give very tight bounds on $h(G^{\boxtimes k})$ in terms of $h(G)$, the proof of this lemma, given in the full version, in fact shows that the second eigenvalue of the normalized Laplacian matrix associated with $G^{\boxtimes k} \boxtimes {K_{2,2}}^{\boxtimes t}$ (for $k > 0$) is *equal to* that of the normalized Laplacian matrix associated with G. This translates to the above bound via Cheeger inequality [5].

Now we sketch how this lemma can be used to prove Theorem 2. Let $G = (U, V, w)$ be the characteristic bipartite graph of p_{UV}. $G^{\boxtimes k} = (U^k, V^k, w^{(k)})$ denotes the graph corresponding to k independent samples from p_{UV}, where $w^{(k)}((u_1, \cdots, u_k), (v_1, \cdots, v_k)) = \prod_{i=1}^{k} w(u_i, v_i)$. Alice gets a node in U^k as her part of the sample from p_{UV}^k (i.e., k independent samples from p_{UV}), and Bob gets a node in V^k. Further, Alice and Bob may use private random coins, say t of them. The characteristic bipartite graph for t pairs of independent coins is ${K_{2,2}}^{\boxtimes t}$. Thus $G^{\boxtimes k} \boxtimes {K_{2,2}}^{\boxtimes t}$ denotes the entire view of the two parties in the protocol. Now, the output of each party is a deterministic function of its view.

W.l.o.g., we assume that each party is outputting a single bit (if necessary, by partitioning the outputs into two appropriately chosen parts, while retaining a constant amount of entropy in the outputs). Let $A_0 \subseteq U^k \times \{0, 1\}^t$ be the set of views on which Alice outputs 0. Similarly define A_1, and also define the sets B_0 and B_1 for Bob. Let w^* be the weight function for $G^{\boxtimes k} \boxtimes {K_{2,2}}^{\boxtimes t}$.

Then, the probability that Alice outputs 0 is $p_0^A = \sum_{a \in A_0} w^*(a)$. Similarly $p_1^A = \sum_{a \in A_1} w^*(a)$, and $p_0^B = \sum_{b \in B_0} w^*(b)$ and $p_1^B = \sum_{b \in B_1} w^*(b)$. W.l.o.g, assume that $p_0^A + p_0^B \leq p_1^A + p_1^B$ (interchanging 0 and 1 if necessary). Then, let $S = A_0 \cup B_0$ and $\overline{S} = A_1 \cup B_1$. Since we required the output of at least one party to have constant entropy, it must be the case that $p_0^A + p_0^B \geq \alpha$ for some constant $\alpha > 0$. The probability that Alice and Bob disagree on their outputs, p^* is given

by the weight of the edges that go across S and $\overline{S}$: i.e., $p^* = \sum_{x \in S, y \in \overline{S}} w^*(x, y)$. By definition of the Cheeger constant, we have

$$h(G^{\boxtimes k} \boxtimes {K_{2,2}}^{\boxtimes t}) \leq \frac{\sum_{x \in S, y \in \overline{S}} w^*(x, y)}{p_0^A + p_0^B} \leq \frac{p^*}{\alpha}.$$

That is, $p^* \geq \alpha h(G^{\boxtimes k} \boxtimes {K_{2,2}}^{\boxtimes t}) \geq \alpha \frac{h^2(G)}{2}$. Since p_{UV} has zero common information, its bipartite characteristic graph G has a single connected component, and $h(G)$ is positive. Thus, we can set $\epsilon = \alpha \frac{h^2(G)}{2}$ to complete the proof of Theorem 2.

Remark. Gács and Körner [11] also considered the case when the setup distribution has non-zero common information. Our proof readily extends to this setting, showing that the entropy of a common output *conditioned on this common information* will have to be $o(1)$ (when the disagreement probability is required to be $o(1)$).

Limits on the Power of Cryptographic Cheap Talk*

Pavel Hubáček[1,**], Jesper Buus Nielsen[1], and Alon Rosen[2,***]

[1] Aarhus University
[2] IDC Herzliya

Abstract. We revisit the question of whether cryptographic protocols can replace correlated equilibria mediators in two-player strategic games. This problem was first addressed by Dodis, Halevi and Rabin (CRYPTO 2000), who suggested replacing the mediator with a secure protocol and proved that their solution is stable in the Nash equilibrium (NE) sense, provided that the players are computationally bounded.

We show that there exist two-player games for which no cryptographic protocol can implement the mediator in a sequentially rational way; that is, without introducing empty threats. This explains why all solutions so far were either sequentially unstable, or were restricted to a limited class of correlated equilibria (specifically, those that do not dominate any NE, and hence playing them does not offer a clear advantage over playing any NE).

In the context of computational NE, we classify necessary and sufficient cryptographic assumptions for implementing a mediator that allows to achieve a given utility profile of a correlated equilibrium. The picture that emerges is somewhat different than the one arising in semi-honest secure two-party computation. Specifically, while in the latter case every functionality is either "complete" (i.e., implies Oblivious Transfer) or "trivial" (i.e., can be securely computed unconditionally), in the former there exist some "intermediate" utility profiles whose implementation is equivalent to the existence of one-way functions.

1 Introduction

The field of game theory offers a variety of ways to reason about the behavior of rational players. One of the most famous analytic tools for that purpose is that of Nash equilibrium [14]. In the basic case of two-player games, a Nash equilibrium (NE) constitutes of two independent plans of action, one for each player, such that no player can unilaterally benefit by deviating from her own plan. The NE solution concept was subsequently generalized by Aumann [2], who allowed players to pick their actions in a correlated way. Correlated equilibria (CE) are in many cases preferable over NE, in part because they can potentially guarantee higher utility to the players. In order to be able to act in

* Supported by the European Research Commission Starting Grant 279447, ISF grant no. 334/08, and Danish National Research Foundation and The National Science Foundation of China (grant 61061130540) for the Sino-Danish Center for the Theory of Interactive Computation.

** The author did part of this work while visiting IDC Herzliya supported by the ISF grant no. 334/08.

*** Supported by the ISF grant no. 334/08.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 277–297, 2013.

a correlated manner, the players are assumed to have access to a mediator (sometimes referred to as correlation device), that provides them with private, correlated, recommendations on the action to be taken.

About a decade and a half ago, Dodis, Halevi and Rabin [7] pointed out the possibility of implementing the mediator without having to refer to any trusted party. To this end, they proposed the use of secure two-party computation, viewing the correlation device as a randomized functionality. Their approach, natural from the cryptographic perspective, gives rise to some game theoretical challenges that need to be addressed. Most notably, the cryptographic protocol preceding the actual play of the strategic game introduces new actions that are observable by the players. Since these actions take place sequentially, the model of the game needs to be adjusted to account for the strategic decisions that players need to take during the protocol execution. While these actions do not directly affect the utility in the underlying strategic game, they can nevertheless influence the players in their decision making. Such pre-play communication is referred to as *cheap talk* in the economic literature.

One crucial difference from the mediated setting, which is inherited from the sequential nature of protocols, is that one of the players may learn her recommendation before the other. If this player is not happy with the protocol's recommendation, she can simply decide to "abort," thus preventing the other player from learning his own recommendation. Another crucial difference is that player A (not necessarily the one who learns her recommendation first), can reveal extra information to player B, changing player B's knowledge and expectation on how player A is going to play.[1]

Given that such deviations can always be observed, it becomes necessary to specify what action players take in case deviation is detected. One could attempt to deter misbehavior by threatening with some punishment. However, it is not *a priori* clear what kind of punishment should a player invoke, assuming that the other player is rational. In the protocol of Dodis *et al.* [7], an "abort" action is punished by employing the min-max strategy (that is, the strategy that minimizes the maximal gain of the deviator). This approach suffers from the well known and often unavoidable shortcoming of being harmful to the punishing player. Consequently, the *threat* of playing the min-max strategy is *empty*, or in other words not credible. Punishing the other type of deviations, in which the deviating player reveals extra information, appears to be even more challenging, as a message reacting to such deviations might not even fall into the scope of the prescribed protocol (for instance, if the deviating player is the last to learn her recommendation, meaning that the protocol actually terminates at that point).

The issue of empty threats is classically handled by the requirement of subgame perfection (SPE), which requires strategies to be in equilibrium at any point during the protocol execution. This requirement insures that any threat is credible. One problem with subgame perfection, that is particularly acute when modeling behavior of computationally bounded players in a cryptographic protocol, is the requirement of optimality at any point in the protocol execution. This problem was first addressed by Gradwohl, Livne and Rosen [9], who by defining empty threats in an explicit manner, were able

[1] For instance, the second player to learn his own recommendation could make his private view of the protocol public, thus revealing his recommendation to the first player and rendering the correlation device useless.

to reason about sequential rationality in face of computationally bounded players. In addition to this modeling, their work proposed a simple cryptographic protocol for the class of convex hull Nash equilibria (i.e., correlated equilibria that can be expressed as a convex combination of the Nash equilibria of the game), assuming the existence of one-way functions. To avoid empty threats, their solution punishes the aborting player with her "worst" NE (i.e., the NE yielding the lowest payoff amongst all NE in the game). Indeed, since the punishment is a Nash equilibrium, a rational punishing player has no incentive to deviate from it, which renders the threat of playing this NE credible.

One significant shortcoming of the Gradwohl *et al.* [9] solution is that it only applies to convex combinations of Nash equilibria. Unfortunately, such equilibria are not very interesting since they do not enjoy the most beneficial feature of CE, namely the ability of dominating the payoffs achieved by any NE. This leaves open the question of whether there exists a sequentially rational cryptographic protocol for implementing the mediator in the cases where playing a CE is preferable over playing any NE.

1.1 Our Results

A necessary requirement for guaranteeing sequential rationality is the ability for a player to threaten credibly. For this to be possible the threat must consist of a rational plan of action. Otherwise, there is no guarantee that a rational player will follow through in case she is tested. We formalize this intuition by putting forward the notion of *Nash equilibrium punishable CE*. These are correlated equilibria for which the expected utility of any player given a recommendation by the mediator is never smaller than in her worst NE. This notion turns out to be crucial for implementing the mediator of a CE using a cryptographic protocol.

Theorem 2 (Informal). *A correlated equilibrium can be implemented in a sequentially rational way using cryptographic cheap talk if and only if it is Nash equilibrium punishable.*

Given the above theorem, it is natural to ask whether every CE is NE-punishable. An affirmative answer would have implied that any player receiving an unsatisfactory recommendation from the cryptographic protocol can be threatened from aborting in a credible way.

Our answer to this question is negative. We show that there exist games with CE that are not NE-punishable. Moreover, these games have utility profiles that can be obtained only by those CE that are not NE-punishable (and so cannot be achieved by other NE-punishable equilibria). Additionally, both players prefer these utility profiles to utility profile of some other NE-punishable CE, thus both would be in favor of implementing such *preferable* CE.

Theorem 1 (Informal). *There exist infinitely many strategic games with preferable CE that cannot be achieved by sequentially rational cryptographic cheap talk.*

The above theorem explains why all solutions so far were either sequentially unstable, or were restricted to a limited class of correlated equilibria.

In addition to the above results, we classify necessary and sufficient cryptographic assumptions for implementing a mediator that allows to achieve a given utility profile

of a CE by a protocol that is in computational NE. We show that there are non-trivial CE in the convex hull of Nash equilibria[2] (CHNE) which can be implemented via cheap talk only if one-way functions exist.

Theorem 3 (Informal). *If the payoff of all non-trivial convex hull Nash equilibria can be achieved via cryptographic cheap talk then one-way functions exist.*

As shown by Gradwohl *et al.* [9], if one-way functions exist then all non-trivial CE in the convex hull of NE can be implemented via computational (and moreover sequentially rational) cheap talk. Taken together these results fully characterize the assumptions under which all convex hull NE can be implemented. We also show that there exist CE outside CHNE which can only be cheap talk implemented if OT exists.

Theorem 4 (Informal). *If the payoff of all correlated equilibria outside the convex hull of NE can be achieved via cryptographic cheap talk then there exists a protocol for oblivious transfer (OT).*

As shown by Dodis *et al.* [7], if there exists a protocol for OT then all correlated equilibria (including those outside the convex hull of NE) can be implemented via computational (but not necessarily sequentially rational) cheap talk. Taken together these results show that OT is complete for implementing all CE (regardless of the issue of sequential rationality). We conjecture that implementing *any* CE outside the CHNE and provide evidence to support the conjecture. We leave it as an open problem to prove or disprove the conjecture.

These are to our best knowledge the first results of this type. Previous work on rational cryptography has focused on sufficiency of cryptography for implementing equilibria. Our results suggest an intriguing connection between the distinction between CE and CHNE on one hand and the distinction between Cryptomania and Minicrypt on the other hand (see Impagliazzo [11]). The picture that emerges is somewhat different than the one arising in semi-honest secure two-party computation. While in the latter case every functionality is either “complete" (i.e. implies OT) or “trivial" (i.e. can be securely computed unconditionally), in the former there exist some “intermediate" utility profiles whose implementation is equivalent to the existence of one-way functions. The details are given in Sect. 6 and Sect. 7.

1.2 Related Work

Osborne and Rubinstein [15] provide a standard introduction to game theory. The notion of correlated equilibrium was introduced by Aumann [2]. A non-technical introduction motivating the notion of cheap talk is given in Farrell and Rabin [8]. Cheap talk implementation of a correlation device in game-theoretical framework was put forward by Bárány [4]. Aumann and Hart [3] show what equilibria payoffs can be achieved via cheap talk preceding games with imperfect information.

We already mentioned the works in [7, 9]. Teague [17], and subsequently Atallah *et al.* [1] gave a protocol for the general problem of correlated element selection achieving

[2] Note that NE, even though contained in the convex hull of NE, are trivial from our perspective, since there is no need for a mediator to play according to them.

better efficiency than [7], but preserving the original solution concept of computational NE. Using results from computational complexity to implement correlation devices was considered by Urbano and Vila [19], aiming for a similar result to Dodis *et al.* [7]. However, Teague [18] showed that their approach is flawed. An alternative solution concept for analyzing game theoretical properties of cryptographic protocols was suggested by Pass and shelat [16].

2 Preliminaries and Definitions

For $m \in \mathbb{N}$, we use $[m]$ to denote the set $\{1,\ldots,m\}$. For a finite set A, we use $\Delta(A)$ to denote the set of probability distributions over A.

Definition 1 (Two-player Strategic Game). *A* two-player strategic game Γ *is a triple* (A_1,A_2,u)*, where* A_i *is a set of actions of player* $i \in \{1,2\}$*, and* $u : A_1 \times A_2 \to \mathbb{R}^2$ *is a utility function assigning a utility profile to every action profile* $a \in A_1 \times A_2$*. We use* u_i *to denote the i'th output of u, i.e.,* $u(a) = (u_1(a),u_2(a))$*.*

In this work we only consider two-player games. Also, we talk about a $k \times k$ strategic game Γ if both players have k strategies in Γ, i.e., $|A_1| = |A_2| = k$. A classical example of strategic game is the game of Chicken as in Fig. 1a.

Definition 2 (Strategy Profile). *A* strategy profile *for a strategic game* Γ *is a probability distribution* γ *on* $A_1 \times A_2$*, i.e.,* $\gamma \in \Delta(A_1 \times A_2)$*. We denote* $\gamma(a)$ *the probability assigned by* γ *to* $a \in A_1 \times A_2$*. The corresponding utility profile* $U(\gamma) \in \mathbb{R}^2$ *is given by* $U(\gamma) = (U_1(\gamma),U_2(\gamma))$*, where* $U_i(\gamma) = \sum_{(a_1,a_2)\in A_1\times A_2} \gamma(a_1,a_2)u_i(a_1,a_2)$ *for* $i \in \{1,2\}$*. If* $U(\gamma) = (v_1,v_2)$*, we say that* γ achieves the utility profile (v_1,v_2)*.*

Definition 3 (Correlated Equilibrium). *A* correlated equilibrium (CE) *of a strategic game* (A_1,A_2,u) *is a strategy profile* $\gamma \in \Delta(A_1 \times A_2)$*, such that for every player* $i \in \{1,2\}$ *and every pair of strategies* $a_i, a_i' \in A_i$ *it holds that*

$$\sum_{a_{-i}\in A_{-i}} \gamma(a_i,a_{-i})u_i(a_i',a_{-i}) \leq \sum_{a_{-i}\in A_{-i}} \gamma(a_i,a_{-i})u_i(a_i,a_{-i}) \ .$$

We denote $U_i(\gamma|a_i)$ the expected utility of player i when given advice $a_i \in A_i$ and the other player also plays according to some advice sampled from γ, i.e., $U_i(\gamma|a_i) = \left(\sum_{a_{-i}\in A_{-i}} \gamma(a_i,a_{-i})\right)^{-1} \sum_{a_{-i}\in A_{-i}} \gamma(a_i,a_{-i})u_i(a_i,a_{-i})$.

Definition 4 ((Convex Hull) Nash Equilibrium). *A* Nash equilibrium (NE) *of a strategic game* $\Gamma = (A_1,A_2,u)$ *is a correlated equilibrium* σ *of* Γ*, such that* σ *is also a product distribution, i.e.,* $\sigma \in \Delta(A_1) \times \Delta(A_2)$*. A* convex hull Nash equilibrium (CHNE) *of a strategic game* Γ *is a correlated equilibrium of* Γ *that can be expressed as a convex combination of Nash equilibria of* Γ*.*

We denote $\mathrm{NE}(\Gamma), \mathrm{CHNE}(\Gamma)$, and $\mathrm{CE}(\Gamma)$ the set of Nash equilibria of Γ, the set of convex hull Nash equilibria of Γ, and the set of correlated equilibria of Γ respectively.[3]

[3] As a convention, we will use γ to denote a strategy profile that is a CE and σ to denote a strategy profile that is a NE.

We are interested in implementing correlated equilibria of two-player strategic games. Given such strategic game Γ one can visualize the utility profiles achievable by all its correlated equilibria in $\mathbb{R}^2$. Figure 1b depicts the polygon of utility profiles achievable by CE of the game of Chicken defined by the payoff matrix in Fig. 1a. The dark grey triangle corresponds to utility profiles achievable by the CHNE of Chicken, and its three corner points are exactly the payoffs of the three NE of the game of Chicken. One can see that the payoffs of CE of Chicken extend the region of CHNE payoffs in both directions, i.e., there are both CE that improve the CHNE payoffs (the white polygon) and those that are dominated by the CHNE payoffs (the light grey triangle).

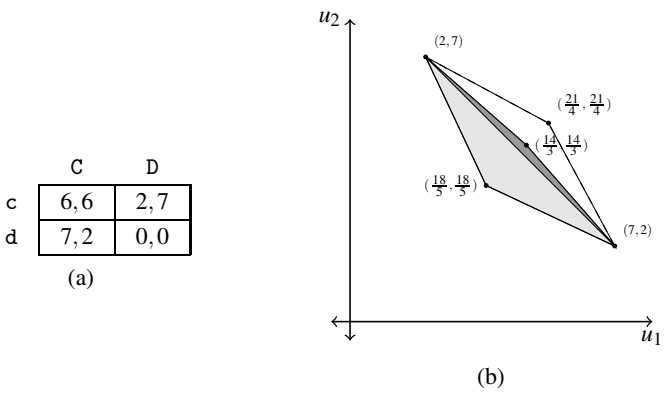


Fig. 1. (a) the game of Chicken (b) the utility profiles achievable by its CE

There is a natural partial ordering on the utility profiles induced by the relation of Pareto dominance.

Definition 5 ((Strict) Pareto Dominance, Weak Pareto Optimality). *Let Γ be a strategic game, and $\gamma, \gamma' \in \mathrm{CE}(\Gamma)$. If $U_i(\gamma) > U_i(\gamma')$ for both $i \in \{1,2\}$, we say that γ* strictly Pareto dominates *γ'. We say that γ* Pareto dominates *γ' if for both $i \in \{1,2\}$ it holds that $U_i(\gamma) \geq U_i(\gamma')$, and there exist $i' \in \{1,2\}$ such that $U_{i'}(\gamma) > U_{i'}(\gamma')$. We say that a $\gamma^* \in \mathrm{CE}(\Gamma)$ is* weakly Pareto optimal *if there exists no $\gamma' \in \mathrm{CE}(\Gamma)$ that Pareto dominates γ^*.*

We sometimes abuse the notation and say that utility profile $v \in \mathbb{R}^2$ (strictly) Pareto dominates $v' \in \mathbb{R}^2$ if there exist $\gamma, \gamma' \in \mathrm{CE}(\Gamma)$, such that $v = U(\gamma), v' = U(\gamma')$ and γ (strictly) Pareto dominates γ'. Consider again the CE payoffs of Chicken in Fig. 1b. The two line segments between $(2,7)$ and $(\frac{14}{3},\frac{14}{3})$, and between $(\frac{14}{3},\frac{14}{3})$ and $(7,2)$ on the boundary of CHNE payoffs are exactly the *weakly Pareto optimal boundary* of the CHNE payoffs of Chicken.

3 Not All CE Are NE-Punishable

In this section, we show that there exists a barrier for using cryptography to implement any interesting correlated equilibrium without empty threats. Intuitively, for a correlated

equilibrium to be implementable by a cryptographic protocol without empty threats, one must be able to effectively punish any deviating player by her worst NE.

Definition 6 (NE-punishable CE). *Let γ be a CE of a strategic game $\Gamma = (A_1, A_2, u)$. We say that γ is a* Nash equilibrium punishable correlated equilibrium *if for all $i \in \{1,2\}$ and every action $a_i \in A_i$ of player i played with non-zero probability in γ it holds that $U_i(\gamma|a_i) \geq U_i(\sigma_i)$, where σ_i is the worst Nash equilibrium for i in Γ.*

It is not at all obvious if there exists any strategic game with a CE that is not NE-punishable; it could also be the case that for any CE there exists a NE-punishable CE achieving the same utility profile. However, we show that none of the above is true. There are in fact many games with correlated equilibria that have some utility profile extending the polygon of CHNE payoffs, but no NE-punishable CE achieves such utility profile.

Theorem 1. *For any $k \in \mathbb{N}$. If $k > 3$, then there exists a $k \times k$ strategic game Γ with a correlated equilibrium $\gamma \in \mathrm{CE}(\Gamma) \setminus \mathrm{CHNE}(\Gamma)$, s.t. every $\gamma' \in \mathrm{CE}(\Gamma)$ with $U(\gamma') = U(\gamma)$ is not a NE-punishable CE of Γ.*

The proof is constructive. We start with a suitable $(k-1) \times (k-1)$ strategic game Λ and extend it into a $k \times k$ game Γ that exemplifies the theorem; the initial game Λ is characterised by some non-trivial properties (given by the criterion in Def. 7) that are exploited when we extend it.

Definition 7 (Extensibility Criterion). *A strategic game $\Lambda = (A_1, A_2, u)$ satisfies the* extensibility criterion *if there exists γ, a CE of Λ, with the following two properties:*

1. *γ strictly Pareto dominates any NE of Λ.*
2. *There exists $a \in A_i$ for some player $i \in \{1,2\}$, such that for every $\gamma' \in \mathrm{CE}(\Gamma)$ with $U(\gamma') = U(\gamma)$ it holds that $U_i(\gamma') > U_i(\gamma'|a)$.*

We use the fact that any strategic game Λ satisfying the extensibility criterion has a CE γ preferable for both players to any NE of Λ. The CE γ is preserved as a correlated equilibrium in the extended game Γ. We are able to carefully devise the payoffs of Γ such that its unique NE is strictly Pareto dominated by γ, however for at least one of the players there exists a recommendation in γ that is inferior to the unique NE.

Lemma 1. *For any $k \in \mathbb{N}^+$, if there exists a $(k-1) \times (k-1)$ strategic game Λ_{k-1} that satisfies the extensibility criterion, then there exists a $k \times k$ strategic game Γ with a correlated equilibrium $\gamma \in \mathrm{CE}(\Gamma) \setminus \mathrm{CHNE}(\Gamma)$, s.t. every $\gamma' \in \mathrm{CE}(\Gamma)$ with $U(\gamma') = U(\gamma)$ is not a NE-punishable CE of Γ.*

Proof. We show how to extend $\Lambda_{k-1} = (A, B, u)$ with one additional action for each player to define Γ. Let a_0 be the new action of player A and b_0 be the new action of player B, thus $\Gamma = (A \cup \{a_0\}, B \cup \{b_0\}, u')$. The utility function u' of Γ corresponds to the utility function of Λ_{k-1} for every action profile in $A \times B$. For some $s, t \in \mathbb{R}$, u' is defined on the remaining action profiles as: $u'(a_0, b_0) = (t, t)$, and $u'(a_0, b) = u'(a, b_0) = (s, s)$ for every $b \in B$ and every $a \in A$.

We show that it is possible to select s and t such that the claim holds. Recall that Λ_{k-1} satisfies the extensibility criterion, so there exists a CE γ satisfying the two conditions from Def. 7. Let i be the player and $a \in A_i$ be the advice from the second condition of the extensibility criterion. Denote v the expectation of player i in γ given recommendation a, i.e., $v = U_i(\gamma|a)$. We can assume without loss of generality that γ is the CE with maximal v. Let v' be the maximal utility obtained in Λ_{k-1} by any of the players in some NE, i.e., $v' = \max(U_A(\sigma_A^*), U_B(\sigma_A^*))$, where σ_i^* is the best NE for player i. Set s such that $\max(v,v') < s < U_i(\gamma)$, and let $t = (s + U_i(\gamma))/2$.

If s and t are selected as above, then no Nash equilibrium of Λ_{k-1} is a Nash equilibrium of Γ. Moreover, the action profile (a_0, b_0) is a unique NE of Γ achieving the utility profile (t,t). However, γ is still a correlated equilibrium in Γ, and the expectation of player i when given a as a recommendation is strictly smaller than the utility obtained by player i in the unique NE (a_0, b_0) of Γ. Thus γ is not a NE-punishable CE.

Consider any other CE γ' of Γ that achieves the same utility profile as γ. Both t and s are smaller than $U_i(\gamma)$, thus any new correlated equilibrium achieving $U(\gamma)$ satisfies the second condition from the extensibility criterion. Since $U_i(\gamma|a) \geq U_i(\gamma'|a)$, any such γ' is also not NE-punishable. □

It remains to show that games satisfying the extensibility criterion exist for any $k > 2$.

Lemma 2. *For every $k \in \mathbb{N}$ with $k > 2$, there exists a $k \times k$ strategic game Λ_k that satisfies the extensibility criterion.*

Proof. Let $c,d,e,f,g \in \mathbb{R}$ be real numbers such that $c < d < e < f < g$, where $g - f < e - c$, and $3f < (e - c)$.[4] Consider the $k \times k$ game $\Lambda_k = (A = \{a_1, \ldots, a_k\}, B = \{b_1, \ldots, b_k\}, u)$ with the utility function $u : A \times B \to \mathbb{R}^2$ defined as follows:

- $u(a_j, b_j) = (f,g)$ for every $j \in [k-1]$,
- $u(a_k, b_k) = (d,e)$,
- $u(a_j, b_{j+1}) = (g,f)$ for every $j \in [k-2]$,
- $u(a_{k-1}, b_k) = (e,f)$,
- $u(a_k, b_1) = (g,d)$, and
- $u(a,b) = (c,c)$ otherwise.

To illustrate the corresponding payoff matrix, we give the payoff matrix of Λ_4 in Fig. 2.

	b_1	b_2	b_3	b_4
a_1	f,g	g,f	c,c	c,c
a_2	c,c	f,g	g,f	c,c
a_3	c,c	c,c	f,g	e,f
a_4	g,d	c,c	c,c	d,e

Fig. 2. The payoff matrix of Λ_4

[4] The two conditions $g - f < e - c$ and $3f < (e - c)$ are required for ease of exposition when describing the candidate CE. In fact, Λ_k defined without this conditions would also satisfy the claim of Lemma 2.

Due to the restrictions on the entries in the payoff matrix, there is no pure Nash equilibrium in Λ_k. Indeed, for every action profile $(a,b) \in A \times B$ there exists either an action a' of player A or an action b' of player B, such that A prefers (a',b) to (a,b) or B prefers (a,b') to (a,b). Following the same reasoning, Λ_k can only have fully mixed Nash equilibria. Notice that any of such NE assigns non-zero probability to the action profiles with utility profile (c,c).

We describe a candidate CE for the claim of Lemma 2. Let γ_k be any probability distribution on $A \times B$ satisfying these conditions.

1. $\gamma_k(a_k,b_1) = \gamma_k(a_{k-1},b_k) = \gamma_k(a_k,b_k) = \frac{g-f}{3(g-f)+(2k-3)(e-c)}$,
2. $\gamma_k(a,b) = \frac{e-c}{3(g-f)+(2k-3)(e-c)}$ for every $(a,b) \notin \{(a_k,b_1),(a_{k-1},b_k),(a_k,b_k)\}$ such that $u(a,b) \neq (c,c)$, and
3. $\gamma_k(a,b) = 0$ otherwise.

A proof of the following claim is given in the full version.

Claim. Any such probability distribution $\gamma_k \in \Delta(A \times B)$ is a correlated equilibrium of Λ_k.

Moreover, γ_k has in its support only the action profiles that do not yield the utility profile (c,c). Therefore, any such CE strictly Pareto dominates any completely mixed NE of Λ_k.

The expectation $U_A(\gamma_k)$ of player A is

$$((k-1)f + (k-2)g)\frac{e-c}{3(g-f)+(2k-3)(e-c)} + \frac{(d+e+g)(g-cf)}{3(g-f)+(2k-3)(e-c)},$$

and this is strictly larger than f when $3f < (e-c)$. On the other hand, any correlated equilibrium γ'_k of Λ_k that achieves the same utility profile as γ_k must assign non-zero probability to every action profile with utility profile different from (c,c). Since the highest utility of player A obtained from any action profile in which A plays action a_{k-1} is f, the expectation of A in any such correlated equilibrium γ'_k when given recommendation a_{k-1} is at most f. Therefore, Λ_k satisfies the extensibility criterion. □

4 Computational Cheap Talk Simultaneous Move Games

In this section we present an overview of our game theoretical model and solution concepts. Full details are given in the full version.

Our core object of study is so-called computational cheap talk, simultaneous move (CTSM) games. A CTSM game without types is fully *specified* by a strategic game (A_1,A_2,u). The game itself is an extensive game with imperfect information modeling an interactive protocol, where the agents take turn in exchanging messages, with agent 1 arbitrarily being chosen to send the first message. At some point each agent must additionally pick an action $a_i \in A_i$ for (A_1,A_2,u). The utility of a play is $u(a_1,a_2)$, i.e., the utility does not depend on the communication, only the actions. We assume that the agents do not get any information on what the action of the other party is,

and hence consider the choice of actions for (A_1, A_2, u) as simultaneous moves. The strategy σ_i of agent i specifies which messages to send in response to the messages sent by the other agent, and which action to pick for (A_1, A_2, u) at the end of the cheap talk. We require that σ_i is poly-time, to allow using cryptography. Any mixed strategy should also by poly-time computable. To conveniently model this, we technically only allow pure strategies, and then we give each such strategy an extra input r_i, which is a uniformly random bit-string not observed by the other agent. Any mixing must be implemented by $\sigma_i(r_i)$ in poly-time.

As described above, for each strategic game (A_1, A_2, u), we have a CTSM game. Correspondingly, for each CTSM game, we have a strategic game, which is just the game (A_1, A_2, u) used to specify it. We say that a CE for a strategic game can be *cheap talk implemented* if there exists a strategy $\sigma = (\sigma_1, \sigma_2)$ for the corresponding CTSM game which obtains the same utility profile as the CE and which is a *computational* NE, which is just an ε-NE for a negligible ε. We say that a CE for a strategic game can be *ETF cheap talk implemented* if it can be cheap talk implemented by some σ which is additionally *empty-threat free*. We define empty-threat freeness along the lines of [9], specialize their general definition to the setting of CTSM games and generalizing to handle imperfect information. The details are in the full version. Here we sketch and motivate the definition.

An *empty threat* posed by me in a CTSM game is a part of my future strategy which I do not currently play and which I would not play should you call my bluff by deviating in a way making the threatening strategy active. You would *demonstrate* the existence of such a future empty threat posed by me by specifying a deviation by you which would make me deviate from playing the supposedly empty threat. We adopt this constructive definition, an advantage being that we can insist that the demonstration be poly-time. Note, however, that using an empty threat to force me to deviate from a threat does not convincingly demonstrate that my threat was empty. We therefore require that your demonstrator itself is empty threat free in future play. Formally we require that the deviation meant to demonstrate the existence of a future empty threat occurs in response to some event D, for *deviate*, and require that the demonstration be empty-threat free in the sub-game defined by D occurring.

Another qualification is that a deviation which makes me abstain from my threat, but which does not at the same time result in you receiving a larger expected utility does not demonstrate that I posed an empty threat. Yes, your deviation made me not execute the threat, but the threat did not serve to prevent you from this particular deviation, as you have no incentive for your deviation in the first place. All in all, a credible demonstration that I am posing an empty threat on you would therefore be an event D observable by you, and a deviation, which you only make when D occurs, which has the property that it leads to an empty-threat free future play, in the sub-game defined by D occurring, in which you have higher utility.

Formalizing the above definition and making it work well with the computational issue, is highly non-trivial, but none of the details really matter for the intuition of the results we describe later. For details, see the full version. Here we only mention and motivate the two main technical choices.

Since our definition of ETF is recursive, we need a last round to start from. Yet, our strategies are allowed any polynomial number of rounds, and the nature of most settings naturally modeled by CTSM games does not make it seem reasonable to postulate some exogenous fixed last round of communication, so we do not want to build a fixed last round into our model. Also, it is by far always given that a party can commit to an external action, like a bid in a real-life auction, until long after the cheap talk protocol was run, so we cannot guarantee that no more communication can take place after the protocol was run. I.e., the natural strategy space contains the possibility of more communication than needed exactly by the protocol in question, so our model should capture this. We essentially handle this by considering CTSM games families of games, $\Gamma = \{\Gamma_R\}_{R\in\mathbb{N}}$, where all Γ_R have the same corresponding strategic game, and where Γ_R has a fixed last round in round R. This allows to easily define ETF for each Γ_R, and we then say that σ is ETF if there exists R_0 such that it is ETF for all Γ_R for $R \geq R_0$. I.e., the stability of a protocol is in particular not jeopardized by leaving some empty rounds after the execution of the strategy, i.e., rounds in which communication could have taken place. Robustness to the presence of such possible communication seems crucial for stability in real world networks.

We have chosen to use a similar mechanism to model poly-time. For a fixed strategic game (A_1,A_2,u) and $T \in \mathbb{N}$, let Γ^T be the CTSM game corresponding to (A_1,A_2,u), where the messages and the action must be computable in time exactly T. For a polynomial p we consider a family of games $\Gamma_p = \{\Gamma^{(\kappa)} = \Gamma^{p(\kappa)}\}_{\kappa\in\mathbb{N}}$. A strategy $\sigma = \{\sigma^{(\kappa)}\}_{\kappa\in\mathbb{N}}$ for Γ_p is one where $\sigma^{(\kappa)}$ is a strategy for $\Gamma^{(\kappa)}$. A strategy σ for Γ is clearly poly-time. We say that σ is a computational NE for Γ_p if there exists negligible ε such that $\sigma^{(\kappa)}$ is an $\varepsilon(\kappa)$-NE for $\Gamma^{(\kappa)}$. We call it a computational CTSM for (A_1,A_2,u) if there exists a polynomial p_0 such that it is a computational NE for Γ_p for all $p \geq p_0$. Using the same flavor of definition to handle the computational issue and the no-last-round issue, allows to give one natural definition handling both issues.

Note that the above two design choices force proposed protocols to run in some fixed polynomial number of rounds and some fixed poly-time, whereas deviations are allowed to deviate to larger polynomials. This seems natural and strong.

To play a NE of any strategic game it is sufficient for the players to randomize independently, and there is no need for any cheap talk. The players need some publicly observable lottery to play according to a CHNE, that can be implemented using the protocol of Gradwohl *et al.* [9]. However, a CE outside the convex hull of NE needs some non-trivially correlated randomness. Motivated by our results from Sect. 6 and Sect. 7, we categorize correlated equilibria payoffs using the terminology of Impagliazzo [11].

Definition 8 (Trivial, Minicrypt, and Cryptomania Utility Profiles). *Let Γ be a strategic game, and $v \in \mathbb{R}^2$ be a utility profile achieved by some $\gamma \in \mathrm{CE}(\Gamma)$.*

- *We call v a* trivial utility profile *if there exists $\sigma \in \mathrm{NE}(\Gamma)$ achieving v.*
- *We call v a* Minicrypt utility profile *if γ is a CHNE and there is no NE achieving v.*
- *We call v a* Cryptomania utility profile *if γ is not a CHNE.*

5 NE-Punishable CE versus Empty-Threat Free NE

We can now formally relate NE-punishable CE and empty-threat free computational NE.

Theorem 2. *Let $\Gamma = (A_1, A_2, u)$ be a strategic game and let $\tilde{\Gamma}$ be the corresponding CTSM game. If there exists a strategy profile σ, a computational ETFE of $\tilde{\Gamma}$ with utility profile (v_1, v_2), then there exists a NE-punishable CE γ for Γ achieving the same utility profile (v_1, v_2).*

The theorem is proven in the full version. Here we provide a sketch of the proof. Consider any computational ETFE σ of $\tilde{\Gamma}$. Remember that σ is a family of strategies, and the utility profile of the members of the family need not converge to a fixed utility profile. However, we assume in the premise of the theorem that it does converge, to some (v_1, v_2). In the same vain, the action profiles of the members need not converge. However, the distribution of the action profile of all the strategies, i.e., the probability distribution over which actions $(a_1, a_2) \in A_1 \times A_2$ they make the players play, belong to a fixed compact space as we consider finite games Γ. Hence we can pick an infinite sub-sequence which converges to some probability distribution γ on $A_1 \times A_2$. It is possible to show that γ is a CE. Namely, in the games of the convergent sub-sequence, the incentive to deviate given any particular action is converging to 0, as σ in particular is an ε-NE for a negligible ε. This means that the incentive to deviate in the limit point γ is 0, by compactness. For the same reason γ has utility profile (v_1, v_2). We now assume that γ is not NE-punishable, and use this to show that σ is not empty threat free, which proves the theorem by contradiction.

If γ is not NE-punishable, then there exist $i \in \{1, 2\}$ and an action $a_i \in A_i$ such that a_i occurs with non-zero probability and such that $U_i(\gamma|a_i) < U_i(\sigma_i^*)$, where σ_i^* is the worst NE for player i and $U_i(\gamma|a_i)$ is the expected utility of player i when playing γ given that the recommendation is a_i.

To prove that σ is not a computational ETFE we must pick a strategy space with enough rounds to run σ, or more rounds, and show that σ is not an ε-ETFE in this strategy space for any negligible ε. This in turn means that we must give an event D observable by P_2 (assume w.l.o.g. that $i = 2$) and a deviation for P_2 in the face of D for which he gets noticeably better expected utility in all ETF plays in the sub-game defined by D occurring.

As for the strategy space, pick the one which after the run of σ leaves at least one extra round of communication and where it is player 2 who sends a message in the last round of the strategy space. As the event D, pick the event that the output of running σ_2 is the bad action a_i for which $U_i(\gamma|a_i) < U_i(\sigma_i^*)$ and that κ is among the values in the infinite sub-sequence which converges to γ. As for the deviation, let player 2 play exactly as in σ_2, except that if D occurs, then player 2 does not play a_i. Instead, it waits until the last communication round where it sends its entire view of the protocol to player 1. Then player 2 picks an action a_2^* according to σ_2^*, and plays a_2^*. To show that σ is not a computational ETFE, it is now sufficient to show that in all ETF continuations after the last communication round, in the sub-game defined by D occurring, player 2 gets noticeably better expected utility than by playing σ. If this is not the case, then there exists an ETF continuation $\tilde{\sigma}$ after the last communication round, in the sub-game

defined by D occurring, such that player 2 gets utility close to what he gets by playing σ when D occurs, which in turn is lower than what he gets by playing the worst NE. It follows that the utility profile of $\tilde{\sigma}$ is not the utility profile of a CHNE. Namely, a CHNE has a utility profile which is a convex combination of utility profiles for NE, so no player can get less than in his worst NE.

To conclude the proof by contradiction it is now sufficient to prove that $\tilde{\sigma}$ is a CHNE. Recall that $\tilde{\sigma}$ is played in the sub-game with a common prior C corresponding to the view of the parties after D occurred. Since player 2 sends his entire view to player 1 when D occurs, in the common prior C, player 1 can efficiently compute the signal of player 2. Denote the signal of player i by s_i. We use that $s_2 = s(s_1)$ for a fixed poly-time function s. If we give unbounded computing time to player 1 and only give it the signal s_2, then it can re-sample a random $(s_1', s_2') \leftarrow C$ with $s(s_1') = s_2$ and play according to $\sigma_1(s_1')$. This will lead to exactly the same strategy, and the unbounded computing power of player 1 does not allow it better deviations: since player 1 can efficiently compute $s_2 = s(s_1)$ from s_1 and since it knows the code σ_2 of player 2, it can use random runs of $\sigma_2(s_2)$ to sample the strategy profile of player 2 up to exponentially good precision in poly-time and and then in poly-time compute an optimal response to this fixed and now known strategy. Hence the unbounded computing power can at most give inverse exponentially more utility, which does not disturb the ε-NE. But then we have an ε-NE where the players have a common signal s_2. It is possible to use compactness of the strategy space to show that a sub-sequence of an ε-NE converges to a CHNE. The details are in the full version.

It is instructive to see how the above *reveal your view* deviation defeats some of the obvious attempts at circumventing the impossibility result.

Consider first a relaxed version of NE-punishable, which we could call *one-sided punishable*, where we only require that there *exists* $i \in \{1,2\}$ such that for every action $a_i \in A_i$ of player i played with non-zero probability in γ it holds that $U_i(\gamma|a_i) \geq U_i(\sigma_i)$, where σ_i is the worst Nash equilibrium for i in Γ. Say $i = 1$ without loss of generality. Consider the protocol which runs an unfair, active secure two-party computation where first player 1 learns a_1 and then in the following round player 2 learns a_2 or learns that player 1 aborted. If player 1 aborts, then player 2 punishes by playing the worst NE for player 1. It seems this should work as player 1 now has no incentive to deviate and player 2 cannot deviate as he learns his recommendation a_2 last. However, this does not work! What player 2 will do if he receives a bad recommendation a_2, i.e., one where $U_2(\gamma|a_2) < U_2(\sigma_2)$, where σ_2 is the worst Nash equilibrium for i in Γ, is to send his entire view, including a_2 to player 1, just before actions are to be played. Now that player 1 has no uncertainty on the view of player 1, all stable ways for the two players to pick their actions in the face of this deviation will give player 2 a payoff which is at least as good as in σ_2.

Consider then the attempt to use gradual release to give a_1 and a_2 to the players, the hope being that we can release a_1 and a_2 in a way such that when learning a_i it is too late to prevent the other party from learning a_{-i}. Again, this is in vain, as the *reveal your view* deviation is played *after* both a_1 and a_2 are fully revealed. For the same reason techniques for fair computation between rational players will fail too, like the protocol in Groce and Katz [10].

We consider it very interesting future work to consider variations of empty-threat freeness which prevent the *reveal your view* deviation, more specifically, can we give realistic models of empty-threat freeness allowing to implement larger classes of CE?

6 All Minicrypt Payoffs Iff One-Way Functions Exist

Recall that we denote Minicrypt utility profiles to be the utility profiles achieved by some non-trivial CHNE. In this section we justify the name by showing that there exists a Minicrypt utility profile which requires one-way functions to be computational cheap talk implemented. This complements the result by Gradwohl *et al.* [9] that one-way functions are sufficient to implement any Minicrypt utility profile.

6.1 Implementing All Minicrypt Payoffs Implies One-Way Functions

In this section we show how to use a computational cheap talk implementation of some CHNE achieving a Minicrypt payoff to construct a protocol for weak coin-flip.

Given a two-party protocol $\pi = (\pi_1, \pi_2)$ with no inputs, and outputs which are in $\{0,1\}$. Let $y_i(\pi) \in \{0,1\}$ denote the output of π_i after running π. Note that $y_i(\pi)$ is a random variable, with the universe being the randomness used by P_1 and P_2 in the run of the protocol. A weak coin-flip protocol is such a protocol, where the following holds:

1. If both players are honest, then they output the same value, i.e., $y_1(\pi_1,\pi_2) = y_2(\pi_1,\pi_2)$. Moreover, $\Pr[y_1(\pi_1,\pi_2)=0] = \Pr[y_1(\pi_1,\pi_2)=1] = \frac{1}{2}$.
2. For any efficient strategy π_1^* of P_1 it holds that $\Pr[y_2(\pi_1^*,\pi_2)=0] \leq \frac{1}{2}+\varepsilon$ for a negligible ε.
3. For any efficient strategy π_2^* of P_2 it holds that $\Pr[y_1(\pi_1,\pi_2^*)=1] \leq \frac{1}{2}+\varepsilon$ for a negligible ε.

It follows from the seminal work of Impagliazzo and Luby [12] that weak coin-flip implies one-way functions.[5]

Consider the CTSM game specified by $\Gamma = (A_1, A_2)$, where $A_1 = \{\mathtt{c},\mathtt{d}\}, A_2 = \{\mathtt{C},\mathtt{D}\}$, and the utility function u is given by the payoff matrix:

	C	D
c	1,1	0,4
d	4,0	0,0

The probability distribution selecting $(\mathtt{c},\mathtt{D})$ and $(\mathtt{d},\mathtt{C})$ with equal probability is a convex hull NE achieving the utility profile $(2,2)$. We show that if it is possible to implement such CHNE using cryptographic cheap talk, then one-way functions exist.

Theorem 3. *If there exists in the CTSM game corresponding to Γ a computational NE σ achieving utility profile $(2,2)$, then one-way functions exist.*

[5] The notion is defined slightly different in [12], but by letting a party P_i who outputs "REJECT" output i instead, the notions become equivalent. Note also that opposed to what is common in contemporary definitions, see e.g. [13], we do not require that the winner can be determined from the communication of the protocol. This is in line with the original definition in [12], so we can still use the implication of one-way functions.

Proof. Consider the two-party protocol π given in Fig. 3.

1. For $i \in \{1,2\}$, party P_i runs the cheap talk phase of strategy σ_i of P_i in the strategy profile σ, using uniformly random randomizers. All the messages are forwarded to party P_{-i}, and the round function is computed on the messages forwarded from P_{-i}.
2. If in round m the strategy σ_i plays $\mathtt{d}$ or $\mathtt{C}$, then P_i outputs $y_i = 0$. If σ_i plays $\mathtt{c}$ or $\mathtt{D}$, then P_i outputs $y_i = 1$.

Fig. 3. Protocol for weak coin-flip given a cheap talk implementation of a specific CHNE

The following statements are logically equivalent.

1. There exists an efficient π_1^* such that P_2 outputs 0 in (π_1^*, π_2) with probability $p_0 > \frac{1}{2}$.
2. There exists an efficient σ_1^* such that P_2 plays $\mathtt{C}$ in (σ_1^*, σ_2) with probability $p_0 > \frac{1}{2}$.
3. There exists an efficient σ_1^* such that P_1 has utility $u_0 > 2$ in (σ_1^*, σ_2).
4. There exists an efficient σ_1^* such that P_1 has utility $u_0 > 2$ in (σ_1^*, σ_2) and such that P_1 never plays $\mathtt{c}$.

By construction statement 1 implies statement 2. If statement 2 is true, then the strategy $\sigma_1^\dagger$ which plays like σ_1^* and then plays $\mathtt{d}$ has expected utility $4p_0 > 2$. Statement 3 implies statement 4 because $\mathtt{d}$ is weakly dominating for P_1, i.e, P_1 never gets less utility by playing $\mathtt{d}$ instead of $\mathtt{c}$. If statement 4 is true, then $4\alpha + 0(1-\alpha) > 2$, where α is the probability that P_2 plays $\mathtt{c}$ in (σ_1^*, σ_2). This implies that $\alpha > \frac{1}{2}$. By letting π_1^* be the strategy playing like σ_1^*, this implies statement 1.

If both parties follow the protocol in Fig. 3 then they both output the same bit b, and it is 0 or 1 with equal probability. Since σ is a computational equilibrium of $(\Gamma, C_\emptyset)$, any player can increase her utility by at most negligible amount. Thus, any player can bias the output of the protocol by at most negligible amount towards her preferred outcome, and the protocol is a weak coin-flip protocol. □

7 All Cryptomania Payoffs iff OT Exists

In this section we show that there exist Cryptomania profiles which imply OT. Implementing any Cryptomania profile given OT follows from [7]. We will also conjecture that implementing any Cryptomania profile implies OT and give supporting evidence.

We recall the notion of *random Rabin OT*. It is a secure two-party computation specified by a randomized function $f(x_1, x_2) = (y_1, y_2)$. The outputs do not depend on the inputs (x_1, x_2). The output y_1 is a bit $y_1 \in \{0,1\}$. The output y_2 is a trit $y_2 \in \{0,1,\perp\}$. The bit y_1 is uniformly random. The probability that $y_2 = \perp$ is $\frac{1}{2}$, independent of y_1. And, if $y_2 \neq \perp$, then $y_2 = y_1$. Note that this implies that party 1 gets no information on whether $y_2 = y_1$ or $y_2 = \perp$ and that if $y_2 = \perp$, then party 2 has no information on y_1. We call a protocol a *semi-honest random Rabin OT* if it implements random Rabin OT

against parties guaranteed to follow the protocol in the model [5]. Semi-honest random Rabin OT is interesting as it is known to be complete for two-party computation, even for active secure two-party computation which can tolerate that the parties deviate from the protocol.

Given semi-honest random Rabin OT one can empty-threat free implement any NE-punishable CE. One uses an active-secure two-party computation to sample the CE and punishes a deviating party by playing the worst NE for that party. The proof that this is empty-threat free follows the proof of Gradwohl *et al.* [9]. We now show that OT is needed for having an implementation of all Cryptomania profiles.

7.1 Playing Chicken Well Implies OT

In this section we show that there exists a version of Chicken which has a CE with a weakly Pareto optimal utility profile which cannot be obtained using a computational NE in the corresponding cheap talk game, unless OT exists. The game has two actions per player, which shows that even in the simplest non-trivial game setting, one can only harvest the maximal utility if OT exists.

Consider the CTSM game specified by $\Gamma_{\texttt{chicken}} = (A_1, A_2, u)$, where $A_1 = \{\texttt{c}, \texttt{d}\}$, $A_2 = \{\texttt{C}, \texttt{D}\}$ and the utility function u is given by the payoff matrix:

	C	D
c	15,15	6,21
d	21,6	0,0

Theorem 4. *If there exists a computational NE σ for the CTSM game corresponding to $\Gamma_{\texttt{chicken}}$ achieving utility profile $(14,14)$, then there exists a protocol for semi-honest random Rabin OT.*

Proof. Let σ be as in the premise. We assume that $u(\sigma) = (14,14)$—extending the proof to handling the case where the payoff of each player i is $14 - \varepsilon_i$ for a negligible ε_i is standard. In the following we use $\text{view}_i(\sigma) = \text{view}_i(\Gamma, \sigma, C)$ to denote the view of player i when the parties play according to σ.

Consider the following two-party protocol π:

1. Party P_i runs the cheap talk phase of strategy σ_i of P_i in the strategy profile σ, using uniformly random randomizers.
2. If in the last round the strategy σ_i plays c or C, then P_i outputs $b_i = 1$. If σ_i plays d or D, then P_i outputs $b_i = 0$.

Let view_i denote the view of party P_i in a run of this protocol. We are going to analyze the distribution of the output of the parties and the distribution of their views, and then conclude that they imply OT.

Since the expected utility $(14,14)$ is symmetric, we know that σ plays $(\texttt{d}, \texttt{C})$ as much as it plays $(\texttt{c}, \texttt{D})$; call the probability of playing each of these α. Let β denote the probability that σ plays $(\texttt{c}, \texttt{C})$. We clearly have that $2\alpha \leq 1 - \beta$. The expected utility is therefore $\alpha(21,6) + \alpha(6,21) + \beta(15,15) \leq 2\alpha(13.5,13.5) + (1-2\alpha)(15,15)$. From $14 \leq 2\alpha 13.5 + (1-2\alpha)15$, it follows that $\alpha \leq \frac{1}{3}$. This means that the expected utility is

at most $\frac{1}{3}21+\frac{1}{3}6+\beta 15$. From $\frac{1}{3}21+\frac{1}{3}6+\beta 15 \geq 14$, we get that $\beta \geq \frac{1}{3}$. The expected utility of P_2 when σ_2 plays $\mathtt{C}$ is $\frac{\beta}{\alpha+\beta}15+\frac{\alpha}{\alpha+\beta}6$. If P_2 would switch to $\mathtt{D}$ when σ_1 says to play $\mathtt{C}$, then the expected utility of P_2 would become $\frac{\beta}{\alpha+\beta}21+\frac{\alpha}{\alpha+\beta}0$. It follows from the fact that σ is a computational NE that $\beta 21 \leq \beta 15+\alpha 6-\varepsilon$ for some negligible ε. We will assume that $\varepsilon = 0$—handling the negligible ε is standard. From $\beta 21 \leq \beta 15+\alpha 6$ we get that $\beta \leq \alpha$. From $\alpha \leq \frac{1}{3}$, $\beta \geq \frac{1}{3}$ and $\beta \leq \alpha$ we get that $\alpha = \beta = \frac{1}{3}$. This means that the joint output of $(\mathsf{P}_1, \mathsf{P}_2)$ in π is uniform on $\{(0,1),(1,0),(1,1)\}$. One can show that an expected constant number of samples from this distribution is sufficient to implement random Rabin OT, see the full version for the details. This, however, is not sufficient to conclude the proof, as the transcript of π might leak information. To finish the proof we therefore have to show that the parties have no extra information to their outputs, i.e., show that

$$[\text{view}_1 \,|b_1 = 1 \wedge b_2 = 1] \approx [\text{view}_1 \,|b_1 = 1 \wedge b_2 = 0]$$
$$[\text{view}_2 \,|b_1 = 1 \wedge b_2 = 1] \approx [\text{view}_2 \,|b_1 = 0 \wedge b_2 = 1] \,,$$

where $\approx$ denotes computational indistinguishability. We show the first relation. The second follows using a symmetric argument.

Assume that there exists an efficient distinguisher D which can distinguish $[\text{view}_1 \,|b_1 = 1 \wedge b_2 = 1]$ and $[\text{view}_1 \,|b_1 = 1 \wedge b_2 = 0]$ with non-negligible probability, i.e., $|\Pr[D([\text{view}_1 \,|b_1 = 1 \wedge b_2 = 1]) = 1] - \Pr[D([\text{view}_1 \,|b_1 = 1 \wedge b_2 = 0]) = 1]|$ is non-negligible. Since we work with non-uniform complexity, we can assume that it is always the case that $\Pr[D([\text{view}_1 \,|b_1 = 1 \wedge b_2 = 1] = 1)] \geq \Pr[D([\text{view}_1 \,|b_1 = 1 \wedge b_2 = 0] = 1)]$. Now consider the following strategy σ_1^*. It plays like σ_1, except that if σ_1 recommends to play $\mathtt{c}$, then σ_1^* switches to $\mathtt{d}$ when $D(\text{view}_1) = 1$, where view_1 is the view of P_1. Note that σ_1 recommending to play $\mathtt{c}$ is logically equivalent to $b_1 = 1$. I.e., $\text{view}_1 \in \{[\text{view}_1 \,|b_1 = 1 \wedge b_2 = 1], [\text{view}_1 \,|b_1 = 1 \wedge b_2 = 0]\}$. Furthermore, since $\alpha = \beta$, we have that b_2 is uniformly random. We use this to compute the utility of switching. We look at the cases that the joint play of σ is $(\mathtt{c},\mathtt{C})$ and $(\mathtt{c},\mathtt{D})$ separately. If the joint play is $(\mathtt{c},\mathtt{C})$, then we switch with probability $\Pr[D([\text{view}_1 \,|b_1 = 1 \wedge b_2 = 1]) = 1]$, for a gain of $\Pr[D([\text{view}_1 \,|b_1 = 1 \wedge b_2 = 1]) = 1](21-15)$. If the joint play is $(\mathtt{c},\mathtt{D})$, then we switch with probability $\Pr[D([\text{view}_1 \,|b_1 = 1 \wedge b_2 = 0]) = 1]$, for a gain of $\Pr[D([\text{view}_1 \,|b_1 = 1 \wedge b_2 = 0]) = 1](0-6)$. This gives a total gain of $6(\Pr[D([\text{view}_1 \,|b_1 = 1 \wedge b_2 = 1]) = 1] - \Pr[D([\text{view}_1 \,|b_1 = 1 \wedge b_2 = 0]) = 1])$. This means that the gain is six times the advantage of D, which is non-negligible. This is a contradiction to σ being a computational NE. □

7.2 Perfectly Implementing any CE Outside CHNE Implies Unconditional OT

We now justify the conjecture that cheap talk implementing any Cryptomania profile implies OT. In particular, we show that if the implementation had been perfect, in the sense that it only leaks the recommendations, then one can always implement OT. We leave it as an open problem to investigate whether the additional protocol transcript of a cheap talk implementation of the correlation device in general leaks sufficiently little information that the result also holds for computational cheap talk implementations.

Theorem 5. *Let γ be a Cryptomania correlation device for a game Γ, i.e., it outputs recommendations which are not in the CHNE of Γ. Then given a polynomial number of samples of γ, two parties can implement unconditionally secure OT against semi-honest adversaries in the model [5].*

We use the result of Crépeau, Morozov and Wolf [6] that any non-trivial Discrete Memoryless Channel implies OT. Thus, it suffices to show that there are some correlation devices that can be used to simulate a non-trivial DMC; the existence of any such correlation device would consequently imply the existence of OT.

Definition 9 (Discrete Memoryless Channel). *A* discrete memoryless channel *is characterized by an input alphabet $\mathcal{A}_X$, an output alphabet $\mathcal{A}_Y$, and a set of conditional probability distributions $P_{y|x}$ for each $x \in \mathcal{A}_X$.*

Note that the binary symmetric channel with probability of error $p \in [0,1]$ is a special case of DMC with $\mathcal{A}_X = \mathcal{A}_Y = \{0,1\}$, and the conditional probabilities $P_{1|0} = P_{0|1} = p$, and $P_{0|0} = P_{1|1} = 1 - p$.

Wolf and Wullschleger [20] considered the problem of two parties with access to correlated random variables X, and Y trying to simulate a DMC characterized by the conditional probabilities $P_{Y|X}$. A correlated equilibrium γ of a strategic two player game corresponds to an identical situation. The two players have access to two correlated random variables that are defined by the randomized advice about what action each one of them should take in the game. Given access to the correlation device, the players can simulate a discrete memoryless channel as described in Fig. 4.

To send bit $d \in \{0,1\}$ from party A to party B:

1. Both players get advice according to γ, and use rejection sampling to make sure that the pair of advice they get is an element $(a,b) \in \{a_0,a_1\} \times \{b_0,b_1\}$ for some actions a_0,a_1 of player A and b_0,b_1 of player B. They use the correlation device for γ multiple times, until both a_0 and a_1 appear in the list of advice received by player A.
2. Party A erases some advice from her list to make a_0 and a_1 equiprobable, and sends to B the index i of the first occurrence of a_d in her list.
3. Party B outputs d', such that $b_{d'}$ is the i-th advice in the list of player B.

Fig. 4. Simulating a DMC when given access to some correlation device for a CE γ

This procedure simulates a DMC defined by the conditional probabilities $P_{y|x}$ corresponding to the CE restricted by the rejection sampling to $\{a_0,a_1\} \times \{b_0,b_1\}$; for example the probability of receiving 0 after sending 1 is $P_{0|1} = \gamma(a_1,b_0)/(\gamma(a_1,b_0) + \gamma(a_1,b_1))$. Note that this procedure in general does not simulate the binary symmetric channel.[6] However, we show that for non-trivial CE the properties of the associated DMC are good enough to imply OT.

We are interested in DMCs that are non-trivial in the following sense.

[6] Some non-trivial CE indeed give rise to well-known channels. For example the CE from previous section corresponds to the Z-channel.

Definition 10 (Crépeau *et al.*[6]). *We call a channel $P_{Y|X}$* trivial *if there exist, after removal of all redundant input symbols, partitions of the (remaining) ranges $\mathcal{X}$ of X and $\mathcal{Y}$ of Y , $\mathcal{X} = \mathcal{X}_1 \cup \ldots \cup \mathcal{X}_n, \mathcal{Y} = \mathcal{Y}_1 \cup \ldots \cup \mathcal{Y}_n$, and channels $P_{Y_i|X_i}$, where the ranges of X_i and Y_i are $\mathcal{X}_i$ and $\mathcal{Y}_i$, respectively, such that*

$$P_{Y|X=x}(y) = \begin{cases} P_{Y_i|X_i=x}(y) & \text{if } x \in \mathcal{X}_i, y \in \mathcal{Y}_i, \\ 0 & \text{if } x \in \mathcal{X}_i, y \in \mathcal{Y}_j, i \neq j \end{cases}$$

holds and such that the capacity of the channel $P_{Y_i|X_i}$ is 0 for all i.

The following lemma justifies the use of correlated equilibria outside the convex-hull of NE to simulate non-trivial DMCs.

Lemma 3. *Let Γ be a strategic game, and γ some correlated equilibrium of Γ. If γ is a CE of Γ outside the convex hull of NE, then there exist a pair of actions $a_i \neq a_j$ of player A and a pair of actions $b_k \neq b_l$ of player B, such that the restriction of γ to $\{a_i, a_j\} \times \{b_k, b_l\}$ allows to simulate a non-trivial DMC.*

Proof. Recall that $P_{b|a} = \gamma(a,b)/(\gamma(a,b_k) + \gamma(a,b_l))$ for any $(a,b) \in \{a_i, a_j\} \times \{b_k, b_l\}$. Since γ is not a CHNE of Γ, there must exist actions $a_i \neq a_j$ of player A and $b_k \neq b_l$ of player B, such that

$$P_{b_k|a_i} \neq P_{b_k|a_j}, \text{ or } P_{b_l|a_i} \neq P_{b_l|a_j} \tag{7.1}$$

(or else γ is a completely mixed NE of Γ). We want to show that the conditional probabilities $P_{b|a}$ characterize a channel with non-zero capacity. Condition (7.1) ensures that it is never the case that $P_{b_k|a_i} = P_{b_l|a_i} = P_{b_k|a_j} = P_{b_l|a_j} = 1/2$. Thus, the resulting DMC does not have entropy 1 (i.e. it has non-zero capacity). On the other hand, we need to show that the resulting DMC has enough entropy to be non-trivial, i.e., that it is not a perfect channel or a channel outputting always the same symbol. It suffices to show that among the tuples of actions consistent with the condition (7.1) we can in fact select the actions a_i, a_j and b_k, b_l so that at most one of the conditional probabilities $P_{b|a}$ is zero. Equivalently, we instead show that it is possible to select the actions where at most one of $\gamma(a,b)$ is equal to zero. We call a candidate *bad* if it has more than one 0. Note that no bad candidate has $\gamma(a_i,b_k) = \gamma(a_j,b_k) = 0$ or $\gamma(a_i,b_l) = \gamma(a_j,b_l) = 0$, since then $P_{b_k|a_i} = P_{b_k|a_j} = 0$ and $P_{b_l|a_i} = P_{b_l|a_j} = 1$, respectively $P_{b_l|a_i} = P_{b_l|a_j} = 0$ and $P_{b_k|a_i} = P_{b_k|a_j} = 1$. So, bad candidates are either of the row type, $\gamma(a_i,b_k) = \gamma(a_i,b_l) = 0$ or $\gamma(a_j,b_k) = \gamma(a_j,b_l) = 0$, or the diagonal type, $\gamma(a_i,b_k) = \gamma(a_j,b_l) = 0$ or $\gamma(a_i,b_l) = \gamma(a_j,b_k) = 0$. One can use this to show that it holds for any two actions a_i and a_j that the residual distribution given a_i and a_j is either identical or disjoint. I.e., either $\gamma(a_i,b_k)/\gamma(a_i) = \gamma(a_j,b_k)/\gamma(a_j)$ for all b_k or $\gamma(a_i,b_k) = 0 \vee \gamma(a_j,b_k) = 0$ for all b_k. This shows that the distribution is a sum of product distributions, each a NE, i.e., a CHNE. There are more details in the full version. □

The following theorem characterizes DMCs with respect to the possibility of their use to create unconditional OT:

Theorem 6 (Crépeau *et al.*[6]). *Let two players A and B be connected by a non-trivial channel $P_{Y|X}$. Then, for any $\alpha > 0$, there exists a protocol for unconditionally secure OT from A to B with failure probability at most α, where the number of uses of the channel is of order $O(\log(1/\alpha)^{2+\varepsilon})$ for any $\varepsilon > 0$. Trivial channels, on the other hand, do not allow for realizing OT in an unconditional way.*

Lemma 3 together with the above result of Crépeau *et al.* [6] give the sought proof of Theorem 5.

Acknowledgements. The authors would like to thank for discussions and useful comments to Ronen Gradwohl, Jonathan Katz, Noam Livne, Peter Bro Miltersen, and Margarita Vald.

References

[1] Atallah, M.J., Blanton, M., Frikken, K.B., Li, J.: Efficient correlated action selection. In: Di Crescenzo, G., Rubin, A. (eds.) FC 2006. LNCS, vol. 4107, pp. 296–310. Springer, Heidelberg (2006)
[2] Aumann, R.J.: Subjectivity and correlation in randomized strategies. Journal of Mathematical Economics 1(1), 67–96 (1974)
[3] Aumann, R.J., Hart, S.: Long cheap talk. Econometrica 71(6), 1619–1660 (2003)
[4] Bárány, I.: Fair distribution protocols or how the players replace fortune. Mathematics of Operations Research 17(2), 327–340 (1992)
[5] Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. IACR Cryptology ePrint Archive 2000, 67 (2000)
[6] Crépeau, C., Morozov, K., Wolf, S.: Efficient unconditional oblivious transfer from almost any noisy channel. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 47–59. Springer, Heidelberg (2005)
[7] Dodis, Y., Halevi, S., Rabin, T.: A cryptographic solution to a game theoretic problem. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 112–130. Springer, Heidelberg (2000)
[8] Farrell, J., Rabin, M.: Cheap talk. The Journal of Economic Perspectives 10(3), 103–118 (1996)
[9] Gradwohl, R., Livne, N., Rosen, A.: Sequential rationality in cryptographic protocols. In: FOCS, pp. 623–632. IEEE Computer Society (2010)
[10] Groce, A., Katz, J.: Fair computation with rational players. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 81–98. Springer, Heidelberg (2012)
[11] Impagliazzo, R.: A personal view of average-case complexity. In: Structure in Complexity Theory Conference, pp. 134–147 (1995)
[12] Impagliazzo, R., Luby, M.: One-way functions are essential for complexity based cryptography (extended abstract). In: FOCS, pp. 230–235. IEEE Computer Society (1989)
[13] Maji, H.K., Prabhakaran, M., Sahai, A.: On the computational complexity of coin flipping. In: FOCS, pp. 613–622. IEEE Computer Society (2010)

[14] Nash, J.: Non-cooperative games. Annals of Mathematics 54(2), 286–295 (1951)
[15] Osborne, M.J., Rubinstein, A.: A course in game theory. MIT Press (1994)
[16] Pass, R., Shelat, A.: Renegotiation-safe protocols. In: Chazelle, B. (ed.) ICS, pp. 61–78. Tsinghua University Press (2011)
[17] Teague, V.: Selecting correlated random actions. In: Juels, A. (ed.) FC 2004. LNCS, vol. 3110, pp. 181–195. Springer, Heidelberg (2004)
[18] Teague, V.: Problems With Coordination in Two-Player Games: Comment on "Computational Complexity and Communication". Econometrica 76(6), 1559–1564 (2008)
[19] Urbano, A., Vila, J.E.: Computational complexity and communication: Coordination in two–player games. Econometrica 70(5), 1893–1927 (2002)
[20] Wolf, S., Wullschleger, J.: Zero-error information and applications in cryptography. In: Information Theory Workshop, pp. 1–6. IEEE (October 2004)

Accuracy-Privacy Tradeoffs for Two-Party Differentially Private Protocols

Vipul Goyal[1,⋆], Ilya Mironov[2], Omkant Pandey[3,⋆], and Amit Sahai[4,⋆⋆]

[1] Microsoft Research India
[2] Microsoft Research Silicon Valley
[3] The University of Texas at Austin
[4] University of California Los Angeles

Abstract. Differential privacy (DP) is a well-studied notion of privacy that is generally achieved by randomizing outputs to preserve the privacy of the input records. A central problem in differential privacy is how much accuracy must be lost in order to preserve input privacy?

Our work obtains general upper bounds on accuracy for differentially private two-party protocols computing any Boolean function. Our bounds are independent of the number of rounds and the communication complexity of the protocol, and hold with respect to computationally unbounded parties. At the heart of our results is a new general geometric technique for obtaining non-trivial accuracy bounds for any Boolean functionality.

We show that for *any* Boolean function, there is a constant accuracy gap between the accuracy that is possible in the client-server setting and the accuracy that is possible in the two-party setting. In particular, we show *tight* results on the accuracy that is achievable for the AND and XOR functions in the two-party setting, completely characterizing which accuracies are achievable for any given level of differential privacy.

Finally, we consider the situation if we relax the privacy requirement to computational differential privacy. We show that to achieve any noticeably better accuracy than what is possible for differentially private two-party protocols, it is essential that one-way functions exist.

1 Introduction

SFE and Differential Privacy. *Secure function evaluation* (SFE) is a fundamental concept in cryptography. Informally, SFE allows two parties to compute a joint function of their inputs without learning anything other than the

⋆ Part of the work was done at Microsoft Research Silicon Valley.

⋆⋆ Research supported in part from a DARPA/ONR PROCEED award, NSF grants 1228984, 1136174, 1118096, 1065276, 0916574 and 0830803, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, and an Okawa Foundation Research Grant. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0389. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 298–315, 2013.

value of the function. An important research program in SFE is characterization of assumptions necessary for computing various classes of functionalities. For instance, an early result of Chor and Kushilevitz [7] established a zero-one law for Boolean functions in the information-theoretic model (against a computationally-unbounded passive adversary).

Differential privacy (DP) is a theoretically sound and practically important notion of privacy [10,12]. In contrast with SFE (which places the "output correctness first and privacy second"), it limits information leaked through the *output* of the function (i.e., places "privacy first and output correctness second"). Differential privacy mechanisms work by randomizing the output to preserve the privacy of the input records. Thus the main question in differential privacy is *quantitative* rather than *qualitative*: How much accuracy must be lost in order to preserve input privacy?

The problem of characterizing accuracy of differentially private mechanisms is well-defined and very challenging even in the case of a single party's holding the input (the client-server setting). However if the input is distributed across several parties, output needs to be computed through an interactive protocol. Throughout the protocol, parties are restricted in how much information their messages should reveal about their input, and this would seem to degrade the quality of the output.

In other words, differential privacy gives a new set of restrictions on a protocol. Compared with the long line of research on feasibility and completeness of SFEs and MPCs for various functionalities in the semi-honest computationally-unbounded setting [3,6,24,1,7,22,23], our understanding of the corresponding properties of differentially private protocols is remarkably incomplete.

The notion of differential privacy has been studied in the distributed setting, starting with the seminal work of Dwork and Nissim [13]. In their work, there are multiple parties each holding a dataset as the input. The study of limitations on accuracy of distributed differentially private protocols was initiated in works of Beimel et al. [2] for the case of n parties each holding its own input, followed up by Chan et al. [5], and of McGregor et al. [25] for the setting of two parties with n-bit inputs. The latter work considers several natural and constructed functionalities that exhibit a stark gap in accuracy that can be as large as $\Theta(n)$ between client-server and two-party protocols.

However, many questions remain. While we have several examples of functionalities for which there is an accuracy gap between the client-server setting and the distributed setting, does such a gap exist for *any* non-trivial functionality? How large must this gap be? Answering these questions for a large and natural class of functionalities in the two-party setting is the main focus of this work.

Boolean Functionalities. In this work, we focus on protocols that attempt to compute a Boolean function. While much work in differential privacy has focused on computing statistics, we note that computing Boolean functions has long been a motivating goal in differential privacy (e.g., answering questions "Does smoking cause cancer?" or "Do millionaires pay proportionally less in tax than their secretaries?"). Our goal is to obtain a characterization of which

Boolean functions must suffer accuracy loss in the two-party setting, as well as lower bounds on how much accuracy loss is inherently needed. We note that our understanding of Boolean functions has been particularly (and perhaps surprisingly) weak: Before this work, even for computing simple Boolean gates, like AND and XOR, we did not understand whether *any* accuracy loss is essential to the two-party setting.

1.1 Our Results

Before we describe our results, we must define the notion of accuracy that we measure. Since our focus is on functions with Boolean output, there is only one natural choice for accuracy measure: the probability that the output is correct. We note that other metrics considered in the literature do not apply to the Boolean setting.

Now we discuss our setting in more detail. There are two parties Alice and Bob holding inputs x and y respectively and interested in computing a Boolean function $f(x, y)$. The protocol should be such that the differential privacy of each bit of x as well as of y should be preserved[1]. We assume that Alice and Bob follow the protocol as specified, but keep a record of what transpired during the protocol (i.e., they are semi-honest in the cryptographic sense).

For a protocol to achieve accuracy a, it must be the case that for any possible inputs (x, y) to the protocol, the protocol computes the correct output with probability at least a, over the coins of the protocol. We concentrate on the worst-case (over the parties' inputs) measure of accuracy as it is the most general type of guarantee for a randomized protocol, independent of distributional assumptions.

Informally speaking, the differential privacy (DP) constraint for Alice states that for any two inputs x_0, x_1 for Alice that differ only in one bit, and for any input y for Bob, the following must hold: For every possible execution of the protocol, the resulting view v of Bob must be such that the probability that v arises on inputs (x_0, y) is within a multiplicative factor of e^ϵ from the probability that v arises on inputs (x_1, y) (see Section 2 for the formal definition of differential privacy). Thus, no matter what Bob sees, he remains uncertain about the value of each bit of Alice's input even if he knows every other bit in her input. Here ϵ is the key privacy parameter. We will denote by λ the value e^ϵ. It is easy to see that in the client-server setting, it is always possible to achieve an accuracy of $\frac{\lambda}{1+\lambda}$ $(= .5 + \Theta(\epsilon)$ for $\epsilon \to 0)$.

Our work obtains general upper bounds on accuracy. Our bounds are independent of the number of rounds and the communication complexity of the protocol, and hold with respect to computationally unbounded parties. At the heart of our results is a new general geometric technique for obtaining non-trivial accuracy bounds for any Boolean functionality.

[1] Stronger notions of privacy are also interesting: for example, where symbols larger than bits, or the entire input of each party should be protected. However, since our focus is on obtaining lower bounds on error, we use the weaker notion of the privacy stated here, with respect to bits.

General Boolean Functions. Our strategy to obtain results on two-party differentially private protocols for general Boolean functions begins by reducing the problem of obtaining upper bounds for general Boolean functions to specific, simple functions. We first note that Boolean functions where one party's input completely determines the output can, of course, be computed just as accurately in the two-party setting as in the client-server setting. We call such functions trivial, following works on classifying which Boolean functions have statistically-secure two-party SFE protocols. We then show that the existence of an ϵ-DP protocol with accuracy a for any *non-trivial* function implies the existence of an ϵ-DP protocol with accuracy a for either the AND or XOR functionalities (defined below). Thus, if we can obtain upper bounds on accuracy for AND and XOR, we obtain upper bounds on accuracy for all non-trivial Boolean functions.

Computing an AND Gate. The AND functionality is as follows: Alice and Bob each hold a bit denoted by x and y respectively and are interested in computing the AND of the two bits. Given the output (and the protocol transcript), each input bit should remain private. Naively, the best differentially private protocol for this task is the randomized response protocol: each party individually perturbs its input and sends it out. The parties then compute the output based on the two input bits appearing in the protocol transcript. It is easy to see that the output and the protocol transcript still maintain privacy of each individual bit; moreover, both players' bits are released with maximal possible accuracy. The randomized response technique gives protocols for AND with accuracy $\frac{\lambda^2}{(1+\lambda)^2}$, which for $\lambda < 1 + \sqrt{2}$ is worse than a random guess.

We show that by augmenting the parties' outputs with one additional symbol, it is possible to improve on the naïve protocol. The new protocol can achieve an accuracy of $\frac{\lambda(\lambda^2+\lambda+2)}{(1+\lambda)^3}$. Moreover, we show that this accuracy is *optimal* for AND, even for protocols with any number of rounds and unbounded (finite) communication complexity. For $\epsilon \to 0$ and $\lambda \approx 1 + \epsilon$ the protocol's advantage over a random guess is $\Theta(\epsilon)$, in line with the canonical protocol in the client-server setting.

Computing an XOR Gate. The XOR functionality is defined analogously to the AND functionality above, except the XOR of the two input bits is to be computed. For the XOR case, the randomized response technique provides an accuracy of $\frac{1+\lambda^2}{(1+\lambda)^2} = .5 + \Theta(\epsilon^2)$ for $\epsilon \to 0$. We show that this is, in fact, optimal for XOR.

Combining the results above, we establish the following: *There does not exist any non-trivial Boolean functionality which can be computed with a differential private protocol in the two party setting with accuracy matching that of the client-server setting.* In fact, we obtain a separation between the level of accuracy obtainable in the client-server setting and the two-party setting for every non-trivial Boolean functionality, where the separation is tight in the case of AND and XOR, and for the XOR functionality is asymptotically significant. Our bounds are shown in Figure 1.

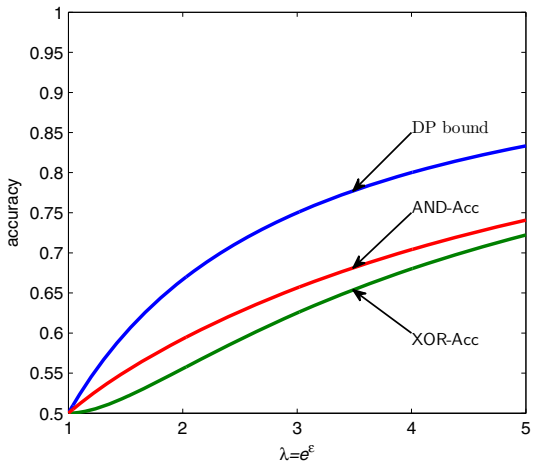


Fig. 1. Bounds on accuracy: for arbitrary Boolean functionalities (DP bound); for protocol-compatible ϵ-DP AND and XOR. Every ϵ-DP protocol for any non-trivial Boolean functionality must be subject to either the AND or XOR bound.

Computational Differential Privacy: What Assumption Is Necessary? One option to restore accuracy in a distributed setting is to resort to a relaxed computational notion of differential privacy [26]. In computational differential privacy (CDP), we relax the privacy condition to require that no *efficient* adversary can predict any bit of the input with probability greater than $\frac{\lambda}{1+\lambda}$, even if the adversary knows all other bits. We ask the question: what computational assumptions are necessary for CDP to enable greater accuracy?

We show that to achieve *any* noticeably greater accuracy with CDP protocols than what is possible with DP protocols, *one-way functions are required*. We show this by presenting a more general result, showing that if one-way functions do not exist, then any CDP protocol must in fact also be a DP protocol.

When discussing CDP protocols, it is important to consider the relationship between CDP protocols and secure computation protocols from cryptography [30,15]. The two notions answer essentially orthogonal problems of *what* and *how*:

- In (computationally) differentially private protocols, "privacy comes first". We would like to ensure privacy of each individual input and then with this constraint, would like to compute an accurate output (the *what* question).
- In secure computation protocols, "accuracy comes first". We would like to release an accurate output to the function we are computing and then with this constraint, would like to ensure privacy of inputs (the *how* question).

Nevertheless, it is immediate that general secure computation methods do give a way to achieve the same level of accuracy in CDP two-party protocols as in the client-server setting. To this end, a secure computation can be used to compute

the algorithm that the server would perform on the joint input in the client-server setting. However, general secure computation is essentially equivalent to secure Oblivious Transfer [21,19]. It remains an important open question whether accuracy better than what is possible with DP protocols may be achievable based on assumptions weaker than the existence of secure Oblivious Transfer protocols.

1.2 Our Techniques

We present a new general geometric technique for bounding the accuracy of differentially private protocols. At a high level, our technique gives us a method for taking the truth table of a function f, a privacy parameter ϵ, and an accuracy level a, and converting this into a linear program P. We prove that if there does exist an ϵ-DP protocol for computing f with accuracy a, then this linear program must have a solution. By analyzing this LP in the case of specific functions, we can show that no solution exists when a is greater than a bound a^*. This proves that no ϵ-DP protocol can exist with accuracy greater than a^*.

For simplicity, let us focus on protocols for Boolean functions where each party holds a single bit. To obtain our bounds, we first think of every possible "transcript" corresponding to some execution of the protocol. We can associate with each such transcript a 2-by-2 "transcript matrix," whose entries are the probability that this transcript occurs when Alice and Bob start with a particular pair of inputs. Now, each such transcript has an associated output value. If we sum together all the transcript matrices with output value 0, we get a 2-by-2 "protocol matrix," whose entries show the probability that the protocol outputs 0 when Alice and Bob start with a particular pair of inputs.

Now let us consider what constraints we can place on these matrices. Two types of constraints are immediate: (1) the differential privacy conditions on each input linearly constrain each transcript matrix; and (2) the accuracy conditions linearly constrain the protocol matrix. But these constraints alone would not yield any bound better than $\frac{\lambda}{1+\lambda}$, which is achievable in the client-server setting. The key to obtaining better bounds, and our main obstacle, are conditions which capture the constraint that these matrices must actually arise from a *protocol* between two players. We consider a condition that we call *protocol compatibility* that essentially captures the fact that if the two parties' inputs are drawn from independent distributions, then they must remain independent even when conditioned on any particular protocol transcript. This post-execution independence has been useful in other works on differential privacy including the work of McGregor et al. [25], as well as in works on secure computation such as the work of Kilian [23].

The protocol-compatibility constraint manifests itself as a non-linear constraint on transcript matrices. Note that there can be an enormous (exponential in communication complexity) number of possible transcript matrices, and we do not want to have to consider such a large space of variables. In particular, we do not want our bounds to depend in any way on the communication complexity or the number of rounds in the protocol. We avoid this by proving a key lemma that shows how to optimally combine the linear differential privacy constraints with

the non-linear protocol-compatibility constraint to yield a new linear constraint (Lemma 3.6 in Section 3). This combined linear constraint establishes an upper bound on sums of probabilities from a transcript matrix that combine both the upper and lower bounds from the differential privacy constraints. Because these constraints are linear, they immediately give constraints on the protocol matrix, as well. This gives us our linear program.

We analyze the linear programs that arise specifically for the AND and XOR functionalities, and prove that the linear program is not satisfiable when the accuracy a is higher than a certain value. We prove that these bounds are tight by showing that this accuracy can be achieved for both the AND and XOR functionalities. We stress that our technique is more general, and can be applied to other specific functions to obtain potentially stronger, although not necessarily tight, bounds. (As mentioned above, we focus our attention on AND and XOR because every non-trivial Boolean function must contain an embedded AND or XOR function.)

Related Work. In addition to the works mentioned above, several other works have focused on the issue of accuracy and privacy. In the client-server setting (i.e., where only one party owns the entire database), limitations for a wide class of private algorithms were first shown by Dinur and Nissim [9]. The optimality of differentially private mechanisms has since been studied in different models such as answering multiple linear queries [17], contingency tables [20], or certain classes of low-sensitivity queries [8]. In a surprising result of Ghosh et al. [14], a simple geometric mechanism (a discrete version of the additive Laplacian mechanism) was shown to be universally optimal for releasing a single count query to Bayesian consumers. Recently, Haitner et al. [16] showed that CDP two-party protocols with accuracy improving upon the information-theoretic bound of McGregor et al. cannot be black-box reduced to random oracles.

In the secure function evaluation model against computationally unbounded semi-honest adversaries, characterization of deterministic Boolean functionalities was completed by Chor and Kushilevitz [7], and for randomized functionalities by [23]. These results establish the "all or nothing" nature of two-party computation under information-theoretic reductions. A related question of characterizing complete *deterministic* functionalities in the computational setting was considered by Harnik et al. [18]. Complete classification of randomized functionalities in the computational setting remains an important research problem.

2 Notation and Definitions

Standard Notation. We use symbols $\neg, \vee, \wedge$, and $\oplus$ to denote the standard Boolean operations: NOT, OR, AND, and XOR respectively. The set of natural numbers is denoted by $\mathbb{N}$; for $n \in \mathbb{N}$, we write by $[n]$ as shorthand for the set $\{1, 2, \ldots, n\}$. The Hamming distance between two strings $x, y \in \{0,1\}^n$ is defined as: $|x - y|_h = |\{i \in [n] \colon x_i \neq y_i\}|$, where x_i, y_i denote the ith bit of x, y respectively. We denote by e, the base of the natural logarithm.

We now recall the definition of ϵ-differential privacy [10] and (ϵ, δ)-differential privacy [11].

Definition 2.1 (ϵ-Differential Privacy). *We say that a randomized function $M\colon \{0,1\}^n \mapsto \mathcal{R}$, with a finite range $\mathcal{R}$, is an ϵ-differentially-private (ϵ-DP) mechanism for $\epsilon \geq 0$ if for every $(x, y) \in \{0,1\}^n \times \{0,1\}^n$ satisfying $|x - x'|_h = 1$ and every subset $S \subset \mathcal{R}$ we have that over the randomness of M:*

$$\Pr[M(x) \in S] \leq e^{\epsilon} \times \Pr[M(x') \in S].$$

Definition 2.2 ((ϵ, δ)-Differential Privacy). *We say that a randomized function $M\colon \{0,1\}^n \mapsto \mathcal{R}$, with a finite range $\mathcal{R}$, is an (ϵ, δ)-differentially-private mechanism for $\epsilon, \delta \geq 0$ if for every $(x, y) \in \{0,1\}^n \times \{0,1\}^n$ satisfying $|x - x'|_h = 1$ and every subset $S \subset \mathcal{R}$ we have that over the randomness of M:*

$$\Pr[M(x) \in S] \leq e^{\epsilon} \times \Pr[M(x') \in S] + \delta.$$

We next recall the definition of computational differential privacy which captures differentially privacy for polynomial time tests. We work with the weakest definition, namely ϵ-IND-CDP [26]. In the following, k denotes the security parameter, implicitly available to all algorithms; algorithms are assumed to run in time polynomial in k unless stated otherwise.

Definition 2.3 (ϵ-IND-CDP Privacy). *We say that an ensemble $\{M_\kappa\}_{k \in \mathbb{N}}$ of randomized functions $M_k\colon \{0,1\}^n \mapsto \mathcal{R}_k$ provides ϵ-IND-CDP if there exists a negligible function $\mathrm{negl}(\cdot)$ such that for every probabilistic polynomial time distinguisher A, for every polynomial $p(\cdot)$, for any adjacent strings $x, x' \in \{0,1\}^n$ (i.e., $|x - x'|_h = 1$), for every sufficiently large $k \in \mathbb{N}$, and for every advice string z_k of size at most $p(k)$, it holds that*

$$\Pr\left[A_k(M_k(x)) = 1\right] \leq e^{\epsilon} \times \Pr\left[A_k(M_k(x')) = 1\right] + \mathrm{negl}(\kappa),$$

where we write $A_k(x)$ for $A(1^k, z_k, x)$ and the probability is taken over the randomness of mechanism M_k and the distinguisher A.

Interactive Setting. Let $\pi := \langle A, B \rangle$ be a two-party protocol. Let $\mathrm{VIEW}_\pi^A(x, y)$ be the random variable which, in a random execution of π with inputs x, y for A, B respectively, consists of (x, R_A, trans), where R_A is the randomness used by A and trans is the sequence of messages exchanged between the parties in the sampled execution. For each x, $\mathrm{VIEW}_\pi^A(x, y)$ is a mechanism over the y's. Define $\mathrm{VIEW}_\pi^B(x, y)$ analogously. When dealing with the computational notion, we consider the family of protocols $\{\pi_k\}_{k \in \mathbb{N}}$ and denote the view of A (resp., B) by $\mathrm{VIEW}_\pi^A(k, x, y)$ (resp., $\mathrm{VIEW}_\pi^B(k, x, y)$).

Definition 2.4 (Two-Party Differentially Privacy). *We say that a protocol $\pi := \langle A, B \rangle$ is ϵ-DP (resp., (ϵ, δ)-DP) if the mechanism $\mathrm{VIEW}_\pi^A(x, y)$ is ϵ-DP (resp., (ϵ, δ)-DP) for all values of x and the same holds for $\mathrm{VIEW}_\pi^B(x, y)$. A family of protocols $\{\pi_k\}_{k \in \mathbb{N}}$ is ϵ-IND-CDP if the mechanism $\mathrm{VIEW}_\pi^A(k, x, y)$ is ϵ-IND-CDP for all values of x and every sufficiently large k, and the same holds for $\mathrm{VIEW}_\pi^B(k, x, y)$.*

Finally, our measure of accuracy for Boolean functions simply looks at how often a randomized mechanism outputs the correct output bit in the worst case.

Definition 2.5 (Accuracy). *The accuracy of a randomized Boolean mechanism* $M\colon \{0,1\}^n \mapsto \{0,1\}$ *with respect to a Boolean function* $f\colon \{0,1\}^n \mapsto \{0,1\}$ *is defined as:*

$$\mathsf{Acc_f}(M) = \min_x \{\Pr[M(x) = f(x)]\},$$

where the probability is taken over the randomness of M*.*

The accuracy of a two party protocol $\pi := \langle A, B\rangle$ w.r.t. to $f\colon \{0,1\}^n \times \{0,1\}^n \mapsto \{0,1\}$ is defined as the accuracy of the mechanism $\textsc{out}_\pi\colon \{0,1\}^n \times \{0,1\}^n \mapsto \{0,1\}$ which returns the (official) output of the protocol in a randomly sampled execution of π. The accuracy for a family of protocols $\{\pi_k\}_{k\in\mathbb{N}}$ is defined analogously for each k.

3 Geometric Analysis

It can be shown that AND and XOR gates are *embedded on adjacent inputs* into any non-trivial Boolean two-party functionality, i.e., any functionality whose output is not fully determined by one side's input (the proof appears in the full version). Therefore, it will be sufficient to analyze AND/XOR gates. A similar claim also appears in [4] but does not guarantee the *adjacency* of inputs that embed AND/XOR gates. Adjacency is crucial in our case, since otherwise we cannot conclude that the protocol for AND/XOR have the same privacy parameter ϵ.

We then formulate necessary conditions for existence of a differentially private two-party protocol implementing a randomized two-party Boolean functionality (Section 3.1), and use these conditions towards tight analysis of accuracy of AND and XOR gates achievable via differentially private protocols (Sections 3.2 and 3.3).

3.1 Differential Privacy and Protocol Compatibility

We begin by introducing several definitions pertaining to properties of matrices that describe joint distributions of protocol outcomes as a function of two inputs. For compactness we will use $\lambda = e^\epsilon$ without stating it explicitly throughout the section.

Definition 3.1 (ϵ-DP Matrix). *A* $2^n \times 2^n$ *matrix* P *indexed by strings* $x, y \in \{0,1\}^n$ *is* ϵ-DP *if its elements satisfy the following conditions for all adjacent pairs* $x, x' \in \{0,1\}^n$ *and* $y, y' \in \{0,1\}^n$*:*

$$p_{xy} \le \lambda \cdot p_{xy'},$$
$$p_{xy} \le \lambda \cdot p_{x'y},$$

Definition 3.2 (Protocol-Compatible Matrix). *A $2^n \times 2^n$ matrix P is* protocol compatible *if for all $x_1, x_2, y_1, y_2 \in \{0,1\}^n$ it holds that*

$$p_{x_1 y_1} \cdot p_{x_2 y_2} = p_{x_1 y_2} \cdot p_{x_2 y_1}$$

The next two definitions extend the concepts of differential privacy and protocol compatibility to two-party Boolean functionalities. By convention, we say that a $2^n \times 2^n$ matrix P represents a randomized Boolean functionality f of two inputs if $p_{xy} = \Pr[f(x,y) = 0]$ for all $x, y \in \{0,1\}^n$.

Definition 3.3 (ϵ-DP Functionality). *We call a $2^n \times 2^n$ matrix P an* ϵ-DP functionality *if both P and $\mathbf{1} - P$ are* ϵ-DP *matrices, where $\mathbf{1}$ is the all-ones matrix.*

Definition 3.4. *A $2^n \times 2^n$ matrix P is* protocol-compatible ϵ-DP functionality *if both matrices P and $\mathbf{1} - P$ can be expressed as sums of protocol-compatible* ϵ-DP *matrices, where $\mathbf{1}$ is the all-ones matrix.*

The following theorem establishes necessary conditions for existence of a differentially private two-party protocol for computing a randomized predicate of two n-bit inputs.

Theorem 3.5. *Let π be a randomized ϵ-DP two-party protocol defined over $x, y \in \{0,1\}^n$ and $\pi(x,y)$ be the Boolean output of the protocol. Let P be a matrix of probabilities $p_{xy} = \Pr[\pi(x,y) = 0]$. Then P is a protocol-compatible* ϵ-DP *functionality.*

Proof. We start by showing that P can be expressed as sums of protocol-compatible ϵ-DP matrices. The proof for $\mathbf{1} - P$ is analogous.

Let T_0 be the set of all transcripts τ for which the protocol output is 0, i.e. $\pi(x,y) = 0$. For a fixed x, y, let $\tau \leftarrow \pi(x,y)$ denote that event that in a random execution of π with inputs (x,y), the transcript is τ. Let P_τ be a $2^n \times 2^n$ matrix indexed by n-bit strings such that $P_\tau(x,y) = \Pr[\tau \leftarrow \pi(x,y)] = p_{\tau,xy}$ (say). Then,

$$p_{xy} = \Pr[\pi(x,y) = 0] = \sum_{\tau \in T_0} \Pr[\tau \leftarrow \pi(x,y)] = \sum_{\tau \in T_0} p_{\tau,xy}.$$

Therefore, it holds that $P = \sum_{\tau \in T_0} P_\tau$. It is easy to verify that the matrices P_τ are ϵ-DP matrices. To complete the proof, we now show that each P_τ is protocol-compatible (following [23]).

Let X and Y be independently and uniformly distributed random variables taking values in $\{0,1\}^n$. Then, using Bayes' rule we see that for any two strings x, y, $p_{\tau,xy} = \Pr[X = x, Y = y | \tau \leftarrow \pi(X,Y)] \cdot \Pr[\tau \leftarrow \pi(X,Y)] / \Pr[X = x, Y = y]$. It is well known in communication complexity (e.g., see [25]) that for any two-party protocol π, if the inputs X and Y are independent before the execution, then for any transcript τ of the protocol, X and Y remain independent when conditioned on the transcript being τ. That is, $\Pr[X = x, Y = y | \tau \leftarrow \pi(X,Y)] =$

$\Pr[X = x|\tau \leftarrow \pi(X,Y)] \cdot \Pr[Y = y|\tau \leftarrow \pi(X,Y)]$. Using this with our previous relation, we see that

$$p_{\tau,xy} = p_{x,\tau} \cdot p_{y,\tau} \cdot p_\tau \cdot 2^{2n},$$

where $p_{x,\tau} = \Pr[X = x|\tau \leftarrow \pi(X,Y)]$; $p_{y,\tau}$ and p_τ are defined analogously. It then follows that for any distinct x_1, y_1, x_2, y_2:

$$p_{\tau,x_1y_1} \cdot p_{\tau,x_2,y_2} = p_{x_1,\tau}p_{x_2,\tau}p_{y_1,\tau}p_{y_2,\tau} \cdot p_\tau^2 \cdot 2^{4n} = p_{\tau,x_1y_2} \cdot p_{\tau,x_2y_1}.$$

This completes the proof for protocol-compatibility, and hence the theorem. □

The following lemma plays a critical role in our analysis, as it replaces a per-transcript quadratic constraint imposed by the protocol-compatibility condition with a system of linear inequalities.

Lemma 3.6. *If P is protocol-compatible* ϵ-DP *functionality, then for all adjacent pairs $x, x' \in \{0,1\}^n$ and $y, y' \in \{0,1\}^n$*

$$p_{xy'} + p_{x'y} \le p_{xy}/\lambda + p_{x'y'} \cdot \lambda,$$
$$p_{xy'} + p_{x'y} \le p_{xy} \cdot \lambda + p_{x'y'}/\lambda,$$

and

$$p_{xy} + p_{x'y'} \le p_{xy'}/\lambda + p_{x'y} \cdot \lambda,$$
$$p_{xy} + p_{x'y'} \le p_{xy'} \cdot \lambda + p_{x'y}/\lambda.$$

Proof. We first verify the statement for protocol-compatible ϵ-DP matrices Q. Indeed, by the ϵ-DP condition $q_{xy'}, q_{x'y} \in [q_{xy}/\lambda, q_{x'y'} \cdot \lambda]$ and by protocol-compatible $q_{xy'} \cdot q_{x'y} = q_{xy} \cdot q_{x'y'}$. If the product of two reals is fixed, their sum is maximized when they are most apart, which corresponds exactly to the endpoints of the feasible interval for $q_{xy'}$, $q_{x'y}$. To formalize this, we observe that by simple algebra, the condition

$$q_{xy}/\lambda \le q_{xy'} \le q_{x'y'} \cdot \lambda$$

is equivalent to the quadratic inequality

$$q_{xy'}^2 - (q_{xy}/\lambda + q_{x'y'} \cdot \lambda)q_{xy'} + q_{xy} \cdot q_{x'y'} \le 0,$$

since all probabilities must be non-negative. Rewriting this inequality and dividing by $q_{xy'}$, and using the fact that $q_{x'y} = q_{xy} \cdot q_{x'y'}/q_{xy'}$, we obtain the desired bound:

$$q_{xy'} + q_{x'y} \le q_{xy}/\lambda + q_{x'y'} \cdot \lambda.$$

Moreover, the bound is linear in all q_{xy}, $q_{xy'}$, $q_{x'y}$, $q_{x'y'}$ and holds for all protocol-compatible ϵ-DP matrices. Therefore, it would also hold for the sum of these matrices, and thus for protocol-compatible ϵ-DP functionalities. The other bounds follow similarly and this completes the proof. □

Lastly, we introduce the following definition that relaxes the notion of the protocol-compatible ϵ-DP functionality to allow for a (typically small or negligible) fraction of non-private transcripts.

Definition 3.7. *A $2^n \times 2^n$ matrix P is* protocol-compatible ϵ-DP functionality *if both matrices P and $\mathbf{1} - P - \Delta$ can be expressed as sums of protocol-compatible ϵ-DP matrices, where $\mathbf{1}$ is the all-ones matrix and all entries of Δ are between 0 and δ.*

An analogue of Theorem 3.5 exists for (ϵ, δ)-functionalities defined over *binary* inputs:

Theorem 3.8. *Let π be a randomized (ϵ, δ)-DP two-party protocol defined over $x, y \in \{0,1\}$ and $\pi(x,y)$ be the Boolean output of the protocol. Let P be a matrix of probabilities $p_{xy} = \Pr[\pi(x,y) = 0]$. Then P is a protocol-compatible $(\epsilon + \sqrt{\delta})$-DP $O(\sqrt{\delta})$-close functionality.*

Proof. In the notation of the previous theorem, define the set of "bad" transcripts B as

$$B = \{\tau \colon \exists \text{ adjacent } x, x', y, y' \in \{0,1\}, \text{s.t. } P_\tau(x,y) > e^{\epsilon+\sqrt{\delta}} P_\tau(x',y')\}.$$

We claim that for all $x, y \in \{0,1\}$, the probability that $\Pr[\tau \in B \colon \tau \leftarrow \pi(x,y)] < O(\sqrt{\delta})$. Applying Theorem 3.5, it is sufficient to prove the claim.

For any two pairs of adjacent inputs x, x', y, y' define

$$B_{x,x',y,y'} = \{\tau \colon P_\tau(x,y) > e^{\epsilon+\sqrt{\delta}} P_\tau(x',y')\}. \tag{1}$$

The probability of seeing a transcript from $B_{x,x',y,y'}$ on input (x,y) is less than $O(\sqrt{\delta})$, since by the guarantee of (ϵ, δ)-DP and (1):

$$\begin{aligned}\Pr[\tau \in B_{x,x',y,y'} \colon \tau \leftarrow \pi(x,y)\} &\le e^{\epsilon} \Pr[\tau \in B_{x,x',y,y'} \colon \tau \leftarrow \pi(x',y')\} + \delta \\ &< e^{-\sqrt{\delta}} \Pr[\tau \in B_{x,x',y,y'} \colon \tau \leftarrow \pi(x,y)\} + \delta,\end{aligned}$$

from which a $O(\sqrt{\delta})$ bound on $\Pr[\tau \in B_{x,x',y,y'} \colon \tau \leftarrow \pi(x,y)\}$ follows immediately.

Applying the (ϵ, δ)-DP condition again, we find that

$$\Pr[\tau \in B_{x,x',y,y'} \colon \pi(x'',y'')] = e^{\epsilon} O(\sqrt{\delta})$$

for all $x'', y'' \in \{0,1\}$. Since the event B is the union of all events $B_{x,x',y,y'}$, summing over all pairs of adjacent inputs and assuming that ϵ is constant, we complete the proof. □

The next two sections apply Theorem 3.5 to tight analysis of accuracy of differentially private protocols for computing two Boolean functionalities of two bit inputs: AND and XOR.

3.2 Analysis of the AND Functionality

We first define accuracy of a Boolean functionality for computing the AND of two bit inputs specified as a 2×2 matrix of probabilities. Recall that by convention, the matrix P consists of elements p_{xy} signifying the probability of obtaining output 0 on inputs (x, y).

Definition 3.9 (AND-Accuracy). *Define AND-accuracy of a* 2×2 *matrix* $\left(\begin{smallmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{smallmatrix}\right)$ *as*

$$\mathsf{AND\text{-}Acc}(\left(\begin{smallmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{smallmatrix}\right)) = \min(p_{00}, p_{01}, p_{10}, 1 - p_{11}).$$

Note that this notion is identical to the accuracy defined in Section 2. We prove the following theorem establishing the maximal accuracies achievable by protocol-compatible and arbitrary ϵ-DP functionalities and, in particular, showing that there is a gap between the two quantities.

Theorem 3.10. *For any* $\lambda \geq 1$ *and a* 2×2 *matrix* M *we have the following:*

1. *If* M *is a* ϵ-DP *functionality, then* $\mathsf{AND\text{-}Acc}(M) \leq \frac{\lambda}{1+\lambda}$*, where* $\lambda = e^{\epsilon}$*.*
2. *If* M *is a* ϵ-DP *protocol-compatible functionality, then*

$$\mathsf{AND\text{-}Acc}(M) \leq \frac{\lambda(\lambda^2 + \lambda + 2)}{(1+\lambda)^3}.$$

In both cases the equality can be achieved.

Proof. Let $M = \left(\begin{smallmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{smallmatrix}\right)$ and $a = \mathsf{AND\text{-}Acc}(M)$.

Claim 1. The accuracy condition implies $p_{01} \geq a$ and $1 - p_{11} \geq a$. On the other hand, by the ϵ-DP constraint $p_{01} \leq p_{11} \cdot \lambda$. Put together we have $a/\lambda \leq p_{11} \leq 1 - a$, which implies $a \leq \lambda/(1+\lambda)$.

The following matrix is indeed a ϵ-DP functionality with accuracy $\lambda/(1+\lambda)$:

$$M = \begin{pmatrix} \lambda/(1+\lambda) & \lambda/(1+\lambda) \\ \lambda/(1+\lambda) & 1/(1+\lambda) \end{pmatrix}.$$

Claim 2. The following conditions relate the probabilities p_{00}, p_{01}, p_{10}, p_{11} to each other and to the accuracy parameter a:

$$\begin{aligned} p_{11} &\leq 1 - a \\ p_{01} + p_{10} &\geq 2 \cdot a \\ p_{01} + p_{10} &\leq p_{00}/\lambda + p_{11} \cdot \lambda \\ (1 - p_{01}) + (1 - p_{10}) &\leq (1 - p_{00}) \cdot \lambda + (1 - p_{11})/\lambda. \end{aligned}$$

The first two inequalities are implied by the accuracy requirement, the last two by applying Lemma 3.6 to M and $\left(\begin{smallmatrix} 1-p_{00} & 1-p_{01} \\ 1-p_{10} & 1-p_{11} \end{smallmatrix}\right)$.

By introducing a new variable $q = p_{01}+p_{10}$ and rewriting the above inequalities, we have

$$q \geq 2 \cdot a \tag{2}$$
$$q \leq p_{00}/\lambda + p_{11} \cdot \lambda \tag{3}$$
$$q \geq 2 - (\lambda + 1/\lambda) + p_{00} \cdot \lambda + p_{11}/\lambda. \tag{4}$$

Consider the intersection of two lines bounding the half-planes (2) and (4), where q and p_{00} are considered as free variables and λ, a, and p_{11} are parameters. It is easy to verify that the lines intersect at the point (p^*_{00}, q^*), where

$$p^*_{00} = 1 + 1/\lambda^2 - 2/\lambda + 2a/\lambda - p_{11}/\lambda^2 \quad \text{and} \quad q^* = 2 \cdot a.$$

The following lemma argues that (p^*_{00}, q^*) satisfies (3):

Lemma 3.11. *Let p^*_{00} be defined as above. Then the following holds:*

$$2 \cdot a \leq p^*_{00}/\lambda + p_{11} \cdot \lambda.$$

Proof. Towards a contradiction, assume that

$$2 \cdot a > p^*_{00}/\lambda + p_{11} \cdot \lambda. \tag{5}$$

Consider two cases:

Case $p_{00} \leq p^*_{00}$. Then

$$q \overset{(3)}{\leq} p_{00}/\lambda + p_{11} \cdot \lambda \leq p^*_{00}/\lambda + p_{11} \cdot \lambda \overset{(5)}{<} 2 \cdot a,$$

contradicting (2).

Case $p_{00} > p^*_{00}$. Then

$$(p_{00} - p^*_{00})/\lambda + 2 \cdot a \overset{(5)}{>} p_{00}/\lambda + p_{11} \cdot \lambda \overset{(3)}{\geq} q \overset{(4)}{\geq}$$
$$2 - (\lambda + 1/\lambda) + p_{00} \cdot \lambda + p_{11}/\lambda = (p_{00} - p^*_{00}) \cdot \lambda + p^*_{00} \cdot \lambda + 2$$
$$- (\lambda + 1/\lambda) + p_{11}/\lambda \overset{\text{def of } p^*_{00}}{=} (p_{00} - p^*_{00}) \cdot \lambda + 2 \cdot a,$$

which is a contradiction since $\lambda \geq 1$ and $p_{00} > p^*_{00}$, concluding the proof of the Lemma. □

Finally, by substituting the value of p^*_{00} into the statement of Lemma 3.11 and using that $p_{11} \leq 1 - a$, we have

$$2 \cdot a \leq (1 + 1/\lambda^2 - 2/\lambda + 2a/\lambda - p_{11}/\lambda^2)/\lambda + p_{11} \cdot \lambda$$
$$\leq \lambda + 1/\lambda - 2/\lambda^2 + a \cdot (2/\lambda^2 - \lambda + 1/\lambda^3),$$

from which after collecting like terms and simplifying, the claim $\mathsf{AND\text{-}Acc}(M) = a \leq \lambda(\lambda^2 + \lambda + 2)/(1 + \lambda)^3$ follows.

The protocol with optimal accuracy appears in Appendix thus proving tightness of the bound. □

We remark that Claim 2 of Theorem 3.10 also applies to δ-close ϵ-DP protocol compatible functionalities, with the upper bound on the accuracy increasing by $(2 + \lambda + 1/\lambda)\delta = O(\delta)$. The proof changes in its application of Lemma 3.6 to $\left(\begin{smallmatrix} 1-p_{00} & 1-p_{01} \\ 1-p_{10} & 1-p_{11} \end{smallmatrix}\right)$ that becomes instead $\left(\begin{smallmatrix} 1-p_{00}-\delta_{00} & 1-p_{01}-\delta_{01} \\ 1-p_{10}-\delta_{10} & 1-p_{11}-\delta_{11} \end{smallmatrix}\right)$, where $\delta_{00}, \delta_{01}, \delta_{10}, \delta_{11} \in [0, \delta]$. It is easy to verify that changes in the inequality (4) can be absorbed by reducing the value of a by $(2 + \lambda + 1/\lambda)\delta = O(\delta)$.

Maximal accuracies attained by ϵ-DP functionalities and protocol-compatible ϵ-DP functionalities are shown in Figure 1.

3.3 Analysis of the XOR Functionality

Recall that we consider the worst-case accuracy of a randomized protocol, i.e., the lowest probability over all inputs of producing a correct answer.

Definition 3.12 (XOR-Accuracy). *Define XOR-accuracy of a* 2×2 *matrix* $\left(\begin{smallmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{smallmatrix}\right)$ *as*

$$\mathsf{XOR\text{-}Acc}\left(\left(\begin{smallmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{smallmatrix}\right)\right) = \min(p_{00}, 1 - p_{01}, 1 - p_{10}, p_{11}).$$

Note that this notion is identical to the accuracy defined in Section 2. The following theorem bounds XOR-accuracy of DP functionalities and protocol-compatible DP functionalities.

Theorem 3.13. *For any* $\lambda \geq 1$ *and a* 2×2 *matrix* M *we have the following:*

1. *If* M *is a* ϵ-DP *functionality, then* $\mathsf{XOR\text{-}Acc}(M) \leq \frac{\lambda}{1+\lambda}$.
2. *If* M *is a* ϵ-DP *protocol-compatible functionality, then* $\mathsf{XOR\text{-}Acc}(M) \leq \frac{1+\lambda^2}{(1+\lambda)^2}$.

In both cases the equality can be achieved.

Proof. Let $M = \left(\begin{smallmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{smallmatrix}\right)$ and $a = \mathsf{AND\text{-}Acc}(M)$.

Claim 1. By the accuracy condition $p_{00} \geq a$ and $1 - p_{01} \geq a$. On the other hand, by the ϵ-DP constraint $p_{00} \leq p_{01} \cdot \lambda$. Put together we have $1 - a \geq p_{01} \geq p_{00}/\lambda \geq a/\lambda$, which implies $a \leq \lambda/(1 + \lambda)$.

The following matrix is indeed a ϵ-DP functionality with accuracy $\lambda/(1+\lambda)$:

$$M = \begin{pmatrix} \lambda/(1+\lambda) & 1/(1+\lambda) \\ 1/(1+\lambda) & \lambda/(1+\lambda) \end{pmatrix}.$$

Claim 2. Lemma 3.6 gives the following bounds on the entries of ϵ-DP protocol-compatible matrices:

$$p_{00} + p_{11} \leq p_{10}/\lambda + p_{01} \cdot \lambda,$$
$$p_{00} + p_{11} \leq p_{10} \cdot \lambda + p_{01}/\lambda.$$

Summing the inequalities and dividing by two, we have

$$p_{00} + p_{11} \leq \frac{\lambda^2 + 1}{2\lambda}(p_{10} + p_{01}). \tag{6}$$

Observe that $\mathsf{XOR\text{-}Acc}(M) = \min(p_{00}, 1-p_{01}, 1-p_{10}, p_{11}) \leq \min(\frac{p_{00}+p_{11}}{2}, 1-\frac{p_{10}+p_{01}}{2})$. Denote $\frac{p_{00}+p_{11}}{2}$ by x and $\frac{p_{10}+p_{01}}{2}$ by y, and write

$$\mathsf{XOR\text{-}Acc}(M) = \min(x, 1-y) \overset{(6)}{\leq} \min(x, 1 - \frac{2\lambda}{1+\lambda^2}x),$$

which attains its maximal value when $x = 1 - \frac{2\lambda}{1+\lambda^2}x$. Solving this for x and substituting in the above expression, we prove that

$$\mathsf{XOR\text{-}Acc}(M) \leq \frac{1+\lambda^2}{(1+\lambda)^2}.$$

This value of accuracy for computing the XOR functionality is achieved by the randomized response protocol (see Appendix). □

4 One-Way Functions from CDP

In this paper, we show that one-way functions are implied by the existence of a family of *computationally* differentially private (CDP) two-party protocols that achieve better accuracy than the bounds proven for DP two-party protocols proven in the previous section. We show this by presenting a more general result: we show that if one-way functions *do not exist*, then the existence of a family of CDP protocols imply the existence of DP protocols with only negligible loss in accuracy and privacy.

Definition 4.1. *An infinite family of two-party protocols $\Pi = \{\pi_k\}$ is defined to be an infinite* family of $(\epsilon, \delta = \text{negligible})$-DP protocols achieving accuracy a for a functionality F *if for every constant $c > 0$, there exists an infinite sequence of $\pi_k \in \Pi$ such that each π_k is an $(\epsilon + k^{-c}, \delta = k^{-c})$-DP protocol with accuracy $a - k^{-c}$ for functionality F.*

Proof of the following theorem appears in the full version:

Theorem 4.2. *Suppose that one-way functions do not exist. Then given any infinite family Π of efficient ϵ-IND-CDP two-party protocols achieving accuracy a for a functionality F, it must be that there is an infinite subfamily $\Pi' \subset \Pi$ such that Π' is an infinite family of $(\epsilon, \delta = \text{negligible})$-DP protocols achieving accuracy a for the functionality F.*

Acknowledgements. We thank the anonymous reviewers for their helpful comments.

References

1. Beaver, D.: Perfect privacy for two-party protocols. In: Feigenbaum, J., Merritt, M. (eds.) Proceedings of DIMACS Workshop on Distributed Computing and Cryptology, vol. 2, pp. 65–77. American Mathematical Society (1989)

2. Beimel, A., Nissim, K., Omri, E.: Distributed private data analysis: Simultaneously solving how and what. In: Wagner (ed.) [29], pp. 451–468
3. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In: Simon (ed.) [28], pp. 1–10
4. Canetti, R., Kushilevitz, E., Lindell, Y.: On the limitations of universally composable two-party computation without set-up assumptions. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 68–86. Springer, Heidelberg (2003)
5. Chan, T.-H.H., Shi, E., Song, D.: Optimal lower bound for differentially private multi-party aggregation. In: Epstein, L., Ferragina, P. (eds.) ESA 2012. LNCS, vol. 7501, pp. 277–288. Springer, Heidelberg (2012)
6. Chaum, D., Crépeau, C., Damgard, I.: Multiparty unconditionally secure protocols. In: Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing, STOC 1988, pp. 11–19. ACM, New York (1988), http://doi.acm.org/10.1145/62212.62214
7. Chor, B., Kushilevitz, E.: A zero-one law for Boolean privacy. SIAM J. Discrete Math. 4(1), 36–47 (1991)
8. De, A.: Lower bounds in differential privacy. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 321–338. Springer, Heidelberg (2012)
9. Dinur, I., Nissim, K.: Revealing information while preserving privacy. In: PODS, pp. 202–210. ACM (2003)
10. Dwork, C.: Differential privacy. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. Part II, LNCS, vol. 4052, pp. 1–12. Springer, Heidelberg (2006)
11. Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., Naor, M.: Our data, ourselves: Privacy via distributed noise generation. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 486–503. Springer, Heidelberg (2006)
12. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)
13. Dwork, C., Nissim, K.: Privacy-preserving datamining on vertically partitioned databases. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 528–544. Springer, Heidelberg (2004)
14. Ghosh, A., Roughgarden, T., Sundararajan, M.: Universally utility-maximizing privacy mechanisms. In: Mitzenmacher, M. (ed.) Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, pp. 351–360. ACM (2009)
15. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: Aho, A.V. (ed.) STOC, pp. 218–229. ACM (1987)
16. Haitner, I., Omri, E., Zarosim, H.: On the power of random oracles. IACR Cryptology ePrint Archive 2012, 573 (2012)
17. Hardt, M., Talwar, K.: On the geometry of differential privacy. In: Schulman (ed.) [27], pp. 705–714
18. Harnik, D., Naor, M., Reingold, O., Rosen, A.: Completeness in two-party secure computation: A computational view. J. Cryptology 19(4), 521–552 (2006)
19. Ishai, Y., Prabhakaran, M., Sahai, A.: Founding cryptography on oblivious transfer—efficiently. In: Wagner (ed.) [29], pp. 572–591
20. Kasiviswanathan, S.P., Rudelson, M., Smith, A., Ullman, J.: The price of privately releasing contingency tables and the spectra of random matrices with correlated rows. In: Schulman (ed.) [27], pp. 775–784

21. Kilian, J.: Founding cryptography on oblivious transfer. In: Simon (ed.) [28], pp. 20–31
22. Kilian, J.: A general completeness theorem for two-party games. In: Koutsougeras, C., Vitter, J.S. (eds.) STOC, pp. 553–560. ACM (1991)
23. Kilian, J.: More general completeness theorems for secure two-party computation. In: STOC, pp. 316–324 (2000)
24. Kushilevitz, E.: Privacy and communication complexity. In: FOCS, pp. 416–421 (1989)
25. McGregor, A., Mironov, I., Pitassi, T., Reingold, O., Talwar, K., Vadhan, S.P.: The limits of two-party differential privacy. In: FOCS, pp. 81–90. IEEE Computer Society (2010)
26. Mironov, I., Pandey, O., Reingold, O., Vadhan, S.: Computational differential privacy. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 126–142. Springer, Heidelberg (2009)
27. Schulman, L.J. (ed.): Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, June 5-8. ACM (2010)
28. Simon, J. (ed.): Proceedings of the 20th Annual ACM Symposium on Theory of Computing, Chicago, Illinois, USA, May 2-4. ACM (1988)
29. Wagner, D. (ed.): CRYPTO 2008. LNCS, vol. 5157. Springer, Heidelberg (2008)
30. Yao, A.C.C.: Protocols for secure computations (extended abstract). In: 23rd Annual Symposium on Foundations of Computer Science (FOCS 1982), pp. 160–164. IEEE (1982)

Secure Computation against Adaptive Auxiliary Information

Elette Boyle[1,⋆], Sanjam Garg[2], Abhishek Jain[3,⋆], Yael Tauman Kalai[4], and Amit Sahai[2,⋆⋆]

[1] MIT
eboyle@mit.edu
[2] UCLA
{sanjamg,sahai}@cs.ucla.edu
[3] MIT and Boston University
abhishek@csail.mit.edu
[4] Microsoft Research, New England
yael@microsoft.com

Abstract. We study the problem of secure two-party and multiparty computation (MPC) in a setting where a cheating polynomial-time adversary can corrupt an arbitrary subset of parties and, in addition, learn arbitrary auxiliary information on the *entire states* of all honest parties (including their inputs and random coins), in an *adaptive* manner, *throughout the protocol execution*. We formalize a definition of multiparty computation secure against adaptive auxiliary information (AAI-MPC), that intuitively guarantees that such an adversary *learns no more than the function output and the adaptive auxiliary information*. In particular, if the auxiliary information contains only partial, "noisy," or computationally invertible information on secret inputs, then *only* such information should be revealed.

We construct a universally composable AAI two-party and multiparty computation protocol that realizes any (efficiently computable) functionality against malicious adversaries in the common reference string model, based on the linear assumption over bilinear groups and the n-th residuosity assumption. Apart from theoretical interest, our result has interesting applications to the regime of leakage-resilient cryptography.

At the heart of our construction is a new two-round oblivious transfer protocol secure against *malicious* adversaries who may receive adaptive auxiliary information. This may be of independent interest.

⋆ Supported by NSF Contract CCF-1018064 and DARPA Contract Number: FA8750-11-2-0225.

⋆⋆ Research supported in part from a DARPA/ONR PROCEED award, NSF grants 1228984, 1136174, 1118096, 1065276, 0916574 and 0830803, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, and an Okawa Foundation Research Grant. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0389. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 316–334, 2013.

1 Introduction

Historically, when formalizing security definitions for cryptographic protocols, it was noted that adversarial parties may enter a protocol with relevant knowledge *from the past*. Meaningful security definitions should thus embed the important requirement that, even armed with such side information, an adversary still must not be able to break the security of the protocol. For example, in a zero-knowledge proof system [22], it should not be the case that an adversarial verifier with partial information about a witness, can now suddenly recover the entire witness from the protocol. This natural property was formalized by requiring that, for any adversary with any potential auxiliary input z on the inputs of the honest parties *prior* to the execution of the protocol, this adversary still learns *nothing* beyond the inputs and prescribed outputs of corrupted parties (and, of course, the auxiliary input z it learned prior).

In the last two decades, as cryptographic protocols have become increasingly prevalent (often within everyday online activities, run in parallel), and as new classes of strong attacks have emerged, it has become increasingly evident that adversaries may also acquire auxiliary information on the internal state of honest parties *during the protocol execution*. This may take place, e.g., by performing *physical attacks* on an implementation of a processor (say, a smart card or a hardware token) [31,2,26], or when the randomness used by an honest party in a protocol is correlated with randomness used in other applications. Unfortunately, this case is no longer covered under historical definitions: the moment an adversary is able to learn any side information, say, about the randomness of an honest party in the protocol, the security guarantees break down.

In this work, we seek to extend the standard notion of security with (static) auxiliary inputs to the setting of general *adaptive* auxiliary information. We study secure general two-party and multiparty computation [34,21] in the setting where an adversary, who corrupts an arbitrary subset of parties in the protocol, is able to learn arbitrary (efficiently computable) auxiliary information on the *entire states* of all the honest parties (including their inputs and random coins), in an *adaptive* manner, *throughout the protocol execution*. We formalize a meaningful definition of security within this setting, and construct two-party and multiparty computation protocols satisfying our definition.

How to Define Security against Adaptive Auxiliary Information? Security of MPC protocols is formalized by comparing the real-world protocol execution to an ideal-world experiment where the parties interact directly with a trusted party who receives all parties' inputs and responds only with the correct function output. Formally, an MPC protocol is said to be secure under the classical definition if for every real-world adversary with some auxiliary input (say) z, there exists an ideal-world adversary (a.k.a. simulator) with the same auxiliary input z, who simulates the output of the real-world experiment.

Our goal is to generalize this definition to the setting where side information can be learned during the protocol. We model *adaptive* auxiliary information by allowing the adversary to specify (efficiently computable) functions f_i,

adaptively throughout the protocol. For each such query, the selected function is evaluated on the entire secret states of the honest parties, and the result is given to the adversary as auxiliary information. Intuitively, we wish to guarantee that an adversary who participates in the protocol and receives adaptive auxiliary information throughout the protocol's lifetime still learns nothing beyond the inputs and outputs of the corrupted parties, and the auxiliary information.

Note that it is not immediately apparent how to formalize this notion. Whereas in the static setting both the adversary and the simulator receive the exact same auxiliary input z, in the adaptive setting, it doesn't make sense even syntactically to say that the *same* auxiliary input functions are applied both in the real world and in the ideal world: this is because the real-world auxiliary input function will expect to take as input the secret states of parties executing a protocol, whereas in the ideal world no protocol is ongoing.

A natural attempt to formalize security in this setting may be to require that if the real adversary learns ℓ bits of auxiliary information, then the simulator can also learn at most ℓ bits of auxiliary information. While this is a natural requirement (and our definition will achieve at least this requirement), unfortunately, it may be too weak. For example, the auxiliary information learnt by a real-world adversary (say, via physical processes) may be large but "noisy," giving very little information about the honest parties' inputs. Or, the honest parties' inputs may be information-theoretically determined but *computationally unpredictable* given the real-world auxiliary information. In these cases, the above definition may not provide a meaningful security guarantee since the simulator may be able to simulate the honest parties "trivially" by first learning a large portion (or all!) of their inputs as auxiliary information. Ideally, we would like to formalize the intuitive requirement that the auxiliary information in the ideal world is "no more" than that in the real world.

Our Security Definition. We capture the desired security notion by placing additional restrictions on the ideal-world simulator. In particular, for each auxiliary input function f that the real-world adversary generates, we require that the simulator generates a "translation function" T that takes as input only the secret inputs of the honest parties, and generates "simulated states" for the honest parties at each point in the protocol. These simulated states should be computationally indistinguishable from the real states, and should be consistent with the simulated transcript. Then, for each auxiliary input function f that is requested in the real world, the *same* auxiliary information function is applied in the ideal world, but it is applied to the simulated states. In other words, the ideal world auxiliary function will be the composed function $f \circ T$. This prevents the ideal-world adversary from "learning too much" via the ideal-world auxiliary information: for example, if the requested function f has short output, reveals only useless unused information, or leaves its inputs unpredictable, then the same restrictions will also hold for the ideal-world auxiliary information.

We say that an MPC protocol is *secure against adaptive auxiliary information* if for every PPT real-world adversary who makes arbitrary adaptive (efficiently computable) auxiliary information queries, there exists a PPT ideal-world

simulator who, given the corresponding auxiliary information (as described above), is able to simulate the output of the real-world experiment. Intuitively, this definition guarantees that *the security of the honest parties "gracefully degrades" with the amount of auxiliary information that the real adversary is able to obtain.* We remark that our definition is similar to that of [3,20], occurring within the setting of zero-knowledge and protocols against passive adversaries.

1.1 Our Results

We construct *two-party* and *multiparty* computation protocols (for any efficiently computable function) secure against adaptive auxiliary information in the universal composability (UC) framework [7] in the common reference string (CRS) model. Namely, we prove the following theorem:

Main Theorem (Informal). *For any $n \geq 2$, there exists a UC-secure n-party protocol in the CRS model for evaluating any efficiently computable function, such that for any malicious PPT adversary who (statically) corrupts any subset of parties and learns any amount of (efficiently computable) adaptive auxiliary information Z, this adversary learns nothing beyond the inputs and outputs of corrupted parties, and the same auxiliary information Z (formalized as discussed above). This holds based on the linear assumption over bilinear groups and the n-th residuosity assumption.*[1]

No Bound on the Auxiliary Information. We emphasize that, as in the classical (static auxiliary input) setting, our result does *not* require any a priori bound on the amount of the auxiliary information that the adversary may be able to learn. Instead, our protocol guarantees that for *any* amount of information the real-world adversary is able to (adaptively) acquire throughout the protocol, this "same amount" of auxiliary information is given to the ideal-world simulator, thus providing *graceful degradation* of security. This advantageous property is in contrast with nearly all existing results in leakage-resilient cryptography, which require the user to specify *at design time* an amount of information leakage he wishes to protect against (growing the system parameters accordingly); if an adversary is able to garner more leakage at runtime than the preset bound, then security of these schemes no longer hold.

Application to Leakage-Resilient Protocols. There has been an extensive amount of work on leakage-resilient cryptography in recent years, primarily focused on the setting of *non-interactive* primitives (see e.g., [29,16,1,15,32,30,6,14]). Our result can be used to achieve leakage-resilient *interactive protocols*, an area that has received comparatively little attention (see Section 1.3).

As alluded to above, our MPC protocol directly provides meaningful security guarantees in the setting of leakage: where such a "leaky" adversary learns no more than the inputs and outputs of the corrupted parties, and the leakage

[1] The n-th residuosity assumption can be replaced with any lossy trapdoor function (LTDF) with some specific properties. Roughly speaking, we require LTDFs that are bijective and "sufficiently lossy".

information. This can be seen by viewing the adaptive auxiliary information as joint leakage on the secret states of the honest parties during the protocol execution.[2] Further, when combined with previous work on leakage-resilient cryptography, our result yields applications where *"standard"* security is guaranteed in the face of *bounded* leakage. Below, we discuss two such applications:

- *Leakage-Resilient MPC in the leak-free preprocessing model.* A recent work of Boyle et al. [4] builds upon our results to construct multi-party secure computation protocols to achieve *standard ideal-world security* (where no leakage is allowed in the ideal world) against real-world adversaries that may leak continuously from the secret state of each honest player *separately*, assuming a one-time leak-free preprocessing phase and a large number of parties. At a very high level, they achieve their result by applying our *multi-party* secure computation protocol to the leakage-resilient computation compiler of Goldwasser and Rothblum [23].
- *Leakage-resilient threshold cryptosystems.* In a threshold cryptosystem [13], parties hold shares of a single secret key, and only a quorum of parties can jointly execute the corresponding secret functionality (e.g., decryption). Our MPC protocol, when combined with an underlying leakage-resilient cryptographic primitive (e.g., [1] for public-key encryption), yields a corresponding leakage-resilient threshold cryptosystem. The security guarantee of our protocol implies that any information learned by an adversary who controls any strict subset of a quorum, and obtains leakage on the joint secret states of all honest parties during the collective decryption protocol, reduces to simply the output value and corresponding leakage on the underlying secret key.

1.2 Technical Overview

Our starting point is the GMW paradigm for building MPC protocols [21].

The First Approach. Recall the GMW paradigm begins by designing an MPC protocol secure against semi-honest (i.e., passive) adversaries, and then compiles the protocol into one secure against malicious adversaries by "enforcing" semi-honest behavior via use of zero-knowledge proofs, a commitment scheme, and a coin-tossing protocol.

We begin by mirroring this approach in the setting of adaptive auxiliary input. We directly achieve MPC secure against adaptive auxiliary information in the semi-honest setting by instantiating the basic GMW protocol with the oblivious transfer protocol of [3] (that has analogous semi-honest security properties). Further, continuing onto the GMW compiler, we see that zero-knowledge proofs secure against adaptive auxiliary information were already constructed in [20,3], and *equivocal* commitment schemes [18] were also shown to have the required security properties [20,3]. We are thus almost within reach of our final goal.

[2] We note, however, this differs from the security model considered in [3], where leakage on the state of each party is "disjoint" in both the real- and ideal-world experiments. Indeed, achieving security in such a model is an interesting open problem.

Unfortunately, in the remaining step, one runs into serious problems. Note that in order to reduce the malicious security of the compiled protocol to the semi-honest security of the original protocol, the coin-tossing protocol to be used in the compiler must be *fully simulatable*, in that the simulator must be able to choose the output of the coin toss, and simulate the protocol to force this output, for both honest and corrupted parties. It is not clear how to construct a *coin-tossing* protocol secure against adaptive auxiliary information.[3] Indeed, as shown recently by Chung et al. [11], constructing such a protocol in the two-party setting is in fact *impossible*. We refer the reader to the full version for further discussion on this issue.

A New Stepping Stone: "Semi-malicious" Adversaries. We thus abandon the approach of mimicking the GMW paradigm "out of the box." We instead consider a new intermediate step, lying closer to security against malicious adversaries, with the goal of eliminating the necessity for fully simulatable coin-tossing in the final compiler. This amounts to constructing protocols that remain secure even if an adversary potentially uses "bad" randomness in the protocol execution. To formalize this requirement, we consider the notion of a *semi-malicious* adversary that follows the protocol execution (similar to a semi-honest adversary), but can choose its random coins (and inputs) in any arbitrary manner.[4]

Once we construct a protocol for semi-malicious adversaries (that can learn arbitrary auxiliary information), we can easily compile it into a secure protocol for *malicious* adversaries by standard techniques. We do so using a modified version of the GMW compiler adapted to our setting, implemented with equivocal commitments [18,9] and the UC-NIZKs of [24] that were shown to be secure against adaptive auxiliary information by Garg et al. [20]. (We refer the reader to the technical sections for more details.) The task then remains to construct an MPC protocol that is secure against adaptive auxiliary information in the presence of semi-malicious adversaries.

A close look at the basic GMW construction reveals that constructing semi-malicious MPC reduces to constructing a semi-malicious oblivious transfer (OT) protocol. (We note that this observation is also implicit in [28].) Since our goal is to protect against adversaries who may learn adaptive auxiliary information, we aim to construct OT protocols with similar security guarantees against semi-malicious adversaries. We discuss this next.

Semi-malicious OT. Our starting point is the adaptively secure semi-honest OT protocol of Canetti et al. [9]. The [9] construction follows the "standard template" of [17] for semi-honest OT, but replaces the underlying encryption scheme with a non-committing encryption (NCE) scheme [8]. Namely, (1) The receiver R generates and sends two public keys pk_0, pk_1 for the (non-committing)

[3] The leakage-resilient coin tossing result of [5] is not relevant to this setting. Their construction requires an honest majority of parties (to attain information theoretic guarantees), whereas our model allows an arbitrary number of corruptions.

[4] The notion of semi-malicious adversaries is somewhat similar in spirit to the notion of defensible adversaries considered by [25]. We refer the reader to Section 3 for a comparison of the two notions.

encryption scheme—one for which he knows the secret key, and one "obliviously" sampled; and (2) the sender S sends an encryption of each of his messages m_i, under the corresponding public key pk_i. The [9] scheme was shown to be secure against adaptive auxiliary information in the semi-honest model by [3]. However, the protocol fails in the *semi-malicious* model. Indeed, a semi-malicious receiver can simply choose bad randomness to "obliviously" sample public keys for which he can decrypt (and thus learn both messages of the sender). Further, a semi-malicious sender may be able to create "malformed" ciphertexts that cause the honest receiver to abort depending on his secret input. Circumventing these fatal roadblocks demands a new set of techniques. We solve these problems as follows:

- First, we construct an underlying NCE scheme with strong security properties, which will guarantee security in the OT protocol *as long as the adversary's randomness does not fall within a very small "bad" set*. We achieve this by building an NCE scheme where the public keys generated via the oblivious key generation algorithm are almost always *lossy* (except if they belong to some exponentially small set, such as the set of DDH tuples). Now, unless the adversary's randomness falls within this very small set, the encryption of non-requested messages under his obliviously sampled public keys will *information theoretically* hide the messages.
- Second, we develop a new methodology for generating private randomness that *prevents* a malicious party from choosing randomness within this small bad set. The challenge is doing so *in the presence of adaptive auxiliary information*, and while *simultaneously* providing the simulator the necessary flexibility to "force" any randomness of his choice for honest parties.

  A potential idea is to design a modified coin tossing protocol to ensure a malicious party's output randomness still maintains sufficient entropy in the auxiliary information setting. However, approaches of this kind seem to inherently necessitate an *a priori bound* on how much auxiliary information can be handled: if honest parties cannot hold onto *any* secret entropy during the protocol, this path appears hopeless.

  We provide a different approach. We construct a *non-interactive* randomness generation procedure that achieves the desired properties by use of *lossy trapdoor functions* (LTDF), together with a CRS. Namely, each party P_i is assigned an LTDF seed σ_i in the CRS; each time the party must sample randomness in the protocol, he first chooses a random value r in the LTDF domain, and then uses the LTDF evaluation $F(\sigma_i, r)$ as his protocol randomness. Loosely speaking, in the simulation, honest parties will be assigned seeds for *injective* functions, whereas corrupted parties will be assigned (computationally indistinguishable) seeds for *lossy* functions. This allows the simulator to efficiently "explain" any possible output for honest parties, while simultaneously restricting malicious parties to a small set of attainable output values that does not "hit" the small set of *bad* values. We refer the reader to Section 4.3 for more details.

Final Touches. While the above ideas essentially handle the issue of "bad" randomness, we still need to find a way to answer the auxiliary information

queries of the adversary correctly. Our starting point for this is the observation of [20,3] that adaptive security (without erasures) provides simulators that can simulate random tapes for honest parties, which can be used to answer auxiliary information queries. However, this is possible if the simulator is able to decide its random tape *after* viewing the random tape of the adversary. Unfortunately, depending upon the "structure" of the protocol, this may not always be possible. To address this problem, we somewhat "soften" this asymmetry: instead of generating the entire random tapes of each party a priori, we generate them in an "online" fashion. In part because the GMW protocol provides perfect security in the OT-hybrid model, it turns out that this essentially suffices for simulation.

1.3 Related Work and Organization

The notion of security considered in this paper is somewhat analogous to that considered by Bitansky et al. [3] and Garg et al. [20], in the context of leakage-resilience. Garg *et. al.* [20] consider the case of zero-knowledge proof systems where a malicious verifier can adaptively leak arbitrary information on the state of the honest prover during the execution of the protocol. Bitansky *et. al.* [3] put forth a general definition of leakage tolerance in the setting of two-party interactive protocols, extending the notion of universally composable security. They show how to realize specific tasks (such as oblivious transfer) in such a scenario against semi-honest adversaries as well as zero knowledge proofs in the CRS model. In addition, they prove a composition theorem for leakage-tolerant protocols that we use in our work. Indeed, our work is greatly inspired by theirs.

A concurrent and independent work of Damgård et al. [12] considers the problem of two-party computation protocol in the leakage setting. They formalize a security definition along the lines of entropic leakage as in [32,27] and show how to realize it for NC^1 functionalities against *semi-honest* adversaries. (In contrast, we consider malicious adversaries, as well as the multiparty setting.) We note that our results, cast in the leakage context, also satisfy their definition (for the case of 2-party protocols).

Guide to the Paper. Section 2 contains partial preliminaries. In Section 3, we present our model and security definition. Section 4 contains the technical core of our work: an oblivious transfer protocol secure against adaptive auxiliary information in the semi-malicious model. Due to space limitations, we defer remaining preliminaries, proofs of security, and formal analysis to the full version.

2 Preliminaries

Non-committing Encryption. Informally, a non-committing (bit) encryption scheme [8] is a semantically secure, possibly interactive encryption scheme, with the additional property that a simulator can generate special ciphertexts that can be "opened" to (i.e. demonstrated to be the encryption of) both 0 and 1.

Definition 1. *[8,10] A non-committing (bit) encryption scheme consists of a tuple* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{NCSim})$*, where* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is a semantically secure encryption scheme, and* NCSim *is a* PPT *simulation algorithm that on input* 1^k *outputs a tuple* $(e, c, r_G^0, r_E^0, r_G^1, r_E^1)$ *such that for every* $b \in \{0,1\}$ *the following distributions are computationally indistinguishable:*

1. *The joint view of an honest sender and an honest receiver in a normal encryption of* b*:* $\{(e, c, r_G, r_E) : (e, d) \leftarrow \mathsf{Gen}(1^k; r_G), c \leftarrow \mathsf{Enc}_e(b; r_E)\}$.
2. *A simulated view of an encryption of* b*:* $\{(e, c, r_G^b, r_E^b) : (e, c, r_G^0, r_E^0, r_G^1, r_E^1) \leftarrow \mathsf{NCSim}(1^k)\}$.

Augmented NCE. In our MPC protocol, we will use an "augmented" NCE scheme, satisfying three additional properties.

Oblivious key generation: It should be possible to sample an encryption key without "knowing" a corresponding secret key via a procedure OGen.

Invertible samplability: The key generation and the oblivious key generation algorithms Gen and OGen should be "invertible." That is, given an output that lies in the range of Gen (resp., OGen) that was potentially generated via a *different* algorithm (e.g., NCSim), we can efficiently generate randomness that "explains" the output as being generated via Gen (resp., OGen).

Alternative simulation: In the standard NCE definition, NCSim generates a simulated ciphertext (and randomness values) *together* with an encryption key e. For our purposes, we require a stronger simulation property, where we can generate a simulated ciphertext for a *fixed* encryption key – namely, one that is obliviously sampled by another party.

In this work, we build upon the NCE construction of Choi *et. al* [10] to construct an *augmented* NCE scheme with additional desired properties. See Section 4 for more details.

Lossy Trapdoor Functions (LTDF). A lossy trapdoor function (LTDF) family [33] consists of two computationally indistinguishable families of functions. Functions in one family are injective and can be efficiently inverted using a trapdoor; functions in the other family are "lossy," in that the size of their image is significantly smaller than the size of their domain. We refer the reader to [33,19] for a complete definition of an (m, ℓ)-LTDF family[5] $(G_{\mathsf{Loss}}, G_{\mathsf{Inj}}, S, F, F^{-1})$, with function sampling algorithms $G_{\mathsf{Loss}}, G_{\mathsf{Inj}}$, domain sampling algorithm S, evaluation algorithm F, and inversion algorithm F^{-1}.

For our purposes, we require a strengthened LTDF family, in which the injective functions are *bijections* (i.e., surjective onto their target space), the lossy branches are sufficiently lossy, and the size parameters satisfy certain relations.

Definition 2. *A collection of* (m, ℓ)*-lossy trapdoor functions* $(G_{\mathsf{Loss}}, G_{\mathsf{Inj}}, S, F, F^{-1})$ *is* bijective *and* (D, α)-admissible *if it satisfies the following properties:*

- **Bijective:** *For every seed* σ *produced by* G_{Inj}*, the algorithm* $F(\sigma, \cdot)$ *computes a* bijective *function* $f_\sigma : D_\sigma \to R_\sigma$ *(where* R_σ *is the output space of* f_σ*).*

[5] The parameters (m, ℓ): Each function f_σ in the family has domain size $|D_\sigma| \geq 2^{m-1}$, and each lossy function f_σ has image size at most $|D_\sigma| \cdot 2^{-\ell}$.

– (D, α)-**Admissible:** *The following hold, corresponding to target output size D, where α fraction of the D values are "bad":*
  1. *Efficiently invertible domain sampling: There exists a PPT algorithm S^{-1} such that for each $x \in D_\sigma$, $\{S^{-1}(x)\} \stackrel{s}{\approx} \{r : S(r) = x\}$.*
  2. *Sufficiently large output space: For each seed σ produced by either $G_{\mathsf{Loss}}(1^k)$ or $G_{\mathsf{Inj}}(1^k)$, the size of the output space R_σ satisfies $|R_\sigma| \geq D$.*
  3. *Sufficiently small image of lossy functions: There exists a negligible function $\mu(k)$ such that for every seed σ produced by G_{Loss}, $\alpha \cdot \left(|D_\sigma| \cdot 2^{-\ell(k)}\right) < \mu(k)$. Recall that $|D_\sigma| \cdot 2^{-\ell(k)}$ is an upper bound for the image size of a lossy function with seed σ.*

In the full version, we show that the composite residuosity-based LTDF family construction (with seed-dependent domains) of Freeman et. al. [19] satisfies our required properties (with parameters tailored to our NCE construction):

Theorem 1. *[19] Under the decisional composite residuosity assumption, there exists a* bijective, (D, α)-admissible *collection of LTDFs $(G_{\mathsf{Loss}}, G_{\mathsf{Inj}}, S, F, F^{-1})$ with seed-dependent domains, for the following two choices of (D, α):*
(1) $D = \binom{4k}{k}$. $\alpha = \binom{3k}{k} / \binom{4k}{k}$. (2) $D = 2^k$. $\alpha = 2^{-k/3}$.

3 Our Model

To define security against adaptive auxiliary information, we turn to the real/ideal paradigm. We consider a real-world execution where an adversary, in addition to corrupting a number of parties, can adaptively learn arbitrary auxiliary information on the *joint* secret states of the honest parties, throughout the protocol execution. Following the works on leakage-resilient cryptography, we model this by allowing the adversary to make auxiliary information queries of the form L, where L is the circuit representation of an efficiently computable function. On making such a query, the adversary learns $L(\mathsf{state})$, where state represents the joint secret state of the honest parties. In the ideal world experiment, the ideal world adversary, i.e., the simulator is allowed to request auxiliary information on the inputs of all the parties from the trusted party. Note that this is similar to those considered in [20,3], although these works focused only on the two-party case, whereas we deal with both two-party case and multi-party case. Below, we describe a standalone security definition. (See full version for the UC setting.)

Ideal World. We first define the ideal world experiment, where n parties $P_1, \ldots, P_n$ interact with a trusted party for computing a function f.

- **Inputs:** Each party P_i obtains an input x_i. The adversary is given (initial) auxiliary input z, selects a subset of parties $M \subset \mathcal{P}$ to corrupt, and receives x_i for every $P_i \in M$.
- **Sending inputs to trusted party:** Each honest party P_i sends its input x_i to the trusted party. For each corrupted party $P_i \in M$, the adversary may select any value x'_i and send it to the trusted party.

- **Trusted party computes output:** Let $x'_1, \ldots, x'_n$ be the inputs that were sent to the trusted party. The trusted party computes $f(x'_1, \ldots, x'_n)$.
- **Adversary learns output:** The trusted party first sends $f(x'_1, ..., x'_n)$ to the adversary. The adversary replies with either continue or abort.
- **Honest parties learn output:** If the message is abort, the trusted party sends $\perp$ to all honest parties. If the adversary's message was continue, then the trusted party sends the evaluation $f(x'_1, \ldots, x'_n)$ to all honest parties.[6]
- **Auxiliary information queries on inputs:** The adversary may send (adaptively chosen) auxiliary information queries in the form of efficiently computable functions L_j (described as a circuit). On receiving such a query, the trusted party computes $L_j(x'_1, \ldots, x'_n)$ and returns the output to the adversary. (We in fact, place further restriction on the communication between the adversary and the trusted party w.r.t. the auxiliary information queries; we discuss this in more detail below.)
- **Outputs:** Honest parties each output the message they obtained from the trusted party. Malicious parties may output an arbitrary PPT function of their initial inputs, auxiliary input, and the interaction with trusted party.

Real World. The real-world execution begins by an adversary $\mathcal{A}$ selecting any arbitrary subset $M \subset \mathcal{P}$ of the parties to corrupt. On being corrupted, each party $P_i \in M$ hands over its input to $\mathcal{A}$. The parties $P_1, \ldots, P_n$ now engage in an execution of a real n-party protocol Π (without any trusted third party). The adversary $\mathcal{A}$ sends all messages on behalf of the corrupted parties, and may follow an arbitrary PPT strategy. In contrast, the honest parties follow the instructions of Π. Furthermore, at any point during the protocol execution, the adversary may make auxiliary information queries of the form L and learn $L(\mathsf{state}_{\mathcal{P}\setminus M})$, where $\mathsf{state}_{\mathcal{P}\setminus M}$ denotes the concatenation of the protocol states state_i of each honest party P_i. We allow the adversary to choose the auxiliary information queries adaptively based on all the information that $\mathcal{A}$ received up to that point (including responses to previous such queries). Honest parties have the ability to toss fresh coins at any point in the protocol; these coins are added to the state of that party at the time they are generated. At the conclusion of the protocol execution, each honest party P_i generates its output according to Π. Malicious parties may output an arbitrary PPT function of the view of $\mathcal{A}$.

Security Definition. We now give our formal definition of MPC secure against adaptive auxiliary information. Our definition crucially relies on the notion of *leakage-oblivious simulation* as defined in [20,3]. We recall it below.

Leakage-Oblivious Simulation. Loosely speaking, an ideal world adversary, i.e., a simulator $\mathcal{S}$, is said to be leakage-oblivious if the auxiliary information obtained

[6] We can also define a more general case, where f may output a *different* value $f_i(x'_1, ..., x'_n)$ to each party P_i. In this setting, the adversary first learns the set of outputs $\{f_i(x'_1, ..., x'_n)\}_{i \in M}$ corresponding to corrupted parties, and then decides whether to abort or to allow the honest parties to receive their respective outputs. We do not dwell on this detail for simplicity of exposition; however, our construction also handles this more general case.

by the simulator is used *only* for the purposes of simulating answers to the auxiliary information queries of the real adversary. More formally, we require that the simulator $\mathcal{S}$ has a special subroutine $\widetilde{\mathcal{S}}$ for handling auxiliary information queries. Whenever $\mathcal{S}$ receives an auxiliary information query L from the real world adversary, $\widetilde{\mathcal{S}}$ is invoked to produce a "state translation circuit" T that takes as input the inputs of the honest parties and produces their joint states. Once T is produced, the ideal functionality is queried on the composed circuit $L \circ T$. When the auxiliary information is returned, it is forwarded directly to the real adversary and $\mathcal{S}$ returns to its state prior to the event. Such a simulator is referred to as a leakage-oblivious simulator.

We now define security w.r.t. the real and ideal world experiments as discussed above, except that we consider leakage-oblivious simulators in the ideal world experiment. The output of the ideal-world experiment consists of the inputs and outputs of all parties, and the answers of all the auxiliary information queries. It is denoted by $\mathsf{IDEAL}^{f}_{\mathcal{S},M}(1^k, \boldsymbol{x}, z)$. The overall output of the real-world experiment consists of the inputs and outputs of all parties at the conclusion of the protocol, and all the auxiliary information learnt by the adversary (including the protocol transcript). It is denoted by $\mathsf{REAL}^{\Pi}_{\mathcal{A},M}(1^k, \boldsymbol{x}, z)$.

Definition 3 (MPC Secure against Adaptive Auxiliary Information). *A protocol Π evaluating a functionality f is said to be* secure against adaptive auxiliary information *if for every* PPT *real adversary $\mathcal{A}$, there exists a* PPT *leakage-oblivious simulator $\mathcal{S}$ such that for every input vector $\boldsymbol{x}$, $z \in \{0,1\}^*$, and $M \subset \mathcal{P}$, it holds that,*

$$\left\{\mathsf{IDEAL}^{f}_{\mathcal{S},M}(1^k, \boldsymbol{x}, z)\right\}_{k\in N} \approx_c \left\{\mathsf{REAL}^{\Pi}_{\mathcal{A},M}(1^k, \boldsymbol{x}, z)\right\}_{k\in N}.$$

3.1 Security against Semi-malicious Adversaries

As a stepping stone toward realizing our definition of MPC secure against adaptive auxiliary information in the presence of malicious adversaries, we define the notion of a *semi-malicious* adversary. Intuitively, a semi-malicious adversary is similar to a "standard" (real-world) semi-honest adversary, in that it follows the protocol specification. However, it differs from semi-honest adversaries in that it may choose its input and its "random" coins for any protocol step in an online fashion, adaptively, following any arbitrary PPT strategy. Once it has chosen these values, however, it must follow the protocol as specified, given the chosen input, and using the chosen coins in place of the random coins.[7] Furthermore, in our setting, a semi-malicious adversary is allowed to learn arbitrary auxiliary information on the (joint) secret states of the honest parties.

More formally, a *semi-malicious adversary* $\mathcal{A}$ is modeled as an interactive Turing machine (ITM) which, in addition to the standard tapes, has a special auxiliary tape. At the start of the protocol, $\mathcal{A}$ selects for each corrupted party

[7] This is reminiscent of the notion of defensible adversaries, introduced by Haitner *et. al.* [25]. We refer the reader to the full version for a detailed comparison.

P_k an input x_k (which may depend on the original inputs of corrupted parties), and writes x_k to its special input auxiliary tape. Then in each round of the protocol, whenever $\mathcal{A}$ produces a new protocol message m on behalf of some party P_k, it must also append to its special auxiliary tape *some* randomness that *explains its behavior*. More specifically, all of the protocol messages sent by the adversary on behalf of P_k up to that point, including the new message m, must exactly match the honest protocol specification for P_k when executed with input x_k and randomness r_k written in the special auxiliary tape. We allow $\mathcal{A}$ to make auxiliary information queries on the joint states of the honest parties in the same manner as discussed earlier. We further assume that the adversary is rushing and hence may choose the randomness r in each round adaptively, after seeing the protocol messages of the honest parties in that round (and all prior rounds), as well as all the auxiliary information that it may have obtained so far. Lastly, the adversary may choose to abort the execution on behalf of P_k in any step of the interaction.

Definition 4 (MPC Secure against Adaptive Auxiliary Information in the Semi-malicious Model). *We say that protocol Π evaluating a function f is* secure against adaptive auxiliary information in the semi-malicious model *if it satisfies Def. 3 when we only quantify over all semi-malicious adversaries $\mathcal{A}$.*

4 Semi-malicious OT

In this section, we construct an oblivious transfer (OT) protocol that is secure against adaptive auxiliary information in the presence of semi-malicious adversaries. This protocol is the technical heart of our MPC protocol.

Recall that our OT protocol construction builds on the adaptively secure protocol of [9] (which is resilient to adaptive auxiliary information in the semi-*honest* model [3]), with two primary new components: (1) A new non-committing encryption scheme with strong security properties (guaranteeing security in the OT of [9] as long as the adversary's randomness does not fall in a very small "bad" set), and (2) A new randomness generation procedure that prevents the adversary from selecting randomness within this small "bad" set, in light of adaptive auxiliary information. We construct these new tools in Section 4.1 and 4.2, and present our OT construction itself in Section 4.3.

4.1 Non-committing Encryption with Lossy Keys

We construct an augmented NCE scheme (see Section 2) with the property that public keys generated using the oblivious key generation algorithm are almost always *lossy*. The specific NCE we use is a slight variant of the one due to Choi *et. al.* [10], which makes use of an underlying encryption scheme with oblivious sampling and inverting algorithms (both for generating keys and ciphertexts). For our construction, we use an underlying encryption scheme that is also *lossy*.

Our Underlying Lossy Encryption Scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{OGen}, \mathsf{OEnc}, \mathsf{IGen}, \mathsf{IEnc})$. Let $\mathbb{G}$ be a group of prime order p. The algorithm Gen samples $g_1, g_2 \leftarrow$

$\mathbb{G}$, $u \leftarrow \mathbb{Z}_p$, and outputs $\mathsf{pk} = (g_1, g_2, g_1^u, g_2^u)$ and $\mathsf{sk} = u$. To encrypt a message $m \in \mathbb{G}$ under $\mathsf{pk} = (g_1, g_2, g_3, g_4)$, Enc samples $\beta_1, \beta_2 \leftarrow \mathbb{Z}_p$ and outputs $(g_1^{\beta_1}, g_2^{\beta_2}, g_3^{\beta_1}, g_4^{\beta_2} \cdot m)$. To decrypt a ciphertext (c_1, c_2) with $\mathsf{sk} = u$, Dec outputs $m = \frac{c_1}{c_2^u}$. The oblivious sampling algorithms $\mathsf{OGen}, OEnc$ simply output random values in the appropriate spaces, and the inversion algorithms $\mathsf{IGen}, \mathsf{IEnc}$ to "explain" a given key pair / ciphertext as being obliviously sampled are trivial.

The Augmented NCE scheme $\mathcal{E}_{\mathsf{NCE}}$.

- $\mathsf{NCGen}(1^k)$**:** Generate $4k$ public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_{4k}$ for the underlying scheme, sampling k normally and $3k$ obliviously. Explicitly, choose a random subset $I \subset [4k]$ of size k. For every $i \in I$, sample $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{Gen}(1^k)$, and for every $i \in [4k] \setminus I$, sample $\mathsf{pk}_i \leftarrow \mathsf{OGen}(1^k)$. Also choose two random messages $M_0, M_1 \leftarrow \mathbb{G}$ from the message space of the underlying scheme. Output $\mathsf{pk}_{\mathsf{NCE}} = (\mathsf{pk}_1, \ldots, \mathsf{pk}_{4k}, M_0, M_1)$, and $\mathsf{sk}_{\mathsf{NCE}} = (I, \{\mathsf{sk}_i\}_{i \in I})$.
- $\mathsf{NCEnc}_{\mathsf{pk}_{\mathsf{NCE}}}(b)$**:** Generate $4k$ ciphertexts $c_1, \ldots, c_{4k}$, where k are encryptions of M_b and $3k$ are obliviously sampled. Explicitly, choose a random subset $J \subset [4k]$ of size k. For every $j \in [J]$ compute $c_j \leftarrow \mathsf{Enc}_{\mathsf{pk}_j}(M_b)$, and for every $j \in [4k] \setminus [J]$ compute $c_j \leftarrow \mathsf{OEnc}(1^k)$. Output $c = (c_1, \ldots, c_{4k})$.
- $\mathsf{NCDec}_{\mathsf{sk}_{\mathsf{NCE}}}(c_1, \ldots, c_{4k})$**:** Decrypt the k ciphertexts for which it knows the secret key. Namely, decrypt $\{c_i\}_{i \in I}$. If $\exists c_i$ decrypting to M_b, and no another c_j decrypts to M_{1-b} then output M_b. Otherwise, output $\perp$.
- We refer the reader to [10] for the simulator algorithm NCSim.
- Augmented NCE algorithms: A straightforward modification to NCSim yields the required alternative simulator algorithm NCSim'. Oblivious key generation ONCGen is achieved by running the oblivious key sampler OGen for the underlying scheme $4k$ times and sampling two random messages M_0, M_1. The corresponding inversion algorithm is immediate.

We remark that for our OT application, the choice of M_0, M_1 will be made once overall for all encryptions, and will be contained in the CRS. In the full version, we prove that $\mathcal{E}_{\mathsf{NCE}}$ is an augmented NCE scheme with two additional properties: first, correctness of decryption holds for all but a tiny fraction of encryption randomness; and second, the NCE scheme inherits certain lossy properties of the underlying encryption scheme. These properties are used in the security proof of our MPC protocol.

4.2 Randomness Generation Procedure

In this section, we present a non-interactive procedure for generating private randomness in the CRS model. Intuitively, we need the following two properties:

Semi-malicious P_j cannot force "bad" output: If party P_j is semi-malicious, then he cannot force the output randomness to lie inside a "special" exponentially small subset of the total space $\{0,1\}^t$.
Simulator can retroactively force any output for honest P_j: Given a trapdoor to the CRS, the simulator can retroactively "explain" any desired outcome within the "special" subset of $\{0,1\}^t$ on behalf of an honest P_j.

The "bad" sets arise as follows. To guarantee security of the OT, we must ensure: (1) a corrupted receiver must sample *lossy* public keys in the oblivious sampling procedure, to ensure an honest sender's second message is information-theoretically hidden; (2) a corrupted sender cannot generate malformed ciphertexts, which could cause an honest receiver to abort depending on his secret input bit; and (3) a corrupted sender-receiver pair cannot jointly select key generation and encryption randomness yielding an incorrect message decryption, which would compromise the correctness of the OT output.

We now incorporate the specifics of our NCE construction (from the previous section), and describe the corresponding randomness generation procedures.

Forcing Obliviously Sampled Public Keys to Be Lossy. A corrupted receiver must be restricted to generate only public keys $(g_1, g_2, g_3, g_4) \in \mathbb{G}^4$ in the underlying encryption scheme that are *non-DDH* tuples (yielding lossy keys), whereas the simulator should be able to choose DDH tuples on behalf of honest parties (allowing him to "obliviously sample" keys for which he can decrypt)

To achieve this, we append an additional 4-tuple of random elements $\boldsymbol{g} = (g_1, g_2, g_3, g_4) \in \mathbb{G}^4$ to the CRS of each party. Each time a party P_j must obliviously sample public keys, he does so by rerandomizing his tuple $\boldsymbol{g}_j$: namely, he samples random exponents $\alpha, \beta, \gamma \leftarrow \mathbb{Z}_p^*$ and outputs the tuple $(g_1^{\alpha}, g_2^{\beta}, g_3^{\alpha\cdot\gamma}, g_4^{\beta\cdot\gamma})$ as the desired randomness. Denote this procedure by $\mathsf{DDH\text{-}Rerand}(g_1, g_2, g_3, g_4)$.

Note that if the original tuple (g_1, g_2, g_3, g_4) is a DDH tuple (as will be the case for honest parties in the simulation), then the output of this procedure will be a random DDH tuple; however, if the original tuple is *not* a DDH tuple (as will be the case for corrupted parties), then for *no value* of $\alpha, \beta, \gamma \neq 0$ will the resulting tuple become DDH.

Preventing Malformed Ciphertexts. When encrypting a bit m under the NCE scheme, the sender must select a random subset of positions $J \subset [4k]$ of size k in which to embed encryptions of the appropriate M_m in the underlying scheme. If J is disjoint from an honest receiver's set $I \subset [4k]$ of positions for which he knows the secret keys, then the resulting ciphertext will fail to decrypt. A corrupted sender must then be restricted so that with overwhelming probability over (honestly) chosen random $I \subset [4k]$ with $|I| = k$, it holds that $I \cap J \neq \emptyset$, *even* if he knows I completely. However, on behalf of an honest sender, the simulator should be able to retroactively "explain" arbitrary J values.

To achieve this, our second procedure makes use of a *bijective,* (D, α)*-admissible* lossy trapdoor function (LTDF) family $(G_{\mathsf{Loss}}, G_{\mathsf{Inj}}, S, F, F^{-1})$,[8] for $D = \binom{4k}{k}$ and $\alpha = \binom{3k}{k}/\binom{4k}{k}$, as in Theorem 1(1). Suppose within the CRS each party P_j is associated with a LTDF seed σ_j. At each point when P_j must generate randomness for sampling the set $J \subset [4k]$, he samples a random value $r \leftarrow S(\sigma_j)$ from the domain D_{σ_j} of the LTDF function corresponding to seed σ_j and outputs the evaluation $F(\sigma_j, r)$ as the desired randomness. Denote this procedure by $\mathsf{LTDF\text{-}Sample\text{-}}J(\sigma_j)$.

[8] Such an LTDF family has the additional properties that injective branches are each *bijective*, the output space has size approximately D, and the lossy branches are *very lossy* (to avoid a "bad" set of fractional size α). See Definition 2.

Loosely speaking, the simulation works as follows. For each honest party, the simulator will sample an *injective* seed σ_j together with an inversion trapdoor, allowing him to "explain" any desired randomness output value. In contrast, each corrupted party will be given a *lossy* seed σ_j, whose corresponding function image will be sufficiently small that with high probability it will not hit the small "special" subset of J values which do not intersect the (random) set I.

Preventing Ciphertexts that Decrypt to *Wrong* Messages. When both sender and receiver are corrupted, we must ensure they cannot collaboratively sample a secret key $u \in \mathbb{Z}_p$ and "obliviously" sampled ciphertext $(x, y) \in \mathbb{G}^2$ for the underlying scheme for which $\frac{y}{x^u} = M_{1-m}$. Such a pair may allow an honest encryption of message bit m to (honestly) decrypt to the *wrong* message $(1-m)$.

To prevent this event, the secret key $u \in \mathbb{Z}_p$ and each element of the obliviously generated ciphertext $(x, y) \in \mathbb{G}^2$ will be sampled using the same LTDF-based procedure as above (but with a different choice of parameters (D, α)). Here, the target output space size D will be set to $p = |\mathbb{G}| = 2^k$, and α will be selected so that the *total* collection of possible values of $\frac{y}{x^u}$, over all possible combinations of u, x, y chosen from the range of the the sender and receiver's (lossy) LTDF functions, form a negligible fraction of the entire space of possible values (i.e., the full message space of the underlying encryption scheme). Thus, when the "special" messages M_0, M_1 are chosen randomly as part of the initial CRS, the probability that they will fall into this small set of "bad" attainable messages is negligible. This holds when α is set to $2^{-k/3}$.

More formally, let $(G_{\mathsf{Loss}}, G_{\mathsf{Inj}}, S, F, F^{-1})$ be a bijective, (D, α)-admissible LTDF family for $D = 2^k$ and $\alpha = 2^{-k/3}$, as given in Theorem 1(2). Suppose each party P_j is associated with a LTDF seed σ_j contained in the CRS. Each time party P_j must generate randomness for (1) sampling a secret key u (when acting as receiver in the OT), or (2) selecting each of the two components in an obliviously sampled ciphertext (x, y), he executes the following procedure $\mathsf{LTDF\text{-}Sample\text{-}}\mathbb{G}(\sigma_j)$: first, sample a random value $r \leftarrow S(\sigma_j)$ from the domain D_{σ_j} of the LTDF function corresponding to seed σ_j, output the evaluation $F(\sigma_j, r)$ as the desired randomness.

Combining the Pieces. In Figure 1 we explicitly define the procedures for generating randomness for each relevant application in the OT protocol. The CRS contains a value of $\sigma^{(1)}, \sigma^{(2)}$, and (g_1, g_2, g_3, g_4) for each party, corresponding to LTDF seeds and group elements as described above. Denote by $\mathbf{A}_k$ the set of all subsets of $[4k]$ of size k. We refer the reader to the full version for a formal treatment and analysis of these procedures.

4.3 Our OT Protocol

We now use the augmented NCE scheme from Section 4.1 and the randomness generation procedures from Section 4.2 to construct a 1-out-of-4 OT protocol secure against adaptive auxiliary information in the semi-malicious setting.

$\mathsf{Rand\text{-}KeyGen}(\sigma^{(2)}, (g_1, g_2, g_3, g_4))$**:** Sample $I \leftarrow \mathbf{A}_k$. For each $i \in I$, sample randomness to be used for Gen, by executing $u_i \leftarrow \mathsf{LTDF\text{-}Sample\text{-}}\mathbb{G}(\sigma^{(2)})$. For each $j \in [4k] \setminus I$, sample randomness to be used for *obliviously* sampling a public key, by executing $r_j \leftarrow \mathsf{DDH\text{-}Rerand}(g_1, g_2, g_3, g_4)$. Output $(I, \{u_i\}_{i \in I}, \{r_j\}_{j \in [4k] \setminus I})$.

$\mathsf{Rand\text{-}OblivKeyGen}(g_1, g_2, g_3, g_4)$**:** For each $i \in [4k]$, sample randomness to be used for obliviously sampling a public key, by executing $r_i \leftarrow \mathsf{DDH\text{-}Rerand}(g_1, g_2, g_3, g_4)$. Output $\{r_i\}_{i \in [4k]}$.

$\mathsf{Rand\text{-}Enc}(\sigma^{(1)}, \sigma^{(2)})$**:** Select $J \in \mathbf{A}_k$, by executing $J \leftarrow \mathsf{LTDF\text{-}Sample\text{-}}J(\sigma^{(1)})$. For each $j \in J$, sample encryption randomness $\mathsf{rand}_j \leftarrow \{0,1\}^*$. For each $i \notin J$, sample randomness for obliviously sampling a ciphertext, by running two executions $x_i, y_i \leftarrow \mathsf{LTDF\text{-}Sample\text{-}}\mathbb{G}(\sigma^{(2)})$. Output $(J, \{\mathsf{rand}_j\}_{j \in J}, \{(x_i, y_i)\}_{i \in [4k] \setminus J})$.

Fig. 1. Randomness generation procedures for the semi-malicious OT protocol

Our 1-out-of-4 OT protocol for semi-malicious parties.
CRS: Uniformly random message $M_0, M_1 \leftarrow \mathbb{G}$. For each party P_i: (injective) LTDF seeds $\sigma_i^{(1)}, \sigma_i^{(2)}$, and (non-DDH) tuple $\boldsymbol{g}^i = (g_1^i, g_2^i, g_3^i, g_4^i)$.

1. The receiver P_i, on input $b \in [4]$, does the following:
   1. For $b \in [4]$, sample a standard NCE key pair (e_b, d_b) via $\mathsf{Rand\text{-}KeyGen}(\sigma_i^{(2)}, \boldsymbol{g}^i)$. For each $b' \in [4] \setminus \{b\}$, obliviously sample an NCE public key $e_{b'}$ via the algorithm $\mathsf{Rand\text{-}OblivKeyGen}(\boldsymbol{g}^i)$.
   2. The receiver sends $(e_1, \ldots, e_4)$ to the sender.
3. The sender P_j, on input $m_1, \ldots, m_4$, where each $m_i \in \mathbb{G}$, and upon receiving the message $(e_1, \ldots, e_4)$ from the receiver, does the following:
   1. For each $b' \in [4]$, encrypt the message $m_{b'}$ under public key $e_{b'}$. This is done by executing the encryption algorithm NCEnc with message $m_{b'}$, key $e_{b'}$, and randomness $r_{b'} \leftarrow \mathsf{Rand\text{-}Enc}(\sigma_j^{(1)}, \sigma_j^{(2)})$.
   2. Send $(c_1, \ldots, c_4)$.
3. The receiver decrypts c_b using the secret key d_b.

References

1. Akavia, A., Goldwasser, S., Vaikuntanathan, V.: Simultaneous hardcore bits and cryptography against memory attacks. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 474–495. Springer, Heidelberg (2009)
2. Anderson, R., Kuhn, M.: Tamper resistance: a cautionary note. In: WOEC 1996: Proceedings of the 2nd Conference on Proceedings of the Second USENIX Workshop on Electronic Commerce, pp. 1–11 (1996)
3. Bitansky, N., Canetti, R., Halevi, S.: Leakage-tolerant interactive protocols. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 266–284. Springer, Heidelberg (2012)
4. Boyle, E., Goldwasser, S., Jain, A., Kalai, Y.: Multiparty computation secure against continual memory leakage. In: STOC (2012)
5. Boyle, E., Goldwasser, S., Kalai, Y.T.: Leakage-resilient coin tossing. In: Peleg, D. (ed.) DISC. LNCS, vol. 6950, pp. 181–196. Springer, Heidelberg (2011)

6. Brakerski, Z., Kalai, Y.T., Katz, J., Vaikuntanathan, V.: Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In: FOCS, pp. 501–510 (2010)
7. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067 (2005)
8. Canetti, R., Feige, U., Goldreich, O., Naor, M.: Adaptively secure multi-party computation. In: STOC, pp. 639–648 (1996)
9. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: STOC, pp. 494–503 (2002)
10. Choi, S.G., Dachman-Soled, D., Malkin, T., Wee, H.: Improved non-committing encryption with applications to adaptively secure protocols. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 287–302. Springer, Heidelberg (2009)
11. Chung, K.M., Lin, H., Liu, F.H., Pass, R., Zhou, H.S.: Physically-aware composability (manuscript, 2013)
12. Damgard, I., Hazay, C., Patra, A.: Leakage resilient two-party computation. Cryptology ePrint Archive, Report 2011/256 (2011)
13. Desmedt, Y.G., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, Heidelberg (1990)
14. Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D.: Efficient public-key cryptography in the presence of key leakage. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 613–631. Springer, Heidelberg (2010)
15. Dodis, Y., Kalai, Y.T., Lovett, S.: On cryptography with auxiliary input. In: STOC, pp. 621–630 (2009)
16. Dziembowski, S., Pietrzak, K.: Leakage-resilient cryptography. In: FOCS (2008)
17. Even, S., Goldreich, O., Lempel, A.: A randomized protocol for signing contracts. Commun. ACM 28(6) (1985)
18. Feige, U., Shamir, A.: Zero knowledge proofs of knowledge in two rounds. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 526–544. Springer, Heidelberg (1990)
19. Freeman, D.M., Goldreich, O., Kiltz, E., Rosen, A., Segev, G.: More constructions of lossy and correlation-secure trapdoor functions. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 279–295. Springer, Heidelberg (2010)
20. Garg, S., Jain, A., Sahai, A.: Leakage-resilient zero knowledge. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 297–315. Springer, Heidelberg (2011)
21. Goldreich, O., Micali, S., Wigderson, A.: How to play ANY mental game. In: ACM (ed.) Proc. 19th STOC, pp. 218–229 (1987)
22. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems. In: Proc. 17th STOC, pp. 291–304 (1985)
23. Goldwasser, S., Rothblum, G.: How to compute in the presence of leakage. In: FOCS (2012)
24. Groth, J., Ostrovsky, R., Sahai, A.: Non-interactive zaps and new techniques for NIZK. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 97–111. Springer, Heidelberg (2006)
25. Haitner, I., Ishai, Y., Kushilevitz, E., Lindell, Y., Petrank, E.: Black-box constructions of protocols for secure computation. SIAM J. Comput. 40(2) (2011)
26. Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W.: Lest we remember: Cold boot attacks on encryption keys. In: USENIX Security Symposium, pp. 45–60 (2008)
27. Halevi, S., Lin, H.: After-the-fact leakage in public-key encryption. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 107–124. Springer, Heidelberg (2011)

28. Ishai, Y., Prabhakaran, M., Sahai, A.: Founding cryptography on oblivious transfer – efficiently. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 572–591. Springer, Heidelberg (2008)
29. Ishai, Y., Sahai, A., Wagner, D.: Private circuits: Securing hardware against probing attacks. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 463–481. Springer, Heidelberg (2003)
30. Katz, J., Vaikuntanathan, V.: Signature schemes with bounded leakage resilience. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 703–720. Springer, Heidelberg (2009)
31. Kocher, P.C.: Timing attacks on implementations of diffie-hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
32. Naor, M., Segev, G.: Public-key cryptosystems resilient to key leakage. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 18–35. Springer, Heidelberg (2009)
33. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. SIAM J. Comput. 40(6) (2011)
34. Yao, A.C.: How to generate and exchange secrets. In: Proc. 27th FOCS (1986)

Leakage-Resilient Symmetric Cryptography under Empirically Verifiable Assumptions

François-Xavier Standaert[1], Olivier Pereira[1], and Yu Yu[2]

[1] ICTEAM/ELEN/Crypto Group, Université Catholique de Louvain, Belgium
[2] East China Normal University and Tsinghua University, China

Abstract. Leakage-resilient cryptography aims at formally proving the security of cryptographic implementations against large classes of side-channel adversaries. One important challenge for such an approach to be relevant is to adequately connect the formal models used in the proofs with the practice of side-channel attacks. It raises the fundamental problem of finding reasonable restrictions of the leakage functions that can be empirically verified by evaluation laboratories. In this paper, we first argue that the previous "bounded leakage" requirements used in leakage-resilient cryptography are hard to fulfill by hardware engineers. We then introduce a new, more realistic and empirically verifiable assumption of simulatable leakage, under which security proofs in the standard model can be obtained. We finally illustrate our claims by analyzing the physical security of an efficient pseudorandom generator (for which security could only be proven under a random oracle based assumption so far). These positive results come at the cost of (algorithm-level) specialization, as our new assumption is specifically defined for block ciphers. Nevertheless, since block ciphers are the main building block of many leakage-resilient cryptographic primitives, our results also open the way towards more realistic constructions and proofs for other pseudorandom objects.

Introduction

Physical cryptanalysis is an important concern for cryptographic implementations. By allowing to circumvent the models in which standard security proofs are obtained, it can lead to powerful attacks (e.g. key recoveries) against large classes of devices. Following the publications of papers about side-channel [24], fault [6] or cold boot attacks [16], a large body of research has investigated solutions to mitigate these security breaches. For this purpose, a natural solution is to add protection mechanisms directly at the hardware level (i.e. independent of the algorithm implemented). Examples of such approaches include masking and hiding against side-channel attacks [26], error-detection codes against fault attacks [20], and their formal extensions as compilers (e.g. [18,19]) - leading to contrasted observations. On the one hand, these countermeasures are useful as they reduce the amount of information leakage provided by physical implementations. On the other hand, they usually imply significant performance overheads, and the security they provide is highly dependent on technological assumptions (that may turn out to be contradicted in practice). Over the years, and starting

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 335–352, 2013.

with the seminal work of Micali and Reyzin [27], the question whether a complementary approach exploiting the formalism of modern cryptography could be used in order to improve physical security consequently triggered the interest of many researchers. In other words, can we design new cryptographic constructions and security models in which the guarantees of provable security can be extended from mathematical objects towards physical ones? And are the results obtained in these models practically relevant (in terms of performance and security)?

Related Work. A look at the recent literature suggests that a wide variety of tools aiming at reflecting different classes of physical attacks exist, ranging from quite abstract to more realistic, and for various types of cryptographic primitives. For example, the bounded retrieval model captures an hypothetical situation in which the total amount of information leaked through the execution of a cryptographic primitive is bounded [1,3]. One important drawback of this abstraction is that quantifying an "overall amount of leakage" is hard for hardware engineers. Besides, if a system is being used continually for a sufficiently long period of time, the amount of leakage observed by the attacker may exceed any a-priori determined leakage bound. As a result, alternative models have been proposed, assuming that the leakage rate is bounded and leaving the overall leakage arbitrarily large, e.g. Dziembowski and Pietrzak's leakage-resilient cryptography [12]. These models have been applied for analyzing different cryptographic primitives, including PRGs and stream ciphers [13,30,38,39], PRFs and PRPs [11,13,38], signature schemes (e.g. [7,21]) and public-key encryption (e.g. [2,22]).

Are We Done? Not Really. Despite significant progresses and many clever design ideas, the fundamental problem of formal approaches to physical security remains to determine reasonable restrictions of the leakage function. Even taking the simple(st) example of leakage-resilient PRGs (that will be our main concern in this work), obtaining security proofs in the standard cryptographic setting turns out to be surprisingly difficult [12,13,30,38,39]. Intuitively, the proofs obtained so far require a combination of seemingly too weak assumptions (e.g. that the leakage may come from any polynomial time function) and seemingly too strong assumptions (e.g. that the information leakage is bounded in a somewhat unrealistic manner) [35]. Consequences of this imperfect modeling are threefold. First, it implies design tweaks that seem motivated by proof artifacts more than physical intuition, and consequently harm performances. Second, obtaining the proofs requires intricate (though sometimes of independent interest) mathematical tools, usually leading to loose security bounds. Third and most importantly, it leaves the question of how to connect the results in leakage-resilient cryptography with the practice of side-channel attacks essentially open.

Our Contribution. In this paper, we start by investigating the relevance of different bounded leakage assumptions. In particular, we confront the notion of HILL pseudoentropy used to prove the leakage-resilience of previous PRGs, PRFs and PRPs to the operation of actual side-channel attacks, and argue that it is hard to verify empirically. We then tackle our main problem, i.e. the construction of a leakage-resilient PRG based exclusively on empirically verifiable

assumptions. For this purpose, our central ingredient is the introduction of a specialized assumption of simulatable leakage. We first show that this requirement is easier to guarantee than maintaining a high pseudoentropy in a leaking device, and detail how it can be tested in actual security laboratories. Next, we show the security of an efficient leakage-resilient PRG under simulatable leakage. Eventually, we put forward that our new modeling allows mitigating the three issues listed in the previous paragraph. In particular, it allows major simplifications of the proofs, with reductions directly connected to our physical assumption (i.e. the quality of the simulator). From a methodological point of view, the idea of specialized assumption that we introduce can be seen as an intermediate path, between fully generic requirements (e.g. bounded leakage that applies to any algorithm) and implementation-specific ones (e.g. as used in hardware-level countermeasures). More precisely, our simulatable leakage assumption is specialized at the algorithm level, and applies to any block cipher. We believe this intermediate path is interesting, as it allows a better connection between the theory and practice of side-channel attacks. It is also general enough for being potentially applicable to the many other symmetric cryptographic primitives.

1 Previous Leakage Assumptions

In this section, we analyze different assumptions that have been introduced in previous works, in order to bound the informativeness of a leakage function. For this purpose, we start by providing a description of leakage traces, as they are obtained from the power consumption or electromagnetic radiation of actual cryptographic devices. We then argue that assuming a leakage function with bounded range, or assuming that the secrets manipulated by a leaking device have high pseudoentropy, is hardly realistic. By contrast, we list a few alternative assumptions that are more in line with what hardware designers try to guarantee, based on unpredictable cipher outputs or hard-to-invert leakage functions.

1.1 Actual Leakage Traces

We will consider the AES Rijndael as a case study. Note however that the observations in this section hold for any block cipher. In this context, let us denote the encryption of a plaintext x under a key k giving rise to a leakage trace $\mathbf{l}$ as $y = \mathsf{AES}_k(x) \rightsquigarrow \mathbf{l}$. Since the AES is made of ten rounds, we further denote the application of these rounds and their corresponding subtraces as $x_{i+1} = \mathsf{R}_{k_i}(x_i) \rightsquigarrow l_{i+1}$, where the initial state is given by the plaintext $x_0 = x$ and the ciphertext is given by the final state $y = x_{10}$. That is, a full leakage trace is a vector containing all the rounds subtraces: $\mathbf{l} = [l_0, l_1, l_2, \ldots, l_9, l_{10}]$. For illustration, we provide a leakage trace obtained from a hardware implementation of the AES in Figure 1, where each sample can be written as $l_i(t) = \mathsf{L}_{i,t}(k, x, \rho)$, with ρ a parameter representing the physical randomness (aka noise) in the measurements [27]. In practice, it frequently turns out that the leakage produced when generating an intermediate state x_i can be approximated by the sum of a polynomial function of the bits of x_i (denoted as $x_i[j]$) and some noise [32]:

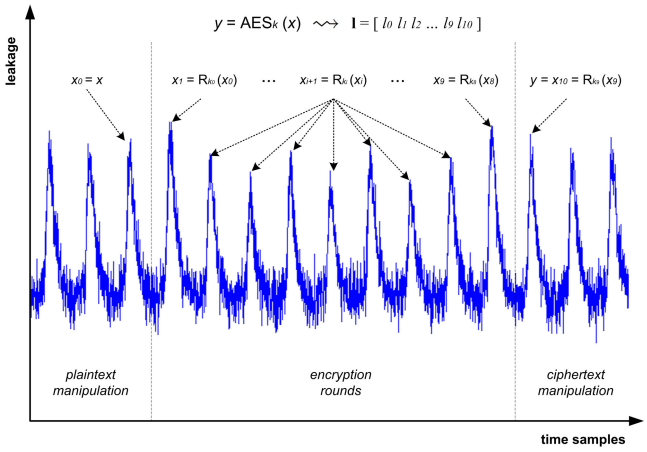


Fig. 1. Exemplary leakage trace of an AES encryption

$$l_i(t) = \mathsf{L}_{i,t}(k, x, \rho) \approx \sum_j \alpha_j \; x_i[j] + \sum_{j_1 \neq j_2} \beta_{j_1,j_2} \; x_i[j_1] \; x_i[j_2] + \ldots + \rho_{i,t}. \quad (1)$$

In the following, and in order to simplify our discussions, we will further assume that each subtrace is made of a single sample (pointed by the arrows in the figure) that can be written as $l_i(k, x) = \mathsf{HW}(x_i) + r_i^{\mathrm{ph}}$, with HW the Hamming weight function and r_i^{ph} a Gaussian distributed physical noise. Note that these are usual assumptions is side-channel attacks [26]. Yet, also keep in mind that we can only loose information by doing this, and that actual adversaries may be more powerful. Summarizing, we will consider illustrative leakage traces defined as:

$$\mathbf{l} = [\mathsf{HW}(x_0) \; \mathsf{HW}(x_1) \ldots \mathsf{HW}(x_{10})] + [r_0^{\mathrm{ph}} \; r_1^{\mathrm{ph}} \ldots r_{10}^{\mathrm{ph}}]. \quad (2)$$

1.2 Bounded Range and HILL Pseudoentropy

One of the most demanding assumption regarding the informativeness of the leakage function is the requirement that its range is bounded to $\{0,1\}^\lambda$, for some parameter λ. Taking the example of a leaking block cipher implementation as in the previous subsection, it is easy to observe that a bounded range is hardly obtained. Starting with our simplifying Hamming weight assumption and considering an n-bit key, we already have that each of the N_r Hamming weights in the trace has range $\approx \log(n)$, leading to an output range proportional to $N_r \cdot \log(n)$ (with N_r the number of block cipher rounds). Then, keeping in mind that the number of samples monitored by an oscilloscope in actual attacks is much larger than N_r, it turns out that the range of the leakage function is frequently larger than $\{0,1\}^n$. In practice, this large range is directly reflected by the memory requirements needed to store the measurements. For example, in a recent work from Eurocrypt 2012, Moradi acquired 200 000 traces, each of

them corresponding to 1μs of power consumption leakage sampled at roughly 10^9 samples per second, i.e. leading to more than 1.5 Gigabits of data storage [28].

Previous works in leakage-resilient cryptography (e.g. [11,12,13,30,39]), it is argued that the bounded range assumption can be relaxed. Loosely speaking, these previous proofs only require that for every key update $k_{j+1} = \mathsf{AES}_{k_j}(x) \rightsquigarrow \mathbf{l}$, the leakage $\mathbf{l}$ does not decrease the HILL pseudoentropy of the updated state k_{j+1} by more than a bounded amount of bits. It is further claimed in [22] that such a requirement seems much more realistic in practice. Unfortunately, a look at our example suggests the opposite. Having a pseudoentropy of $n - \lambda$ bits for k_{j+1} requires that there exist a dense set of $2^{n-\lambda}$ keys $\tilde{k}$ that no efficient distinguisher is able to tell apart from k_{j+1} given $\mathbf{l}$. But again considering that the leakage trace contains a sequence of (pseudorandom) Hamming weights, the number of keys $\tilde{k}$ that give rise to the correct sequence of Hamming weights rapidly vanishes, roughly decreasing the pseudoentropy of k_{j+1} according to $n - N_r \cdot \log(n)$. Of course, the high pseudoentropy requirement is weaker than the bounded range assumption. For example, having multiple correlated samples in the traces would not significantly decrease the pseudoentropy of k_{j+1}, while it would increase the output range of the leakage function. Yet, falsifying the pseudoentropy assumption simply requires that an adversary can check whether the trace $\mathbf{l}$ is consistent with the actual k_{j+1}, allowing him to efficiently distinguish it from most fake $\tilde{k}$'s.

Summarizing, while these simple examples exclude the additional randomness due to physical noise, they clearly suggest that maintaining high pseudoentropy in a leaking device is challenging. Interestingly, this observation nicely connects with the conclusions of Micali and Reyzin, who showed the non-equivalence between unpredictability and indistinguishability in physically observable cryptography [27]. We argue in the next subsections that also in terms of practical assumptions, unpredictability is arguably easier to guarantee.

1.3 Side-Channel Attacks

Most distinguishers published in the literature are based on a divide-and-conquer strategy, where independent pieces of a masker key (denoted as subkeys) are recovered independently. Examples include Kocher et al.'s differential power analysis [24], Gandolfi et al.'s electromagnetic analysis [14], Chari et al.'s template attacks [9], Brier et al.'s correlation power analysis [8], Schindler et al.'s stochastic approach [32], Gierlichs et al.'s mutual information analysis [15] and many variations. These attacks are "standard DPAs" in the sense defined by Mangard et al. [25], and operate according to the three following steps:

1. *Prediction.* The adversary predicts subkey-dependent intermediate values manipulated during the encryption process (e.g. a 1^{st}-round S-box output).
2. *Modeling.* The adversary models the leakage corresponding to these intermediate values (e.g. assuming it depends on the HW of the manipulated data).
3. *Comparison.* The adversary compares the subkey-dependent models with actual measurements (e.g. with Pearson's correlation coefficient). If the attack is successful, the best comparison holds for the correct subkey candidate.

The result of a standard DPA attack against the AES usually corresponds to 16 lists of 256 scores (typically proportional to subkey likelihoods), that are then recombined to obtain a master key candidate, e.g. using key enumeration [36].

One consequence of this description is that actual adversaries are usually not able to exploit all the leakage samples in a trace. In practice, only the intermediate computations that can be guessed will be useful. Taking our AES example again, it means that out of a vector $\mathbf{l} = [l_0, l_1, l_2, \ldots, l_9, l_{10}]$, only the external rounds are exploited (i.e. before the diffusion is complete). Furthermore, considering an attack exploiting the first-round leakage l_1, and under our current assumption that the AES is implemented in 10 clock cycles, we have:

$$l_1 = \mathsf{HW}(x_1) + r_1^{\mathrm{ph}} = \mathsf{HW}(x_1[0]) + \mathsf{HW}(x_1[1]) + \ldots + \mathsf{HW}(x_1[15]) + r_1^{\mathrm{ph}}, \quad (3)$$

where $x_1[i]$ denotes the ith byte of x_1. But actual adversaries are not able to guess all the 16 bytes of x_1 at once either. As a result, a part of this information is usually considered as "algorithmic noise". That is, in a (usual) attack where the 16 AES key bytes are targeted independently, the leakage sample l_1 as seen by the adversary can be rewritten as:

$$l_1^{\mathrm{adv}}[0] = \mathsf{HW}(x_1[0]) + \underbrace{\mathsf{HW}(r_1) + \ldots + \mathsf{HW}(r_{15})}_{\text{algorithmic noise}} + r_1^{\mathrm{ph}}, \quad (4)$$

when targeting the first key byte, whith the r_i's uniformly random unknown bytes (a similar equation holds for all the key bytes). In other words, only a single (or at most a couple of) byte Hamming weight(s) is (are) actually considered as useful signal at a time in this computationally bounded setting.

1.4 One-Way and Seed-Preserving Leakage Functions, Unpredictability

The previous description allows shedding another light on why ensuring high pseudoentropy for cryptographic keys in leaking devices is challenging. The main issue is that it requires that these keys remain difficult to distinguish in front of an adversary who can predict the whole device state (hence, exploit the full vector of Equation 2 rather than the noisy samples of Equation 4). In fact, this task is arguably more difficult than the (already difficult) task of securing an implementation against standard DPA attacks. Therefore, we can at least claim that constructions that strictly need this assumption to hold for being secure are not going to "help hardware designers", as usually advertised by leakage-resilient cryptography. This observation naturally provides a strong incentive to look at alternative assumptions that could be easier to fulfill and evaluate.

In general, a weaker assumption than the high HILL pseudoentropy requirement is that the leakage function is hard-to-invert, or that the key/seed is computationally infeasible to predict given the leakage (see [4,17] for relations between several forms of pseudoentropy). This is easily seen the minimal assumption

since no security is possible if the adversary can recover the key/seed. It is also directly connected to the practice of side-channel attacks that usually aim to predict keys/seeds. Unfortunately, how to build leakage-resilient symmetric cryptographic primitives under such assumptions remains an open problem. So far, only some weaker forms of security results have been obtained in this case, such as the encryption schemes of [10] in the auxiliary-input setting (based on a non-standard lattice problem), and the leakage-resilient stream cipher in [39] (assuming PRGs to behave as random oracles that the leakage functions cannot access). In view of this state-of-the-art, another natural solution would be to use the simulation paradigm. Namely, to argue that some information reveals nothing substantial, it suffices to show that it can be efficiently simulated from some other information that is already part of the adversary's knowledge. This approach is empirically verifiable since it challenges the designer to build such a simulator, and the adversary to break the indistinguishability game. In the next sections, we argue that in the context of block ciphers, simulatable leakage is at least easier to guarantee than high pseudoentropy - and that efficient leakage-resilient PRGs can be proven secure under this assumption.

2 Simulatable Leakage

Concretely, we will study the physical security of a "natural" (i.e. conform to engineering intuition) PRG relying on the iterative application of a length-doubling 2PRG, represented in the left part of Figure 2 (the iterative application of length-$q$pling generator qPRG would allow improved efficiency at the cost of more physical information leakage, and relies on similar security proofs). Furthermore, we will focus on the block cipher based instantiation of 2PRG represented in the right part of the figure, where p_0 and p_1 are public constants (larger expansion factors q's are directly obtained by encrypting more p_i's). The (leakage-free) security of this PRG is easily seen by a hybrid argument. It enjoys many advantages such as simplicity, efficiency and forward security (see more discussions in [5]). From a physical security point of view, it also avoids the alternating structure and large randomness requirements of previously published proposals [13,30,38]. However, it turns out to be extremely difficult to prove the leakage-resilience of this construction in a standard setting (independent of its instantiation).

In order to obtain practically-relevant proofs of leakage-resilience, we want our assumption to be local (i.e. focusing on a single iteration), and re-usable. The second condition suggests to consider block cipher implementations for this purpose. On one hand, they are among the work horses of today's secure communications [23]. On the other hand, they are frequent targets of side-channel analysis, with a vast literature on attacks and countermeasures - making them natural candidates for mitigating the instantiation issues raised in [33]. In the rest of this section, we will consequently define the simulatable leakage assumption for block ciphers (denoted as $\mathsf{BC} : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$ from now on), and argue about its empirical verifiability. The next section will then show how to use this assumption to prove the leakage-resilience of the PRG from Figure 2.

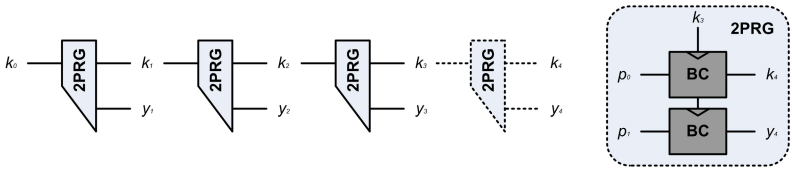


Fig. 2. Left: leakage-resilient PRG. Right: 2PRG instantiation with block ciphers.

2.1 Formal Definition

As discussed in Section 1.1, actual leakage traces are made of multiple samples, each of them being the output of a leakage function. Yet, since our goal is to define our assumptions in general terms, this section will take advantage of a slightly more concise notation that is independent of the actual representation of these traces. That is, we will denote the probabilistic leakage corresponding to a block cipher execution as: $y = \mathsf{BC}_k(x) \rightsquigarrow \mathbf{l} \stackrel{\mathsf{def}}{=} \mathsf{L}(k, x)$, with L the (global) leakage function (i.e. including all the samples).

In practice, we do not know how to express this function as a circuit or a program that a computer could evaluate: we can only sample it by taking leakage measurements from the target circuit on given inputs. Leakages resulting from a complex physical process, it is even unclear how efficiently a Turing machine could compute them. For this reason, they will be available through queries to a public oracle in our model, and our complexity measures will take the number of these queries into account: an (s, t)-bounded adversary $\mathcal{A}^{\mathsf{L}}$ will do at most s queries to L and run in time at most t. Note that we define the leakage oracle as stateless, to capture the usual situation in side-channel attacks where leakages only depend on the current state of the target device and some independent randomness. Using this notation, the requirement we make on a block cipher implementation is that the leakages are *simulatable*. That is, we require that a (stateless) leakage simulator oracle $\mathcal{S}^{\mathsf{L}}(\cdot, \cdot, \cdot)$ can be built, possibly relying on accessing the implementation and measuring equipment producing the real leakages. It must be able to return a simulated leakage corresponding to any (possibly inconsistent) key, plaintext and ciphertext, and its responses must be such that no efficient adversary $\mathcal{A}$ can guess the bit b in the following q-sim game except with a small advantage.

In this game, the adversary can query the device for the encryption of q values of his choice. If $b = 0$, he receives the encryption of his queries and the corresponding real leakages. If $b = 1$, he receives the encryption of his queries and simulated leakages, based on the plaintext and ciphertext, but ignoring the (real) key k that was used to compute them, which is replaced by another random k^*

Game q-$\mathsf{sim}(\mathcal{A}, \mathsf{BC}, \mathsf{L}, \mathcal{S}, b)$.		
The challenger selects two random keys k and k^* in $\{0,1\}^n$. The output of the game is a bit b' computed by $\mathcal{A}^{\mathsf{L}}$ based on the challenger responses to a total of at most q adversarial queries of the following type:		
Query	Response if $b = 0$	Response if $b = 1$
$\mathsf{Enc}(x)$	$\mathsf{BC}_k(x)$, $\mathsf{L}(k, x)$	$\mathsf{BC}_k(x)$, $\mathcal{S}^{\mathsf{L}}(k^*, x, \mathsf{BC}_k(x))$
and one query of the following type:		
Query	Response if $b = 0$	Response if $b = 1$
$\mathsf{Gen}(z, x)$	$\mathcal{S}^{\mathsf{L}}(z, x, k)$	$\mathcal{S}^{\mathsf{L}}(z, x, k^*)$

in an invocation of $\mathcal{S}^{\mathsf{L}}(\cdot, \cdot, \cdot)$. Independently of these encryption queries, $\mathcal{A}$ gets one more chance of winning the game by being able to query $\mathcal{S}^{\mathsf{L}}(\cdot, \cdot, \cdot)$ on inputs of his choice, the ciphertext being the real or random key depending on b. This extra query captures the case where the key used in a block cipher was itself a ciphertext from a previous iteration. Note that it departs from the real world/ideal world paradigm, as $\mathcal{S}^{\mathsf{L}}$ is invoked for both values of b. This aspect plays a central role in our further developments. Additional types of (e.g. decryption) queries could be added to the game. However, the two proposed ones capture the usual situation where a block cipher is used to produce a key, which is then used to encrypt multiple plaintexts. It can be observed that we do not use the fact that BC is a block cipher so far. Its invertibility will however be used in the next subsection, when proposing our instance of leakage simulator.

Definition 1 (q-simulatable Leakage). *A block cipher BC with leakage function L has $(s_{\mathcal{S}}, t_{\mathcal{S}}, s_{\mathcal{A}}, t_{\mathcal{A}}, \varepsilon)$ q-simulatable leakages if there is an $(s_{\mathcal{S}}, t_{\mathcal{S}})$-bounded simulator $\mathcal{S}^{\mathsf{L}}$ such that, for every $(s_{\mathcal{A}}, t_{\mathcal{A}})$-bounded adversary $\mathcal{A}^{\mathsf{L}}$, we have:*

$$|\Pr[q\text{-}\mathsf{sim}(\mathcal{A}, \mathsf{BC}, \mathsf{L}, \mathcal{S}, 1) = 1] - \Pr[q\text{-}\mathsf{sim}(\mathcal{A}, \mathsf{BC}, \mathsf{L}, \mathcal{S}, 0) = 1]| \leq \epsilon.$$

Note that $\mathcal{A}^{\mathsf{L}(\cdot,\cdot)}$ can query the leakage function $s_{\mathcal{A}}$ times, independently of the q queries to the target implementation in the q-sim game. In practice, these $s_{\mathcal{A}}$ queries could correspond to profiling efforts to build a leakage model (e.g. as in step 2 of the attack in Section 1.3). They will also be useful to generate simulated leakages in our security proofs. As previously mentioned, we will keep small constant values for q in any practical instantiation of the q-simulatability game. This choice connects with the observation that 1-simulatability does not imply q-simulatability without severe security degradation. For example, it is easy to see that there might be block cipher implementations that offer perfect $q - 1$ simulatability but not q-simulatability. Consider a block cipher BC' built from a block cipher BC as follows: $\mathsf{BC}'_{k_1 \cdots k_q}(x) := \mathsf{BC}_{k_1 \oplus \cdots \oplus k_q}(x)$ (for a constant q), and a leakage function that leaks one of the k_i's every time the device computes. Clearly, $q - 1$ leakages will not provide any information about the cipher key, while the q-th leakage will fully disclose this key, making it trivial to detect the simulation in our game. In fact, this example also matches the usual intuition in side-channel attacks that security degrades almost exponentially with the number of queries, as will be illustrated experimentally in the next subsection.

2.2 Empirical Verifiability

To show that the previous simulatability assumption is realistic, we will first instantiate an efficient simulator $\mathcal{S}^{\mathsf{L}}_{\mathrm{s\&c}}(\cdot,\cdot,\cdot)$ to be used in the Enc and Gen queries of the q-sim game, based on a block cipher implementation. We will then discuss the interpretation of this assumption with respect to actual side-channel attacks.

As suggested by the acronym $\mathcal{S}^{\mathsf{L}}_{\mathrm{s\&c}}(\cdot,\cdot,\cdot)$, our proposal of simulator is based on the splitting and concatenation of leakage traces. For this purpose, and as we now consider concrete instantiation issues, we again need the specific notations of Section 1.1, and take the case of the AES for illustration. Namely, we will use $y = \mathsf{AES}_k(x) \rightsquigarrow \mathbf{l} = \mathbf{l}^{\mathsf{p}} \| \mathbf{l}^{\mathsf{c}}$, with $\mathbf{l}^{\mathsf{p}} = [l_0, l_1, \ldots, l_5]$ (resp. $\mathbf{l}^{\mathsf{c}} = [l_6, l_7, \ldots, l_{10}]$) denoting the first (resp. second) half of the traces, and $||$ the concatenation operator. Next, we want to build a simulator for such traces using only the knowledge of the public values x and y. In this context, a central observation already made in Section 1.2 is that any known intermediate value during a cryptographic computation can be exploited to check its consistency with the leakage. That is, taking the example of (noiseless) Hamming weight samples for illustration, it is quite easy to check whether the triple $(\mathbf{l}, x, y)$ is consistent by checking whether $l_0 = \mathsf{HW}(x)$ and $l_{10} = \mathsf{HW}(y)$. Yet, we still have that the "middle samples" $l_1, l_2, \ldots, l_8, l_9$ may not be as easy to exploit since the intermediate values $x_1, x_2, \ldots, x_8, x_9$ are not given to the adversary. As a result, the goal of the simulator will be to build traces that are at least consistent with the input/output pair (x, y). This is where the specialization to (invertible) block ciphers turns out to be useful, leading to the following proposal:

Leakage simulator instantiation $\mathcal{S}^{\mathsf{L}}_{\mathrm{s\&c}}(k^*, x, y)$.
1. Run $y' = \mathsf{AES}_{k^*}(x) \rightsquigarrow \mathbf{l}^{\mathsf{p}} \| \alpha$;
2. Compute $x' = \mathsf{AES}^{-1}_{k^*}(y)$;
3. Run $y = \mathsf{AES}_{k^*}(x') \rightsquigarrow \beta \| \mathbf{l}^{\mathsf{c}}$;
4. Output $\mathbf{l}^{\mathsf{p}} \| \mathbf{l}^{\mathsf{c}}$;

It is easy to verify that this simulator instance generates leakages that are consistent with the public values x and y, since in practice it does nothing else than generating traces from these values with a randomly generated key and concatenating them. Hence, it can be implemented using the same hardware as the target device containing the correct key. Note also that the same instance can directly be used in the Gen queries by adapting its inputs.

Interpretation. The assumption in this section suggests that there exists situations in which the leakage of a cryptographic implementation can be simulated without knowing all its secrets. For this purpose, our instance of simulator essentially relies on the possibility to use the same hardware as the one manipulating the actual cipher key. We believe this fact nicely captures the idea that the only secret in a cryptographic implementation should indeed be this key (not the device manipulating it). The assumption is also expressed as a game that can be tested by evaluation laboratories, since they could control both keys k and k^*. In practice, the main question naturally is whether the probability to win the q-sim

game can remain sufficiently low in front of actual side-channel distinguishers. There are two natural strategies that could be considered to answer it:

1. Performing standard DPA attacks exploiting the first and last encryption round leakages, e.g. trying to find an inconsistency between the x's and y's.
2. Targeting the middle rounds where concatenation occurs to find a direct inconsistency in the trace, possibly based on the key information gathered.

Starting with the first type of distinguishers, an important observation is that resisting them is at least easier than guaranteeing high HILL pseudoentropy for a block cipher key. This relates to the previously observed fact that attacks checking the consistency between the traces and the device state are not possible in the q-sim game, since the key is not given to the adversary. In other words, the device state is not known and can only be guessed, just as usually considered in side-channel analysis. Of course, being more realistic than the HILL pseudoentropy assumption does not imply empirical verifiability yet. Typically, there is little hope to ensure any security for small and unprotected devices (e.g. 8-bit microcontrollers), as key recovery is usually possible with very limited data complexities in these cases [34]. Under certain hypotheses, it is even possible to exploit the middle round leakages against such devices [31]. By contrast, a reasoning in the lines of Section 1.2 suggests that the simulatable leakage assumption could be realistic for (unprotected but parallel) hardware implementations.

For example, Figure 3 depicts the security evaluation of the best attack performed against such an implementation during the DPA Contest V2 (after two years of public investigations) [29,37]. It indicates that as long as the number of queries q remains limited (e.g. below 10), the success probability in recovering the key (hence, in finding inconsistencies between x's and y's) remains close to 2^{-128} in this case. Say that an adversary would try to exploit the q first- and last-round leakages corresponding to his Enc queries, together with the last-round leakage of his additional Gen query, and would be able to combine this information efficiently (which is unlikely in view of the large number of key candidates that remain possible after attacks with low data complexity). Then the amount of information leakage would at most be multiplied by three, still leaving comfortable security margins. Therefore, as long as our leakage-resilient PRG iterates qPRG's with small enough q's, we can conclude that this first strategy will not succeed against this hardware implementation[1]. Note that the linearity of the min/max bounds on Figure 3 typically illustrates the exponential security degradation (in time) that was mentioned in the previous subsection.

Although the second strategy is admittedly less investigated, we argue that it can also be verified for a wide variety of implementations based on the following

[1] Leakage-resilient constructions as proposed in this (and previous) works are naturally interesting in the context of small embedded devices as well, in combination with other hardware level countermeasures. In particular, they simplify the goal of protecting an implementation against arbitrary number of queries into the easier goal of protecting it against a bounded number of queries. We gave the example of the DPA Contest V2 for illustration, and because it is publicly available.

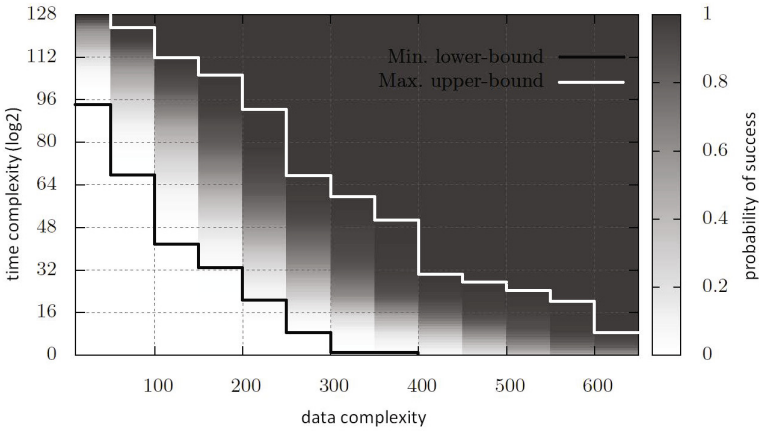


Fig. 3. Security evaluation for best attack of the DPA contest v2

reasoning. What is needed for this strategy to fail is an efficient method for concatenating side-channel traces in an indistinguishable manner. For this purpose, the key observation is that most current microelectronic devices are based on sequential logic circuits. As illustrated in Figure 4 for a couple of rounds of an AES implementation, such circuits essentially update some memory elements (i.e. the registers in dark gray on the figure) every clock cycle. And the length of these clock cycles is selected according to the longest delay needed to perform a round (aka the critical path), with some security margin. One consequence of this setup is that the circuit activity (hence, its leakage) is maximum at the beginning of each cycle (when the round computation actually takes place), and rapidly decreases afterwards. As indicated in the figure, the fact that each clock cycle should anyway be longer than the critical path guarantees that there exist samples with little or no activity. Interestingly, these points where no activity occurs usually contain no exploitable information. This observation actually

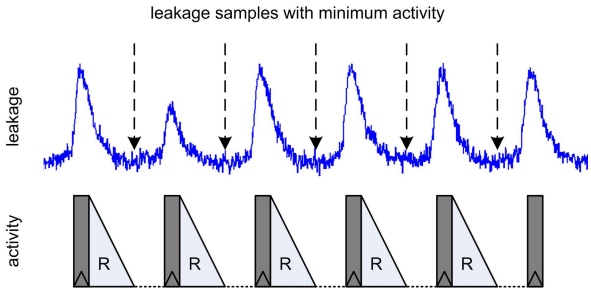


Fig. 4. Selection of samples for the concatenation of leakage traces

connects with the intuition from side-channel attacks that only a few samples in the measurements contain useful signal, i.e. the so-called "points of interest".

For example, the Signal-to-Noise Ratio (SNR) curves in [26] (Section 4.3.1) illustrate this fact. In general, concatenating traces exactly at their non-informative points can be done without risk of being distinguishable, since both the actual traces and the simulated ones would exhibit a noise following the same distribution at these points. Hence, our assumption for this second strategy to fail boils down to the existence of a couple of points *without* interest in the traces (which we believe is generally verified) and the ability to detect them. The latter task is relatively easy since (*i*) any side-channel distinguisher (e.g. the ones in Section 1.3) can be used for this purpose and (*ii*) the simulator can predict the full state corresponding to his fake inputs, hence allowing it to easily plot SNR curves. For illustration we performed such concatenations in the context of actual power traces and compared their spectrum with the one of original traces, without being able to detect any significant bias. As a result, we conclude that security against this second type of distinguishers can sometimes be ensured too.

Challenges. As for any cryptographic assumption, the claim that the simulatable leakage requirement is empirically verifiable will take strength with further investigations by physical cryptanalysts. In this respect, we believe that a central benefit of our security game is that it can be challenged using the techniques developed by the cryptographic hardware community. In order to stimulate research in this direction, we conclude this section by stating three challenges:

C1 (constructive). Design alternative instances of simulators. For example, the proposal in this section relies on the splitting and concatenation of leakage traces, based on the ability to detect "points without interest". But more sophisticated techniques for mixing the traces could be investigated.

C2 (constructive). Given any instance of simulator, design efficient block cipher implementations with q-simulatable leakages, for the largest possible q's. This challenge concurs with the one of securing these implementations against side-channel key recoveries with data complexity bounded to q.

C3 (destructive). Given a block cipher implementation and an instance of simulator, break the q-sim game with non-negligible advantage.

Regarding this last challenge, we finally note that falsifying the simulatable leakage assumption for one given instance of block cipher implementation and simulator does not imply that it cannot be verified at all. Our hope and belief is that it will be verified for a sufficiently wide range of realistic implementations.

3 Security Analysis and Proofs under Simulatable Leakages

We now want to show that the PRG of Figure 2 is secure when implemented with a secure block cipher that has 2-simulatable leakages (as previously mentioned, the proof would be similar for any constant value of q). The property we require from BC is to be a PRF. Our PRF adversary is a regular interactive Turing machine, augmented with an access to a leakage oracle $\mathsf{L}(\cdot, \cdot)$. While this oracle

is independent of the PRF challenger, nothing theoretically precludes that it might do some cryptanalytic work, and we therefore include the number of times it is queried in the adversary's total computational power.

Definition 2 (Pseudorandom Function). *A block cipher* $\mathsf{BC} : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$ *is a* (s,t,ε) pseudorandom function *(PRF) in the presence of leakage function* L *if, for every* (s,t)*-bounded adversary* $\mathcal{A}^{\mathsf{L}(\cdot,\cdot)}$*, we have that:*

$$|\Pr[\mathcal{A}^{\mathsf{L}(\cdot,\cdot),\mathsf{BC}_k(\cdot)} = 1] - \Pr[\mathcal{A}^{\mathsf{L}(\cdot,\cdot)\mathsf{F}(\cdot)} = 1]| \leq \varepsilon,$$

where k *is a random key in* $\{0,1\}^n$ *and* F *is a random function.*

Note that if the leakage function was polynomial time, this definition would be strictly equivalent to the standard definition of a PRF. However, it remains an open problem to determine the exact complexity of such physical functions (which essentially corresponds to the cost of solving Maxwell's equations for a complex circuit). Therefore, and despite it is unlikely that actual leakage functions perform any cryptanalytic work, we believe it is conceptually cleaner to keep track of their queries separately, as specified in Definition 2.

The first step towards showing the security of our stream cipher consists in proving that one call of the 2PRG construction remains secure when it leaks and when the computation of its seed also leaks, as expressed in the next lemma. All bounds include the number of calls to the leakage function and the running time.

Lemma 1 (Single Iteration). *Let* $\mathsf{BC} : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$ *with leakage function* L *be an* $(s,t,\varepsilon_{\mathsf{prf}})$ *PRF having* $(s_{\mathcal{S}}, t_{\mathcal{S}}, s, t, \varepsilon_{\mathsf{sim}})$ *2-simulatable leakages, and let* $\mathcal{S}^{\mathsf{L}}$ *be an appropriate* $(s_{\mathcal{S}}, t_{\mathcal{S}})$*-bounded leakage simulator. Then, for every* k^-, p_0, p_1 *in* $\{0,1\}^n$ *and every* $(s - 3s_{\mathcal{S}}, t - \max(t_{\mathsf{prf}}, t_{\mathsf{sim}}))$*-bounded distinguisher* $\mathcal{D}^{\mathsf{L}}$*, the following inequation holds:*

$$|\Pr[\mathcal{D}^{\mathsf{L}}(y^+, k^+, \mathsf{L}(k,p_0), \mathsf{L}(k,p_1), \mathcal{S}^{\mathsf{L}}(k^-, p_1, k)) = 1] -$$
$$\Pr[\mathcal{D}^{\mathsf{L}}(y^{+*}, k^{+*}, \mathcal{S}^{\mathsf{L}}(k,p_0,y^{+*}), \mathcal{S}^{\mathsf{L}}(k,p_1,k^{+*}), \mathcal{S}^{\mathsf{L}}(k^-,p_1,k)) = 1]| \leq \varepsilon_{\mathsf{prf}} + \varepsilon_{\mathsf{sim}},$$

with $k, y^{+*}, k^{+*} \xleftarrow{R} \{0,1\}^n$*,* $y^+ = \mathsf{BC}(k,p_0)$*,* $k^+ = \mathsf{BC}(k,p_1)$*,* t_{prf} *being equal to* $3t_{\mathcal{S}}$ *augmented with the time needed to make 2 oracle queries to the PRF challenger and select a random key uniformly in* $\{0,1\}^n$*, and* t_{sim} *being the time to relay the content of two* Enc *and one* Gen *queries from and to a* q*-sim challenger.*

Proof. The proof makes use of an intermediary distribution that provides round outputs computed with one key and leakages simulated with another key. Details appear in the long version of the paper on the IACR ePrint archive.

Based on this Lemma, we show that the last output after l iterations of 2PRG remains pseudorandom even in the presence of the public outputs and leakages for all the previous iterations. For this purpose, we first specify our PRG instance:

Definition 3 (PRG Instance). *We denote as* $\mathsf{PRG}(k_0)$ *the pseudorandom generator in Figure 2 with n-bit initial state* k_0*. Each iteration of* PRG *expands the current state by running a length-doubling PRG (2PRG*: $\{0,1\}^n \rightarrow \{0,1\}^{2n}$*) following the recurrence* $(y_i, k_i) = \mathsf{2PRG}(k_{i-1})$*, and produces* y_1*,* y_2*, ... as output.*

Next, we define our notion of leakage-resilient stream cipher as follows:

Definition 4 (Leakage-Resilient Stream Cipher). *Let* PRG *be the stream cipher of Definition 3 and let* $\mathsf{L}(k_i) = \mathsf{L}(k_i, p_0)||\mathsf{L}(k_i, p_1)$ *be the leakages from its* i^{th} *iteration. The implementation of this* PRG *is* (l, s, t, ε)*-LR-pseudorandom if, for every* (s, t) *bounded distinguisher* $\mathcal{D}^{\mathsf{L}}$*, the following inequation holds:*

$$|\Pr[\mathcal{D}^{\mathsf{L}}(y_1, \cdots, y_l, \mathsf{L}(k_0), \cdots, \mathsf{L}(k_{l-1})) = 1] \; - \\ \Pr[\mathcal{D}^{\mathsf{L}}(y_1, \cdots, y_{l-1}, U_n, \mathsf{L}(k_0), \cdots, \mathsf{L}(k_{l-1})) = 1])| \leq \epsilon,$$

with k_0 *and* U_n *uniformly random values chosen in* $\{0,1\}^n$*.*

We can now state our main theorem, which shows the leakage-resilience of the stream cipher above and offers tight bounds: we only require 2-simulatable leakages, and the security degrades linearly with the number of rounds.

Theorem 1. *Let* $\mathsf{BC} : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$ *be a block cipher that is an* $(s, t, \varepsilon_{\mathsf{prf}})$ *PRF with a leakage function* L *and* $(s_{\mathcal{S}}, t_{\mathcal{S}}, s, t, \varepsilon_{\mathsf{sim}})$ *2-simulatable leakages. Then, the implementation of* PRG *instantiated with* BC *is* $(s', t', \varepsilon', l)$*-LR-pseudo-random, where* $s' = s - (2l-1)(s_{\mathcal{S}} + 1)$*,* $\varepsilon' = 2l(\varepsilon_{\mathsf{prf}} + \varepsilon_{\mathsf{sim}})$*, and* $t' = t - t_{12}$ *where* t_{12} *is* $2lt_{\mathcal{S}}$ *augmented with the time needed to sample* $2l$ *random n-bit strings and evaluate* BC $2l$ *times, plus the time needed to relay these block cipher inputs, outputs and leakages from and to oracles*[2]*.*

Proof. We rely on Lemma 1 and on a hybrid argument. The full proof appears in the long version of the paper on the IACR ePrint archive.

We may observe that this proof, like the one of Lemma 1, does not make full use of the power of the adversary in the q-sim game. They could indeed accommodate a non-interactive variant of game in which the plaintexts of the Enc and Gen queries are fixed, and the key of the Gen query is chosen randomly.

Conclusion

This paper suggests that the specification of realistic leakage assumptions may allow simplifying the proofs of natural constructions (such as the stream cipher in Figure 2), for which one intuitively expects an improved resistance against

[2] We do not include these relay times in the operation counts, because we assume them to be small compared to the time needed for the block cipher evaluations.

practical side-channel attacks. While the simulatable leakage requirement introduced in this work naturally raises open questions regarding the implementation scenarios in which it can be fulfilled (e.g. the challenges in Section 2.2), we can at least claim that it is more realistic than requirements such as the bounded range or high HILL pseudoentropy used in previous proofs for similar (symmetric cryptographic) constructions. Interesting scopes for further investigations include the application of simulatability to other primitives, and the quest for more generic yet empirically verifiable assumptions that could be exploited to analyze the leakage of cryptographic implementations.

Acknowledgements. We thank Ran Canetti and Martijn Stam for interesting discussions and useful suggestions. This work has been funded in parts by the European Commission through the ERC project 280141 (acronym CRASH) and the European ISEC action grant HOME/2010/ISEC/AG/INT-011 B-CCENTRE project. François-Xavier Standaert is an associate researcher of the Belgian Fund for Scientific Research (FNRS-F.R.S.). Yu Yu was supported by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61172085, 61061130540, 61073174, 61103221, 11061130539, 61021004 and 61133014.

References

1. Akavia, A., Goldwasser, S., Vaikuntanathan, V.: Simultaneous hardcore bits and cryptography against memory attacks. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 474–495. Springer, Heidelberg (2009)
2. Alwen, J., Dodis, Y., Naor, M., Segev, G., Walfish, S., Wichs, D.: Public-key encryption in the bounded-retrieval model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 113–134. Springer, Heidelberg (2010)
3. Alwen, J., Dodis, Y., Wichs, D.: Leakage-resilient public-key cryptography in the bounded-retrieval model. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 36–54. Springer, Heidelberg (2009)
4. Barak, B., Shaltiel, R., Wigderson, A.: Computational analogues of entropy. In: Arora, S., Jansen, K., Rolim, J.D.P., Sahai, A. (eds.) RANDOM 2003 and APPROX 2003. LNCS, vol. 2764, pp. 200–215. Springer, Heidelberg (2003)
5. Bellare, M., Yee, B.S.: Forward-security in private-key cryptography. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 1–18. Springer, Heidelberg (2003)
6. Boneh, D., DeMillo, R.A., Lipton, R.J.: On the importance of checking cryptographic protocols for faults (extended abstract). In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 37–51. Springer, Heidelberg (1997)
7. Boyle, E., Segev, G., Wichs, D.: Fully leakage-resilient signatures. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 89–108. Springer, Heidelberg (2011)
8. Brier, E., Clavier, C., Olivier, F.: Correlation power analysis with a leakage model. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 16–29. Springer, Heidelberg (2004)
9. Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 13–28. Springer, Heidelberg (2003)

10. Dodis, Y., Kalai, Y.T., Lovett, S.: On cryptography with auxiliary input. In: Mitzenmacher, M. (ed.) STOC, pp. 621–630. ACM (2009)
11. Dodis, Y., Pietrzak, K.: Leakage-resilient pseudorandom functions and side-channel attacks on feistel networks. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 21–40. Springer, Heidelberg (2010)
12. Dziembowski, S., Pietrzak, K.: Leakage-resilient cryptography. In: FOCS, pp. 293–302. IEEE Computer Society (2008)
13. Faust, S., Pietrzak, K., Schipper, J.: Practical leakage-resilient symmetric cryptography. In: Prouff, E., Schaumont, P. (eds.) CHES 2012. LNCS, vol. 7428, pp. 213–232. Springer, Heidelberg (2012)
14. Gandolfi, K., Mourtel, C., Olivier, F.: Electromagnetic analysis: Concrete results. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 251–261. Springer, Heidelberg (2001)
15. Gierlichs, B., Batina, L., Tuyls, P., Preneel, B.: Mutual information analysis. In: Oswald, E., Rohatgi, P. (eds.) CHES 2008. LNCS, vol. 5154, pp. 426–442. Springer, Heidelberg (2008)
16. Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W.: Lest we remember: Cold boot attacks on encryption keys. In: van Oorschot, P.C. (ed.) USENIX Security Symposium, pp. 45–60. USENIX Association (2008)
17. Hsiao, C.-Y., Lu, C.-J., Reyzin, L.: Conditional computational entropy, or toward separating pseudoentropy from compressibility. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 169–186. Springer, Heidelberg (2007)
18. Ishai, Y., Prabhakaran, M., Sahai, A., Wagner, D.: Private circuits II: Keeping secrets in tamperable circuits. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 308–327. Springer, Heidelberg (2006)
19. Ishai, Y., Sahai, A., Wagner, D.: Private circuits: Securing hardware against probing attacks. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 463–481. Springer, Heidelberg (2003)
20. Joye, M., Tunstall, M.: Fault Analysis in Cryptography. Springer (2012)
21. Katz, J., Vaikuntanathan, V.: Signature schemes with bounded leakage resilience. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 703–720. Springer, Heidelberg (2009)
22. Kiltz, E., Pietrzak, K.: Leakage resilient elGamal encryption. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 595–612. Springer, Heidelberg (2010)
23. Knudsen, L.R., Robshaw, M.: The Block Cipher Companion. In: Information Security and Cryptography. Springer (2011)
24. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
25. Mangard, S., Oswald, E., François-Xavier: One for all – all for one: unifying standard differential power analysis attacks. IET Information Security 5(2), 100–110 (2011)
26. Mangard, S., Oswald, E., Popp, T.: Power analysis attacks - revealing the secrets of smart cards. Springer (2007)
27. Micali, S., Reyzin, L.: Physically observable cryptography (extended abstract). In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 278–296. Springer, Heidelberg (2004)
28. Moradi, A.: Statistical tools flavor side-channel collision attacks. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 428–445. Springer, Heidelberg (2012)

29. Telecom ParisTech, http://www.dpacontest.org/ (retrieved on August 1, 2012)
30. Pietrzak, K.: A leakage-resilient mode of operation. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 462–482. Springer, Heidelberg (2009)
31. Renauld, M., Standaert, F.-X., Veyrat-Charvillon, N.: Algebraic side-channel attacks on the AES: Why time also matters in DPA. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 97–111. Springer, Heidelberg (2009)
32. Schindler, W., Lemke, K., Paar, C.: A stochastic model for differential side channel cryptanalysis. In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 30–46. Springer, Heidelberg (2005)
33. Standaert, F.-X.: How leaky is an extractor? In: Abdalla, M., Barreto, P.S.L.M. (eds.) LATINCRYPT 2010. LNCS, vol. 6212, pp. 294–304. Springer, Heidelberg (2010)
34. Standaert, F.-X., Gierlichs, B., Verbauwhede, I.: Partition vs. comparison side-channel distinguishers: An empirical evaluation of statistical tests for univariate side-channel attacks against two unprotected cmos devices. In: Lee, P.J., Cheon, J.H. (eds.) ICISC 2008. LNCS, vol. 5461, pp. 253–267. Springer, Heidelberg (2009)
35. Standaert, F.-X., Pereira, O., Yu, Y., Quisquater, J.-J., Yung, M., Oswald, E.: Leakage resilient cryptography in practice. In: Sadeghi, A.-R., Naccache, D. (eds.) Towards Hardware-Intrinsic Security, Information Security and Cryptography, pp. 99–134. Springer, Heidelberg (2010)
36. Veyrat-Charvillon, N., Gérard, B., Renauld, M., Standaert, F.-X.: An optimal key enumeration algorithm and its application to side-channel attacks. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 390–406. Springer, Heidelberg (2013)
37. Veyrat-Charvillon, N., Gérard, B., Standaert, F.-X.: Security evaluations beyond computing power. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 126–141. Springer, Heidelberg (2013)
38. Yu, Y., Standaert, F.-X.: Practical leakage-resilient pseudorandom objects with minimum public randomness. In: Dawson, E. (ed.) CT-RSA 2013. LNCS, vol. 7779, pp. 223–238. Springer, Heidelberg (2013)
39. Yu, Y., Standaert, F.-X., Pereira, O., Yung, M.: Practical leakage-resilient pseudorandom generators. In: ACM Conference on Computer and Communications Security, pp. 141–151. ACM (2010)

Highly-Scalable Searchable Symmetric Encryption with Support for Boolean Queries

David Cash[1], Stanislaw Jarecki[2], Charanjit Jutla[3], Hugo Krawczyk[3], Marcel-Cătălin Roşu[3], and Michael Steiner[3]

[1] Rutgers University
david.cash@cs.rutgers.edu
[2] University of California Irvine
stasio@ics.uci.edu
[3] IBM Research
{csjutla,hugokraw,rosu,msteiner}@us.ibm.com

Abstract. This work presents the design and analysis of the first searchable symmetric encryption (SSE) protocol that supports conjunctive search and general Boolean queries on outsourced symmetrically-encrypted data and that scales to very large databases and arbitrarily-structured data including free text search. To date, work in this area has focused mainly on single-keyword search. For the case of conjunctive search, prior SSE constructions required work linear in the total number of documents in the database and provided good privacy only for structured attribute-value data, rendering these solutions too slow and inflexible for large practical databases.

In contrast, our solution provides a realistic and practical trade-off between performance and privacy by efficiently supporting very large databases at the cost of moderate and well-defined leakage to the outsourced server (leakage is in the form of data access patterns, never as direct exposure of plaintext data or searched values). We present a detailed formal cryptographic analysis of the privacy and security of our protocols and establish precise upper bounds on the allowed leakage. To demonstrate the real-world practicality of our approach, we provide performance results of a prototype applied to several large representative data sets, including encrypted search over the whole English Wikipedia (and beyond).

1 Introduction

Outsourcing data storage to external servers ("the cloud") is a major industry trend that offers great benefits to database owners. At the same time, data outsourcing raises confidentiality and privacy concerns. Simple encryption of outsourced data is a hindrance to search capabilities such as the data owner wanting to search a backup or email archive, or query a database via attribute-value pairs. This problem has motivated much research on advanced searchable encryption schemes that enable searching on the encrypted data while protecting the confidentiality of data and queries.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 353–373, 2013.

Searchable Symmetric Encryption (SSE) is a cryptographic primitive addressing encrypted search. To securely store and search a database with an SSE scheme, a client first uses a special encryption algorithm which produces an encrypted version of the database, including encrypted metadata, that is then stored on an external server. Later, the client can interact with the server to carry out a search on the database and obtain the results (this is called the symmetric setting as there is only one writer to the database, the owner, who uses symmetric encryption – the public key variant of the problem has also been extensively studied, see further related work).

An important line of research (e.g., [23,11,6,8,7,19]) gives practical constructions of SSE that support searching for documents that contain a *single* specified keyword. In these schemes, the server's work scales with the size of the result set (independently of the database size) and the leakage to the server is limited to the set of returned (encrypted) documents and a few global parameters of the system, such as total data size and number of documents. While efficient and offering good privacy, all of these SSE schemes are severely limited in their expressiveness during search: A client can *only* specify a single keyword to search on, and then it receives *all* of the documents containing that keyword. In practical settings, like remotely-stored email or large databases, a single-keyword search will often return a large number of documents that the user must then download and filter herself to find the relevant results.

Conjunctive and Boolean Search. To provide a truly practical search capability, a system needs to at least support conjunctive search, namely, given a set of keywords find all documents that contain all these keywords. Clearly, this problem can be reduced to the single-keyword case by performing a search for each individual keyword and then letting the server or client do the intersection between the resultant document sets. This often results in inefficient searches (e.g., half the database size if one of the conjunctive terms is "gender=male") and significant leakage (e.g., it reveals the set of documents matching each keyword). Yet, this naïve solution is the only known sublinear solution to SSE conjunctive search (other than those using generic techniques such as FHE or ORAM). All other dedicated solutions require server work that is *linear in the size of the database*. Of these solutions, the one that provides the best privacy guarantees is due to Golle et al. [13], with variants presented in later work, e.g., [1,3]. They show how to build for each conjunctive query a set of tokens that can be tested against *each* document in the database (more precisely, against an encoded version of the document's keywords) to identify matching documents. These solutions only leak the set of matching documents (and possibly the set of attributes being searched for) but are unsuited for large databases due to the $O(d)$ work incurred by the server, where d is the number of documents or records in the database. This cost is paid for every search regardless of the size of the result set or the number of documents matching each individual conjunctive term. Moreover, these solutions require either $O(d)$ communication and exponentiations between server and client or $O(d)$ costly pairing operations (as well as dedicated cryptographic assumptions). Another serious limitation of this

approach is that it works only for structured attribute-value type databases and does not support free text search. In addition, none of the above solutions extend to general Boolean queries.

The Challenge of Large Databases and the Challenge of Being Imperfect. In this work we investigate solutions to conjunctive queries and general Boolean queries that can be practical even for very large databases where linear search is prohibitively expensive. Our application settings include databases that require search over tens of millions documents (and billions of document-keyword pairs), with search based on attribute-value pairs (as in relational databases) and free text - see below for specific numbers used in evaluating our prototype. To support such scale in a truly practical way one needs to relax absolute privacy and allow for some leakage beyond the result set.

As an example, compare the case of a conjunction of two highly-frequent keywords whose intersection returns a small number of documents but whose individual terms are very frequent (e.g., search for "name=David AND gender=Female"), with the case of a conjunction that returns the same number of documents but all the individual terms in the conjunction are themselves infrequent. Search complexity in these two cases, *even in the case of plaintext data (hence in any encrypted solution),* is likely to be different and noticeable to the searching server, except if searches are artificially padded to a full database search hence leading to $O(d)$ complexity[1]. Note that even powerful tools, such as ORAM, that can be used to search on encrypted data in smaller-scale databases already incur non-trivial leakage if the search performance is to be sublinear. Indeed, the mere computational cost, in number of ORAM operations, of a given search is sufficient to distinguish between the two cases above (of all high-frequency conjunctive terms vs. all small-frequency terms) unless the searches are padded to the maximal search size, resulting in $O(d)$ search cost. Thus, resorting to weaker security guarantees is a necessity for achieving practical conjunctive search. Not only this presents design challenges but also raises non-trivial theoretical challenges for analyzing and characterizing in a precise way the form and amount of leakage incurred by a solution.

Ideally, we would like to run the search with complexity proportional to the number of matches of the *least frequent term in the conjunction*, which is the standard of plaintext information retrieval algorithms. In addition, the computational efficiency of database processing and of search is of paramount importance in practice. Generic tools such as FHE [10] or ORAM [12] are too costly for very large databases, although they may be used as sub-components of a solution if applied to small data subsets.

Our Contributions. We develop the first non-generic sublinear SSE schemes supporting conjunctive keyword search (and more general Boolean queries, see below) with a non-trivial combination of security and efficiency. The schemes performance scales to very large datasets and arbitrarily-structured data, including free-text search. We attain efficiency by allowing some forms of access-pattern

[1] A costly alternative is to pre-compute all n-term conjunctions in time $O(|\mathsf{W}|^n)$.

leakage, but with a much better leakage profile than the naïve solution implied by single-keyword SSE, as discussed above. Further, we establish the security of our solution via an explicit and precise leakage profile and a *proof that this is all the leakage* incurred by this solution. Our formal setting follows a simulation-based abstraction that adapts the SSE models of Curtmola et al. [8] and Chase and Kamara [7], and assumes an adaptive adversarial model. The essence of the security notion is that the view of the server (the attacker in this setting) can be efficiently simulated given a precisely-defined *leakage profile* but without access to the actual plaintext data. Such a profile may include leakage on the total size of the database, on access patterns (e.g., the intersection between two sets of results) and on queries (e.g., repetition of queries), but never the direct exposure of plaintext data or searched values. Thus, a protocol proven secure ensures that the server holding the encrypted data and serving the queries does not learn anything about the data and queries other than what can be deduced from the specified leakage[2]. The characterization of leakage and the involved proof of security that we present are central technical contributions that complement our protocol design work.

The centerpiece of the protocol design is a "virtual" secure two-party protocol in which the server holds encrypted pointers to documents, the client holds a list of keywords, and the output of the protocol is the set of encrypted pointers that point to documents containing all the client's keywords. The client is then able to decrypt these pointers and obtain the matching (encrypted) documents but the server cannot carry this decryption nor can it learn the keywords in the client's query. While this underlying protocol is interactive, the level of performance targeted by our solutions requires avoiding multiple rounds of interaction. We achieve this by a novel approach that pre-computes parts of the protocol messages and stores them in encrypted form at the server. Then, during search, the client sends information to the server that allows to unlock these pre-computed messages without further interaction. Our implementation of this protocol, which we name OXT, uses only DH-type operations over any Diffie-Hellman group which enables the use of the secure and most efficient DH elliptic curves (with additional common-base optimizations).[3] The complexity of our search protocols is *independent* of the number of documents in the database. To search for documents containing $w_1, \ldots, w_n$, the search complexity of our scheme scales with the number of documents matching the estimated *least frequent keyword in the conjunction*. We note that while building a search based on term frequency is standard in information retrieval, our solution seems to be the first to exploit this approach in the encrypted setting. This leads not only to good performance but also improves privacy substantially. All our solutions support search on structured data (e.g., attribute-value databases) as well as on free text, and combinations of both.

[2] See the discussion in Section 6 on "semantic leakage".

[3] We also present a scheme (BXT in Section 3.1) that only uses symmetric-key operations but provides less privacy, and a pairings-based scheme (PXT in the full version [5]) that optimizes communication at the expense of more computation.

Boolean Queries. Our solution to conjunctive queries extends to answer *any Boolean query.* This includes negations, disjunctions, threshold queries, and more. The subset of such queries that we can answer efficiently includes any expression of the form "$w_1 \wedge \phi(w_2, \ldots, w_m)$" (intended to return any document that matches keyword w_1 and in addition satisfies the (unconstrained) formula ϕ on the remaining keywords)[4]. The search complexity is proportional to the number of documents that contain w_1. Surprisingly, the leakage profile for such complex expressions can be reduced to the leakage incurred by a conjunction with the same terms $w_1, w_2, \ldots, w_n$, hence allowing us to re-use the analysis of the conjunctive case to the much more general boolean setting. Finally, any disjunction of the above forms can also be answered with an additive cost over the disjunction expressions.

Further Extensions. In [4] we report on further practical enhancements to our protocols, including support for dynamic databases (i.e., allowing additions, deletions and modification of documents in the database). Our protocols can also be applied to the *multi-client setting* [7,17,18] where a data owner outsources its encrypted data to an external server and enables *other parties* to perform queries on the encrypted data by providing them with search tokens for specific queries. In this case, one considers not only leakage to the server but also leakage to clients beyond the information that their tokens are authorized to disclose. In subsequent work [16], we address issues of authorization in this setting as well as the challenging problem of hiding the queries not only from the server but also from the token provider - see for example IARPA's SPAR program and its requirement for supporting private queries on very large databases [14]. See also [21] for an independent, concurrent work in the latter setting from which a solution to the SSE problem can also be extracted. Finally, in ongoing work, we are extending the set of supported queries with range queries, substring matching, and more.

Implementation. To show the practical viability of our solution we prototyped OXT and ran experiments with two data sets: the Enron email data set [9] with more than 1.5 million documents (email messages and attachments) where all words, including attachments and envelope information, have been indexed; and the ClueWeb09 [20] collection of crawled web-pages from which we extracted several databases of increasing size with the largest one consisting of 13 million documents (0.4TB of HTML files). Approximately one third of the latter database is a full snapshot of the English Wikipedia. The results of these tests show not only the suitability of our conjunction protocols for data sets of medium size (such as the Enron one) but demonstrate the scalability of these solutions to much larger databases (we target databases of one or two orders of magnitude larger). Existing solutions that are linear in the number of documents would be mostly impractical even for the Enron dataset. Refer to Section 5 for more information on implementation and performance. More advanced results are reported in [4].

[4] An example of such query on an email repository is: Search for messages with Alice as Recipient, not sent by Bob, and containing at least two of the words {searchable, symmetric, encryption}.

Other Related Work and Research Questions. See full version [5] for more discussion on related work and Section 6 for several interesting research questions arising from our work.

2 Definitions and Tools

2.1 SSE Syntax and Security Model

Searchable Symmetric Encryption. A database is composed of a collection of d documents, each comprised of a set of keywords W_i (we use "documents" generically; they can represent text documents, records in a relational database - in which case keyword are represented as attribute-value pairs, a combination of both, etc.). The output from the SSE protocol for a given search query are indices (or identifiers) ind corresponding to the documents that satisfy the query. A client program can then use these indices to retrieve the encrypted documents and decrypt them. This definition allows to decouple the storage of payloads (which can be done in a variety of ways, with varying types of leakage) from the storage of metadata that is the focus of our protocols.

SSE Scheme Syntax and Correctness. Let λ be the security parameter. We will take identifiers and keywords to be bit strings. A database $\mathsf{DB} = (\mathsf{ind}_i, \mathsf{W}_i)_{i=1}^d$ is represented as a list of identifier/keyword-set pairs, where $\mathsf{ind}_i \in \{0,1\}^\lambda$ and $\mathsf{W}_i \subseteq \{0,1\}^*$. We will always write $\mathsf{W} = \bigcup_{i=1}^d \mathsf{W}_i$ (we think of the ind values as identifiers that can be revealed to the outsourced server, e.g., a randomization of the original document identifiers; in particular these are the identifiers that will be used to retrieve query-matching documents). A *query* $\psi(\bar{w})$ is specified by a tuple of keywords $\bar{w} \in \mathsf{W}^*$ and a boolean formula ψ on $\bar{w}$. We write $\mathsf{DB}(\psi(\bar{w}))$ for the set of identifiers of documents that "satisfy" $\psi(\bar{w})$. Formally, this means that $\mathsf{ind}_i \in \mathsf{DB}(\psi(\bar{w}))$ iff the formula $\psi(\bar{w})$ evaluates to true when we replace each keyword w_i with true or false depending on if $w_i \in \mathsf{W}_i$ or not. Below we let d denote the number of documents in DB, $m = |\mathsf{W}|$ and $N = \sum_{w \in \mathsf{W}} |\mathsf{DB}(w)|$.

A *searchable symmetric encryption (SSE) scheme* Π consists of an algorithm $\mathsf{EDBSetup}$ and a protocol Search between the client and server, all fitting the following syntax. $\mathsf{EDBSetup}$ takes as input a database DB, and outputs a secret key K along with an encrypted database EDB. The search protocol is between a *client* and *server*, where the client takes as input the secret key K and a query $\psi(\bar{w})$ and the server takes as input EDB. At the end of the protocol the client outputs a set of identifiers and the server has no output. We say that an SSE scheme is *correct* if for all inputs DB and queries $\psi(\bar{w})$ for $\bar{w} \in \mathsf{W}^*$, if $(K, \mathsf{EDB}) \xleftarrow{\$} \mathsf{EDBSetup}(\mathsf{DB})$, after running Search with client input $(K, \psi(\bar{w}))$ and server input EDB, the client outputs the set of indices $\mathsf{DB}(\psi(\bar{w}))$.

Security of SSE. We recall the semantic security definitions from [8,7]. The definition is parametrized by a *leakage function* $\mathcal{L}$, which describes what an adversary (the server) is allowed to learn about the database and queries when

interacting with a secure scheme. Formally, security says that the server's view during an adaptive attack (where the server selects the database and queries) can be simulated given only the output of $\mathcal{L}$.

Definition 1. *Let* $\Pi = (\mathsf{EDBSetup}, \mathsf{Search})$ *be an SSE scheme and let* $\mathcal{L}$ *be a stateful algorithm. For algorithms* A *and* S*, we define experiments (algorithms)* $\mathbf{Real}_A^{\Pi}(\lambda)$ *and* $\mathbf{Ideal}_{A,S}^{\Pi}(\lambda)$ *as follows:*

$\mathbf{Real}_A^{\Pi}(\lambda)$ *:* $A(1^\lambda)$ *chooses* DB*. The experiment then runs* $(K, \mathsf{EDB}) \leftarrow \mathsf{EDBSetup}(\mathsf{DB})$ *and gives* EDB *to* A*. Then* A *repeatedly chooses a query* q*. To respond, the game runs the* Search *protocol with client input* (K, q) *and server input* EDB *and gives the transcript and client output to* A*. Eventually* A *returns a bit that the game uses as its own output.*

$\mathbf{Ideal}_{A,S}^{\Pi}(\lambda)$ *: The game initializes a counter* $i = 0$ *and an empty list* $\mathbf{q}$*.* $A(1^\lambda)$ *chooses* DB*. The experiment runs* $\mathsf{EDB} \leftarrow S(\mathcal{L}(\mathsf{DB}))$ *and gives* EDB *to* A*. Then* A *repeatedly chooses a query* q*. To respond, the game records this as* $\mathbf{q}[i]$*, increments* i*, and gives to* A *the output of* $S(\mathcal{L}(\mathsf{DB}, \mathbf{q}))$*. (Note that here,* $\mathbf{q}$ *consists of all previous queries in addition to the latest query issued by* A*.) Eventually* A *returns a bit that the game uses as its own output.*

We say that Π *is* $\mathcal{L}$*-semantically-secure against adaptive attacks if for all adversaries* A *there exists an algorithm* S *such that* $\Pr[\mathbf{Real}_A^{\Pi}(\lambda) = 1] - \Pr[\mathbf{Ideal}_{A,S}^{\Pi}(\lambda) = 1] \leq \mathrm{neg}(\lambda)$.

We note that in the security analysis of our SSE schemes we include the client's output, the set of indices $\mathsf{DB}(\psi(\bar{w}))$, in the adversary's view in the real game, to model the fact that these ind's will be used for retrieval of encrypted document payloads.

2.2 T-Sets

We present a definition of syntax and security for a new primitive that we call a *tuple set*, or *T-set*. Intuitively, a T-set allows one to associate a list of fixed-sized data tuples with each keyword in the database, and later issue keyword-related tokens to retrieve these lists. We will use it in our protocols as an "expanded inverted index". Indeed, prior single-keyword SSE schemes, e.g. [8,7], can be seen as giving a specific T-set instantiation and using it as an inverted index to enable search – see Section 2.3. In our SSE schemes for conjunctive keyword search, we will use a T-set to store more data than a simple inverted index, and we will also compose it with other data structures. The abstract definition of a T-set will allow us to select an implementation that provides the best performance for the size of the data being stored.

T-Set Syntax and Correctness. Formally, a T-set implementation $\Sigma = (\mathsf{TSetSetup}, \mathsf{TSetGetTag}, \mathsf{TSetRetrieve})$ will consist of three algorithms with the following syntax: $\mathsf{TSetSetup}$ will take as input an array $\mathbf{T}$ of lists of equal-length bit strings indexed by the elements of W. The $\mathsf{TSetSetup}$ procedure outputs

a pair (TSet, K_T). $\mathsf{TSetGetTag}$ takes as input the key K_T and a keyword w and outputs stag. $\mathsf{TSetRetrieve}$ takes the TSet and an stag as input, and returns a list of strings. We say that Σ is *correct* if for all W, $\mathbf{T}$, and any $w \in \mathsf{W}$, $\mathsf{TSetRetrieve}(\mathsf{TSet}, \mathsf{stag}) = \mathbf{T}[w]$ when $(\mathsf{TSet}, K_T) \leftarrow \mathsf{TSetSetup}(\mathbf{T})$ and $\mathsf{stag} \leftarrow \mathsf{TSetGetTag}(K_T, w)$. Intuitively, $\mathbf{T}$ holds lists of tuples associated with keywords and correctness guarantees that the $\mathsf{TSetRetrieve}$ algorithm returns the data associated with the given keyword.

T-Set Security and Implementation. The security goal of a T-set implementation is to hide as much as possible about the tuples in $\mathbf{T}$ and the keywords these tuples are associated to, except for vectors $\mathbf{T}[w_1], \mathbf{T}[w_2], \ldots$ of tuples revealed by the client's queried keywords $w_1, w_2, \ldots$. (For the purpose of T-set implementation we equate client's query with a single keyword.) The formal definition is similar to that of SSE (think of the SSE setting for single-keyword queries) and we provide it in [5] where we also show a specific T-set implementation that achieves optimal security, namely, it only reveals (an upper bound on) the aggregated database size $N = \sum_{w \in \mathsf{W}} |\mathsf{DB}(w)|$ *We refer to such a T-set implementation as* $\mathsf{optimal}$.

2.3 T-Sets and Single Keyword Search

Here we show how a T-set can be used as an "secure inverted index" to build an SSE scheme for single-keyword search. The ideas in this construction will be the basis for our conjunctive search SSE schemes later, and it essentially abstracts prior constructions [8,7]. The details of the scheme, called SKS, are given in

$\underline{\mathsf{EDBSetup}(\mathsf{DB})}$

- Select key K_S for PRF F, and parse DB as $(\mathsf{ind}_i, \mathsf{W}_i)_{i=1}^d$.
- Initialize $\mathbf{T}$ to an empty array indexed by keywords from $\mathsf{W} = \cup_{i=1}^d \mathsf{W}_i$.
- For each $w \in \mathsf{W}$, build the tuple list $\mathbf{T}[w]$ as follows:
  - Initialize $\mathbf{t}$ to be an empty list, and set $K_e \leftarrow F(K_S, w)$.
  - For all $\mathsf{ind} \in \mathsf{DB}(w)$ in random order: $\mathsf{e} \xleftarrow{\$} \mathsf{Enc}(K_e, \mathsf{ind})$; append e to $\mathbf{t}$.
  - Set $\mathbf{T}[w] \leftarrow \mathbf{t}$.
- $(\mathsf{TSet}, K_T) \leftarrow \mathsf{TSetSetup}(\mathbf{T})$.
- Output the key (K_S, K_T) and $\mathsf{EDB} = \mathsf{TSet}$.

$\underline{\mathsf{Search\ protocol}}$

- The client takes as input the key (K_S, K_T) and a keyword w to query. It computes $\mathsf{stag} \leftarrow \mathsf{TSetGetTag}(K_T, w)$ and sends stag to the server.
- The server computes $\mathbf{t} \leftarrow \mathsf{TSetRetrieve}(\mathsf{TSet}, \mathsf{stag})$, and sends $\mathbf{t}$ to the client.
- Client sets $K_e \leftarrow F(K_S, w)$; for each e in $\mathbf{t}$, it computes $\mathsf{ind} \leftarrow \mathsf{Dec}(K_e, \mathsf{e})$ and outputs ind.

Fig. 1. SKS: Single-Keyword SSE Scheme

Figure 1. It uses as subroutines a PRF $F : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^\lambda$, and a CPA secure symmetric encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$ that has λ-bit keys.

3 SSE Schemes for Conjunctive Keyword Search

Existing SSE schemes for conjunctive queries ([13] and subsequent work) work by encoding each document individually and then processing a search by testing *each* encoded document against a set of tokens. Thus the server's work grows linearly with the number of documents, which is infeasible for large databases. In addition, these schemes only work for attribute-value type databases (where documents contain a single value per attribute) but not for unstructured data, e.g., they cannot search text documents.

Here *we develop the first sub-linear conjunctive-search solutions for arbitrarily-structured data, including free text.* In particular, when querying for the documents that match all keywords $w_1, \ldots, w_n$, our search protocol scales with the size of the (estimated) *smallest* $\mathsf{DB}(w_i)$ set among all the conjunctive terms w_i.

The naïve solution. To motivate our solutions we start by describing a straightforward extension of the single-keyword case (protocol SKS from Figure 1) to support conjunctive keyword searching. On input a conjunctive query $\bar{w} = (w_1, \ldots, w_n)$, the client and server run the search protocol from SKS independently for each term w_i in $\bar{w}$ with the following modifications. Instead of returning the lists t to the client, the server receives K_{e_i}, $i = 1, ..., n$, from the client and decrypts the e values to obtain a set of ind's for each w_i. Then, the server returns to client the ind values in the intersection of all these sets. The search complexity of this solution is proportional to $\sum_{i=1}^{n} |\mathsf{DB}(w_i)|$ which improves, in general, on solutions whose complexity is linear in the number of documents in the whole database. However, this advantage is reduced for queries where one of the terms is a very high-frequency word (e.g., in a relational database of personal records, one may have a keyword $w = (gender, male)$ as a conjunctive term, thus resulting in a search of, say, half the documents in the database). In addition, this solution incurs excessive leakage to the server who learns the complete sets of indices ind for each term in a conjunction.

Our goal is to reduce both computation and leakage in the protocol by tying those to the less frequent terms in the conjunctions (i.e., terms w with small sets $\mathsf{DB}(w)$).

3.1 Basic Cross-Tags (BXT) Protocol

To achieve the above goal we take the following approach that serves as the basis for our main SSE-conjunctions scheme OXT presented in the next subsection. Here we exemplify the approach via a simplified protocol, BXT. Assume (see below) that the client, given $\bar{w} = (w_1, \ldots, w_n)$, can choose a term w_i with a relatively small $\mathsf{DB}(w_i)$ set among $w_1, \ldots, w_n$; for simplicity assume this is w_1. The parties could run an instance of the SKS search protocol for the keyword w_1 after which the client gets all documents matching w_1 and locally searches for the remaining conjunctive terms. This is obviously inefficient as it may require

retrieving many more documents than actually needed. The idea of BXT is indeed to use SKS for the server to retrieve $\mathsf{TSet}(w_1)$ but then perform the intersection with the terms $w_2, \ldots, w_n$ at the server who will only return the documents matching the full conjunction. We achieve this by augmenting SKS as follows.

During $\mathsf{EDBSetup}(\mathsf{DB})$, in addition to TSet, a set data structure XSet is built by adding to it elements xtag computed as follows. For each $w \in \mathsf{W}$, a value $\mathsf{xtrap} = F(K_X, w)$ is computed where K_X is a PRF key chosen for this purpose; then for each $\mathsf{ind} \in \mathsf{DB}(w)$ a value $\mathsf{xtag} = f(\mathsf{xtrap}, \mathsf{ind})$ is computed and added to XSet where f is an unpredictable function of its inputs (e.g., f can be a PRF used with xtrap as the key and ind as input). The Search protocol for a conjunction $(w_1, \ldots, w_n)$, chooses the estimated least frequent keyword, say w_1, and sets, as in SKS, $K_e \leftarrow F(K_S, w_1)$, $\mathsf{stag} \leftarrow \mathsf{TSetGetTag}(K_T, w_1)$. Then, for each $i = 2, \ldots, n$, it sets $\mathsf{xtrap}_i \leftarrow F(K_X, w_i)$ and sends $(K_e, \mathsf{stag}, \mathsf{xtrap}_2, \ldots, \mathsf{xtrap}_n)$ to the server. The server uses stag to retrieve $\mathbf{t} = \mathsf{TSetRetrieve}(\mathsf{TSet}, \mathsf{stag})$. Then, for each ciphertext e in $\mathbf{t}$, it decrypts $\mathsf{ind} = \mathsf{Dec}(K_e, \mathsf{e})$ and if $f(\mathsf{xtrap}_i, \mathsf{ind}) \in \mathsf{XSet}$ for all $i = 2, \ldots, n$, it sends ind to the client.[5]

Correctness of the BXT protocol is easy to verify. Just note that a document indexed by ind includes a word w represented by stag if and only if $\mathsf{xtag} = f(\mathsf{xtrap}, \mathsf{ind}) \in \mathsf{XSet}$. Regarding implementation of XSet, it can use any set representation that is history-independent, namely, it is independent of the order in which the elements of the set were inserted. For TSet security and implementation see Section 2.

Terminology (s-terms and x-terms): We will refer to the conjunctive term chosen as the estimated least frequent term among the query terms as the *s-term* ('s' for SKS or "small") and refer to other terms in the conjunction as *x-terms* ('x' for "cross"); this is the reason for the 's' and 'x' in names such as $\mathsf{stag}, \mathsf{xtag}, \mathsf{stag}, \mathsf{xtrap}$, etc.

The server's work in BXT scales with $n \cdot |\mathsf{DB}(w_1)|$, where w_1 is the conjunction's s-term. This represents a major improvement over existing solutions which are linear in $|\mathsf{DB}|$ and also a significant improvement over the naïve solution whenever there is a term with relatively small set $\mathsf{DB}(w_1)$ that can be identified by the client, which is usually the case as discussed in Section 3.1. Communication is optimal ($O(n)$-size token plus the final results set) and computation involves only PRF operations.

Security-wise this protocol *improves substantially* on the above-described naïve solution by leaking only the (small) set of ind's for the s-term and not for x-terms. Yet, this solution lets the server learn statistics about x-terms by correlating information from different queries. Specifically, the server can use the value xtrap_i received in one query and check it against any ind found through an s-term of another query. But note that direct intersections between x-terms of different queries are not possible other than via the s-terms (e.g., if two queries (w_1, w_2) and (w_1', w_2') are issued, the server can learn the (randomly permuted) results of (w_1, w_2') and (w_1', w_2) but not (w_2, w_2').

[5] While in SKS one can choose to let the server decrypt the ind's directly instead of the client, in BXT this is necessary for computing the xtag's.

In settings where computation and communications are very constrained BXT may provide for an acceptable privacy-performance balance. In general, however, we would like to improve on the privacy of this solution even if at some performance cost. We do so in the next section with the OXT protocol, so we omit a formal analysis BXT – we note that the security of BXT needs the set of ind's to be unpredictable, a condition not needed in the other protocols.

Choosing the S-Term. The performance and privacy of our conjunction protocols improves with "lighter" s-terms, namely, keywords w whose $\mathsf{DB}(w)$ is of small or moderate size. While it is common to have such terms in typical conjunctive queries, our setting raises the question of how can the client, who has limited storage, choose adequate s-terms. In the case of relational databases one can use general statistics about attributes to guide the choice of the s-term (e.g., prefer a last-name term to a first-name term, always avoid gender as the s-term, etc.). In the case of free text the choice of s-term can be guided by term frequency which can be made available, requiring a small state stored at the client or retrieved from the server. We extend on this topic in the full version [5].

3.2 Oblivious Cross-Tags (OXT) Protocol

The BXT scheme is vulnerable to the following simple attack: When the server gets xtrap_i for a query $(w_1, \ldots, w_n)$, it can save it and later use it to learn if any revealed ind value is a document with keyword w_i by testing if $f(\mathsf{xtrap}_i, \mathsf{ind}) \in \mathsf{XSet}$. This allows an honest-but-curious server to learn, for example, the number of documents matching each queried s-term with each queried x-term (even for terms in different queries). This attack is possible because BXT reveals the keys that enable the server to compute $f(\mathsf{xtrap}_i, \cdot)$ itself.

One way to mitigate the attack is to have the client evaluate the function for the server instead of sending the key. Namely, the server would send all the encrypted ind values that it gets in $\mathbf{t}$ (from the TSet) to the client who will compute the function $f(\mathsf{xtrap}_i, \mathsf{ind})$ and send back the results. However, this fix adds a round of communication with consequent latency, it allows the server to cheat by sending ind values from another query's s-term (from which the server can compute intersections not requested by the client), and is unsuited to the multi-client SSE setting [7] discussed in the introduction (since the client would learn from the inds it receives the results of conjunctions it was not authorized for). Note that while the latter two issues are not reflected in our current formal model, avoiding them expands significantly the applicability of OXT.

These issues suggest a solution where we replace the function $f(\mathsf{xtrap}, \cdot)$ (where $\mathsf{xtrap} = F(K_X, w)$) with a form of *oblivious shared computation between client and server.* A first idea would be to use *blinded exponentiation* (as in Diffie-Hellman based oblivious PRF) in a group G of prime order p: Using a PRF F_p with range Z_p^* (and keys K_I, K_X), we derive a value $\mathsf{xind} = F_p(K_I, \mathsf{ind}) \in Z_p^*$ and define each xtag to be $g^{F_p(K_X,w)\cdot \mathsf{xind}}$. The shared computation would proceed by the client first sending the value $g^{F_p(K_X,w_i)\cdot z}$ where $z \in Z_p^*$ is a blinding factor;

EDBSetup(DB)

- Select key K_S for PRF F, keys K_X, K_I, K_Z for PRF F_p (with range in Z_p^*), and parse DB as $(\mathsf{ind}_i, \mathsf{W}_i)_{i=1}^{d}$.
- Initialize $\mathbf{T}$ to an empty array indexed by keywords from W.
- Initialize XSet to an empty set.
- For each $w \in \mathsf{W}$, build the tuple list $\mathbf{T}[w]$ and XSet elements as follows:
  - Initialize $\mathbf{t}$ to be an empty list, and set $K_e \leftarrow F(K_S, w)$.
  - For all ind in $\mathsf{DB}(w)$ in random order, initialize a counter $c \leftarrow 0$, then:
    - Set $\mathsf{xind} \leftarrow F_p(K_I, \mathsf{ind})$, $z \leftarrow F_p(K_Z, w \,\|\, c)$ and $y \leftarrow \mathsf{xind} \cdot z^{-1}$.
    - Compute $\mathsf{e} \leftarrow \mathsf{Enc}(K_e, \mathsf{ind})$, and append (e, y) to $\mathbf{t}$.
    - Set $\mathsf{xtag} \leftarrow g^{F_p(K_X, w) \cdot \mathsf{xind}}$ and add xtag to XSet.
  - $\mathbf{T}[w] \leftarrow \mathbf{t}$.
- $(\mathsf{TSet}, K_T) \leftarrow \mathsf{TSetSetup}(\mathbf{T})$.
- Output the key $(K_S, K_X, K_I, K_Z, K_T)$ and $\mathsf{EDB} = (\mathsf{TSet}, \mathsf{XSet})$.

Search protocol

- The client's input is the key $(K_S, K_X, K_I, K_Z, K_T)$ and query $\bar{w} = (w_1, \ldots, w_n)$.

  It sends to the server the message $(\mathsf{stag}, \mathsf{xtoken}[1], \mathsf{xtoken}[2], \ldots)$ defined as:
  - $\mathsf{stag} \leftarrow \mathsf{TSetGetTag}(K_T, w_1)$.
  - For $c = 1, 2 \ldots$ and until server sends stop
    - For $i = 2, \ldots, n$, set $\mathsf{xtoken}[c, i] \leftarrow g^{F_p(K_Z, w_1 \,\|\, c) \cdot F_p(K_X, w_i)}$
    - Set $\mathsf{xtoken}[c] = \mathsf{xtoken}[c, 2], \ldots, \mathsf{xtoken}[c, n]$.
- The server has input $(\mathsf{TSet}, \mathsf{XSet})$. It responds as follows.
  - It sets $\mathbf{t} \leftarrow \mathsf{TSetRetrieve}(\mathsf{TSet}, \mathsf{stag})$.
  - For $c = 1, \ldots, |\mathbf{t}|$
    - retrieve (e, y) from the c-th tuple in $\mathbf{t}$
    - if $\forall i = 2, \ldots, n : \mathsf{xtoken}[c, i]^y \in \mathsf{XSet}$ then send e to the client.
  - When last tuple in $\mathbf{t}$ is reached, send stop to $\mathcal{C}$ and halt.
- Client sets $K_e \leftarrow F(K_S, w_1)$; for each e received, computes $\mathsf{ind} \leftarrow \mathsf{Dec}(K_e, \mathsf{e})$ and outputs ind.

Fig. 2. OXT: Oblivious Cross-Tags Protocol

the server would raise this to the power xind and finally the client would de-blind it by raising to the power z^{-1} mod p to obtain $g^{F_p(K_X, w_i) \cdot \mathsf{xind}}$. Unfortunately, this idea does not quite work as the server would learn $\mathsf{xtag} = g^{F_p(K_X, w_i) \cdot \mathsf{xind}}$ and from this, and its knowledge of xind, it would learn $g^{F_p(K_X, w_i)}$, which allows it to carry out an attack similar to the one against BXT. This also requires client-server interaction on a per-xind basis, a prohibitive cost.

In the design of OXT we address these two problems. The idea is to *precompute* (in EDBSetup) the blinding part of the oblivious computation and store it in the EDB. I.e., in each tuple corresponding to a keyword w and document xind, we store a blinded value $y_c = \mathsf{xind} \cdot z_c^{-1}$, where z_c is an element in Z_p^* derived (via a PRF) from w and a tuple counter c (this counter, incremented for each tuple in $\mathbf{t}$ associated with w, serves to ensure independent blinding values z).

During search, the server needs to compute the value $g^{F_p(K_X,w_i)\cdot \mathsf{xind}}$ for each xind corresponding to a match for w_1 and then test these for membership in XSet. To enable this, the client sends, for the c-th tuple in $\mathbf{t}$, a n-long array $\mathsf{xtoken}[c]$ defined by $\mathsf{xtoken}[c,i] := g^{F_p(K_X,w_i)\cdot z_c}$, $i = 1,\ldots,n$, where z_c is the precomputed blinding derived by from w (via a PRF) and the tuple counter c. The server then performs the T-set search to get the results for w_1, and filters the c-th result by testing if $\mathsf{xtoken}[c,i]^{y_c} \in \mathsf{XSet}$ for all $i = 2,\ldots,n$. This protocol is correct because $\mathsf{xtoken}[c,i]^{y_c} = g^{F_p(K_X,w_i)\cdot z_c \cdot \mathsf{xind} \cdot z_c^{-1}} = g^{F_p(K_X,w_i)\cdot \mathsf{xind}}$, meaning that the server correctly recomputes the pseudorandom values in the XSet.

Putting these ideas together results in the OXT protocol of Figure 2. Note that the client sends the xtoken arrays (each holding several values of the form $g^{F_p(K_X,w_i)\cdot z}$) until instructed to stop by the server. There is no other communication from server to client (alternatively, server can send the number of elements in $\mathsf{TSet}(w)$ to the client who will respond with such number of xtoken arrays).[6]

Note that while the description above is intended to provide intuition for the protocol's design, assessing the security (leakage) of OXT is non-trivial, requiring an intricate security analysis that we provide in Section 4.

OXT consists of a single round of interaction, where the message sent by the client is of size proportional to $|\mathsf{DB}(w_1)|$, [7] and the response to the client is minimal, consisting only of the result set (i.e., the set of encrypted ind's matching the query). The computational cost of OXT lies in the use of exponentiations, however, thanks to the use of very efficient elliptic curves (we only require the group to be DDH) and *fixed-base exponentiations*, this cost is practical even for very large databases as demonstrated by the performance numbers in Section 5.

OXT leaks much less information to the server than BXT. Indeed, since the server, call it $\mathcal{S}$, learns neither the ind values nor $\mathsf{xtrap}_j, j = 2,\ldots,n$, its ability to combine conjunctive terms from one query with terms from another query is significantly reduced. In particular, while in BXT $\mathcal{S}$ learns the intersection between s-terms of any two queries, in OXT this is possible only in the following case: the two queries can have different s-terms, but same x-term and there is a document containing both s-terms (the latter is possible since if the s-terms of two queries share a document ind and an x-term xtrap then the xtag value $f(\mathsf{xtrap},\mathsf{ind})$ will be the same in both queries indicating that ind and xtrap are the same). The only other leakage via s-terms is that $\mathcal{S}$ learns when two queries have the same s-term w_1 and the size of the set $\mathsf{DB}(w_1)$. Finally, regrading intra-query leakage if $\mathcal{C}$ responds with the values $\mathsf{xtag}_j, j = 2,\ldots,n$, in the same order for all ind's, then in case $n > 2$, $\mathcal{S}$ learns the number of documents matching any sub-conjunction that includes w_1 and any subset of $w_2,\ldots,w_n$. If, instead, $\mathcal{C}$

[6] The same protocol supports single-keyword search (or 1-term conjunctions) by skipping the $c = 1, 2, \ldots$ at both client and server, hence falling back to the SKS protocol of Figure 1.

[7] For typical choices of w_1, such message will be of small or moderate size. For large values of $|\mathsf{DB}(w_1)|$ one can cap the search to the first k tuples for a threshold k, say 1000. For example, in the case of a 3-term conjunction and xtag values of size 16 bytes, this will result in just 32 Kbyte message.

randomly permutes the values $\mathsf{xtag}_j, j = 2, \ldots, n$ before sending these values to $\mathcal{S}$, then $\mathcal{S}$ learns the maximal number of satisfied terms per tuple in $\mathsf{TSet}(w_1)$, but not the size of sets matching $w_1 \wedge w_i, i = 1, \ldots, n$, or any other proper sub-conjunctions (except for what can be learned in conjunction with other leakage information). *In Section 4 we formally analyze the security of* OXT *making the above description of leakage precise.*

As noted before, even a leakage profile as the above that only reveals access patterns can still provide valuable information to an attacker that possesses prior information on the database and queries. We don't discuss here specific countermeasures for limiting the ability of an attacker to perform such statistical inference – see [15] for an example of potential masking techniques.

3.3 Processing Boolean Queries with OXT

We describe an extension to OXT that can handle *arbitrary* Boolean query expressions. We say that a Boolean expression in n terms is in *Searchable Normal Form* (SNF) if it is of the form $w_1 \wedge \phi(w_2, \ldots, w_n)$ where ϕ is an *arbitrary* Boolean formula (e.g., "$w_1 \wedge (w_2 \vee w_3 \vee \neg w_4)$"). OXT can be extended to answer such queries: On input a query of the form $w_1 \wedge \phi(w_2, \ldots, w_n)$, the client creates a modified boolean expression $\hat{\phi}$ in new boolean variables v_i $(i = 2, \ldots, n)$, which is just ϕ but with each w_i replaced by v_i. Thus, the client uses w_1 as the s-term and computes its stag as in OXT, and computes the xtrap (i.e. the xtoken array) for all the other terms w_i $(i > 1)$. It then sends the stag and the xtraps in the order of their index. It also sends the server the above modified boolean expression $\hat{\phi}$. The server fetches the TSet corresponding to the stag as in OXT. It also computes the xtag corresponding to each x-term, also as in OXT. But, it decides on sending (to the Client) the encrypted ind corresponding to each tuple in the TSet based on the following computation (which is the only different part from OXT): for each $i = 2, \ldots, n$, the server treats the variable v_i as a boolean variable and sets it to the truth value of the expression $(\mathsf{xtoken}[c, i])^y \in \mathsf{XSet}$. Then it evaluates the expression $\phi(v_2, \ldots, v_n)$. If the result is true, it returns the e value in that tuple to the Client.

OXT can also be used to answer any disjunction of SNF expressions. Actually, note that OXT can answer *any* Boolean query by adding to the database a field TRUE which all documents satisfy. Then a search for any expression $\phi(w_1, \ldots, w_n)$ can be implemented as "TRUE $\wedge \phi(w_1, \ldots, w_n)$", which is in SNF and can be searched as in the SNF case above. Clearly, this will take time linear in the number of documents but it can be implemented if such functionality is considered worth the search complexity.

4 Security of OXT

In the full version [5] we provide a complete detailed analysis of OXT and its extension to Boolean queries. Due to space constraints we illustrate the security claim for the particular case of two-term conjunctions, but this restricted case is

representative of the general case. We start by describing the protocol's leakage profile, $\mathcal{L}_{\text{oxt}}(\mathsf{DB}, \mathbf{q})$, followed by a security theorem showing that this is *all* of the information leaked by the protocol.

In describing the leakage profile of the OXT protocol we will assume an *optimal* T-set implementation (see Section 2) as the one presented in [5], namely, with optimal leakage $N = \sum_{i=1}^{d} |\mathsf{W}_i|$. We represent a sequence of Q two-term conjunction queries by $\mathbf{q} = (\mathbf{s}, \mathbf{x})$ where an individual query is a two-term conjunction $\mathbf{s}[i] \wedge \mathbf{x}[i]$ which we write as $\mathbf{q}[i] = (\mathbf{s}[i], \mathbf{x}[i])$.

We define $\mathcal{L}_{\text{oxt}}(\mathsf{DB}, \mathbf{q})$, for $\mathsf{DB} = (\mathsf{ind}_i, \mathsf{W}_i)_{i=1}^{d}$ and $\mathbf{q} = (\mathbf{s}, \mathbf{x})$, as a tuple $(N, \mathsf{EP}, \mathsf{SP}, \mathsf{RP}, \mathsf{IP})$ formed as follows:

- $N = \sum_{i=1}^{d} |\mathsf{W}_i|$ is the total number of appearances of keywords in documents.
- EP is the *equality pattern* of $\mathbf{s} \in \mathsf{W}^Q$ indicating which queries have the equal s-terms. Formally, $\mathsf{EP} \in [m]^Q$ is formed by assigning each keyword an integer in $[m]$ determined by the order of appearance in $\mathbf{s}$. For example, if $\mathbf{s} = (a, a, b, c, a, c)$ then $\mathsf{EP} = (1, 1, 2, 3, 1, 3)$. To compute $\mathsf{EP}[i]$ one finds the least j such that $\mathbf{s}[j] = \mathbf{s}[i]$ and then lets $\mathsf{EP}[i] = |\{\mathbf{s}[1], \ldots, \mathbf{s}[j]\}|$ be the number of unique keywords appearing at indices less than or equal to j.
- SP is the *size pattern* of the queries, i.e. the number of documents matching the first keyword in each query. Formally, $\mathsf{SP} \in [d]^Q$ and $\mathsf{SP}[i] = |\mathsf{DB}(\mathbf{s}[i])|$.
- RP is the *results pattern*, which consists of the results sets $(R_1, \ldots, R_Q)$, each defined by $R_i = I_\pi(\mathbf{s}[i]) \cap I_\pi(\mathbf{x}[i])$.
- IP is the *conditional intersection pattern*, which is a Q by Q table with entries defined as follows: $\mathsf{IP}[i, j]$ is an empty set if either $\mathbf{s}[i] = \mathbf{s}[j]$ or $\mathbf{x}[i] \neq \mathbf{x}[j]$. However, if $\mathbf{s}[i] \neq \mathbf{s}[j]$ and $\mathbf{x}[i] = \mathbf{x}[j]$ then $\mathsf{IP}[i, j] = \mathsf{DB}(\mathbf{s}[i]) \cap \mathsf{DB}(\mathbf{s}[j])$.

Understanding the Leakage Components. The parameter N can be replaced with an upper bound given by the total size of EDB but leaking such a bound is unavoidable. The equality pattern EP leaks repetitions in the s-term of different queries; this is a consequence of our optimized search that singles out the s-term in the query. This leakage can be mitigated by having more than one TSet per keyword and the client choosing different incarnations of the Tset for queries with repeated s-terms. SP leaks the number of documents satisfying the s-term in a query and is also a direct consequence of our approach of optimizing search time via s-terms; it can be mitigated by providing an upper bound on the number of documents rather than an exact count by artificially increasing Tset sizes. RP is a the result of the query and therefore no real leakage. Finally, the IP component is the most subtle and it means that if two queries have different s-terms but same x-term, then the indexes which match both the s-terms are leaked (if there are no documents which match both s-terms then nothing is leaked). This "conditional intersection pattern" can be seen as the price for the rich functionality enabled by our x-terms and XSet approach that allows for the computation of arbitrary boolean queries. Note that on query $\mathbf{q}[i] = (\mathbf{s}[i] \wedge \mathbf{x}[i])$ the OXT protocol lets the server compute a deterministic function $f(\mathsf{xtrap}(\mathbf{x}[i]), \mathsf{ind})$ of the x-term $\mathbf{x}[i]$ for all ind's that match the s-term $\mathbf{s}[i]$. Therefore a repeated xtag value in two queries $\mathbf{q}[i]$ and $\mathbf{q}[j]$ implies

that $\mathbf{x}[i] = \mathbf{x}[j]$ and that $\mathsf{DB}(\mathbf{s}[i])$ and $\mathsf{DB}(\mathbf{s}[j])$ contain a common index. Even though this index is not revealed to the server, we still model this information by simply revealing $\mathsf{DB}(s[i]) \cap \mathsf{DB}(s[j])$ if $\mathbf{x}[i] = \mathbf{x}[j]$. This "pessimistic" upper bound on the leakage simplifies the leakage representation. As said, if the above intersection is empty then no information about the equality of the x-terms $\mathbf{x}[i], \mathbf{x}[j]$ is revealed. The probability of non-empty intersections is minimized by consistently choosing low-frequent s-terms. Note also that for queries with s-terms belonging to the same field with unique per-document value (e.g., both s-terms containing different last names in a database with a last-name field), the IP leakage is empty.

Theorem 1. *The SSE scheme* OXT *implemented with an optimal T-set is* $\mathcal{L}_{\mathrm{oxt}}$*-semantically-secure if all queries are 2-conjunctions, assuming that the DDH assumption holds in* G*, that* F *and* F_p *are secure PRFs, and* $(\mathsf{Enc}, \mathsf{Dec})$ *is an IND-CPA secure symmetric encryption scheme.*

Proof Sketch. The proof of the theorem is delicate and lengthy, and is presented in [5] for the general case of multi-term conjunctions (with extensions to the case of Boolean queries). To get some intuition for why the scheme is secure, we start by examining why each of the outputs of $\mathcal{L}$ is necessary for a correct simulation. Of course, this does nothing to show that they are *sufficient* for simulation, but it will be easier to see why this is all of the leakage once their purpose is motivated. For the sake of this sketch we assume a non-adaptive adversary.

The size of the XSet is equal to the value N leaked. The equality pattern EP (or something computationally equivalent to it) is necessary due to the fact that the stag values are deterministic, so a server can observe repetitions of stag values to determine if $\mathbf{s}[i] = \mathbf{s}[j]$ for all i, j. The size pattern is also necessary as the server will always learn the number of matches for the first keyword in the conjunction by observing the number of tuples returned by the T-set. We include the results pattern to enable the simulator to produce the client results for queries in way consistent the conditional intersection pattern.

The final and most subtle part of the leakage is the conditional intersection pattern IP. The IP is present in the leakage because of the following passive attack. During the computation of the search protocol, the values tested for membership in the XSet by the server have the form $f(\mathsf{xtrap}(w_i), \mathsf{ind}) = g^{F_p(K_X, w_i) \cdot F_p(K_I, \mathsf{ind})}$, where w_i is the i-th x-term in a search query and ind is an identifier for a document that matched the s-term in that query. As we explain above, the leakage comes from the fact that f is a deterministic function, and so these values will *repeat* if and only if (except for a negligible probability of a collision) (1) the two queries involve the same x-term and (2) the sets of indexes which match the s-terms involved in these queries have some indexes in common.

Our proof makes formal the claim that the output of $\mathcal{L}$ is sufficient for a simulation. We outline a few of the technical hurdles in the proof without dealing with the details here. For this discussion, we assume that reductions to PRF security and encryption security go through easily, allowing us to treat PRF outputs as random and un-opened ciphertexts as encryptions of zeros.

We first handle the information leaked by the XSet. An unbounded adversary could compute the discrete logarithms of the XSet elements and derive information about which documents match which keywords. We want to show however that a poly-time adversary learns nothing from the XSet due to the assumed hardness of the DDH problem. Formally, we need to show that we can replace the elements of XSet with random elements that carry no information about the database, but there is a technical difficulty: some of the exponents (specifically, the xind values) that will play the roll of hidden exponents in the DDH reduction are used in the computation of the xtrap values, and these are revealed in the transcripts. A careful rearrangement of the game computation will show that this is not as bad as it seems, because the xind values are "blinded out" by the z values. We stress that this requires some care, because the z values are also used twice, and we circumvent this circularity by computing the XSet first and then computing the transcripts "backwards" in way that is consistent with the XSet. Now a reduction to DDH becomes clear, as the XSet values can be dropped in obliviously as real-or-random group elements.

With the XSet leakage eliminated, the rest of the work is in showing that the simulator can arrange for a correct-looking pattern of "repeats" in the documents matched and in the values tested against the XSet. While riddled with details, this is intuitively a rather straightforward task that is carried out in the latter games of the proof.

5 OXT Implementation and Experimental Results

This section reports the status of the OXT implementation and several latency and scalability measurements, which should be viewed as providing empirical evidence to the performance and scalability claims made earlier in the paper. Additional details on the implementation can be found in the full version [5].

Prototype. The realization of OXT consists of EDBSetup, which generates the EDB, Client, which generates the stag and the xtoken stream, and EDBSearch, which uses the EDB to process the Client's request. All three use the same cryptographic primitives, which leverage the OpenSSL 1.0.1 library. As the DH groups we use NIST 224p elliptic curve. The overall C code measures roughly 16k lines.

To scale beyond the server's RAM, the TSet is realized as a disk-resident paged hash table. Each tuple list $\mathbf{T}[w]$ in the TSet is segmented into fixed-size blocks of tuples keyed by a tag stag_c. This tag is derived by a PRF from the list's stag and a segment counter. EDBSearch uses page-level direct I/O to prevent buffer cache pollution in the OS, as the hash table pages are inherently uncachable, and parallelizes disk accesses using asynchronous I/O (`aio_*` system calls). The XSet is realized as a RAM-resident Bloom filter [2], which enables the sizing the false positive rate to the lowest value that the server's RAM allows. For the experiments presented next, the false positive rate is 2^{-20} and the Bloom filter XSet still occupies only a small fraction of our server's RAM.

Data Sets. To show the practical viability of our solution we run tests on two data sets: the Enron email data set [9] with more than 1.5 million documents

(email messages and attachments) which generate 145 million distinct (keyword, docId) pairs, and the ClueWeb09 [20] collection of crawled web-pages from which we extracted several databases of increasing size, where the largest one is based on 0.4TB of HTML files which generate almost 3 billion distinct (keyword, docId) pairs. For the Enron email data, the TSet hash table and the XSet Bloom filter are 12.4 GB and 252 MB, respectively. The corresponding sizes for the largest ClueWeb09 data set are 144.4 GB and 9.7 GB, respectively.

Experimental Results. All experiments were run on IBM Blades HS22 attached to a commodity SAN system. Figure 3 shows the latency of queries on one-term, called v, and three variants of two-term conjunctive queries on the Enron data set. In one-term queries, the selectivity of v (the number of documents matching v) varies from 3 to 690492. As this query consists only of an s-term, the figure illustrates that its execution time is linear in the cardinality of $\mathsf{TSet}(v)$. The two-term conjunctive queries combine the previous queries with a fixed reference term. In the first of these queries, the fixed term acts as an x-term: each tuple retrieved from the TSet is checked against the XSet at the cost of an exponentiation. However, as we perform these operation in parallel to retrieving the TSet buckets from the disk, their cost is completely hidden by the disk I/O latency. Micro-benchmarks show that the average cost of retrieving a bucket which has a capacity of 10 tuples is comparable to $\sim 1,000$ single-threaded exponentiations. Similarly, the client-side exponentiation in the OXT protocol can be overlapped with disk and network I/O. It illustrates the fact that exponentiations, over fast elliptic curves, are relatively cheap when compared to the cost of accessing storage systems. The last two conjunctive queries use two fixed terms with different selectivity, α and β, as s-terms. Their invariable execution time is dominated by the cost of retrieving the TSet tuples corresponding to their s-terms, irrespective the variable selectivity of the xterm v: the two horizontal lines intersect with the single-term curve exactly where v corresponds to α and β, respectively. This illustrates the importance of s-term selection, as discussed in Section 3.1.

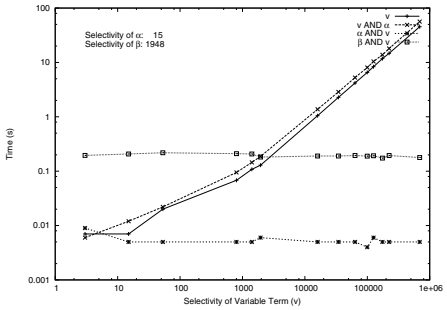


Fig. 3. Enron Performance Measurement: Single Term & Conjunction

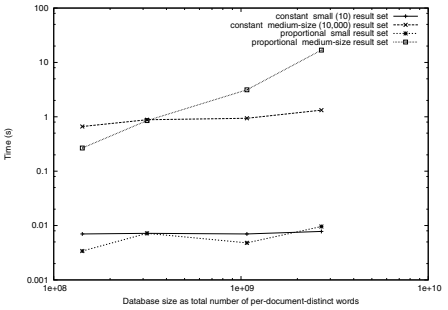


Fig. 4. Clueweb09 Performance Measurement: Scaling Database Size

To further assess the scalability of EDBSearch, we generated several EDBs from increasingly larger subsets of the ClueWeb09 data set ranging from 408,450 to 13,284,801 HTML files having a size from 20 to 410 GBs and from 142,112,027 to 2,689,262,336 distinct (keyword, docId) pairs. To make these databases comparable, we injected several artificial and non-conflicting keywords to randomly selected documents simulating words of various selectivity. Figure 4 confirms that our implementation matches the scalability of our (theoretical) algorithms even when our databases exceed the size of available RAM: If the size of the result set is constant, then query time is largely independent of the size of the database and for result sets where the size is proportional to the database size, the cost is linear in the database size.

6 Conclusions and Research Questions

The premise of this work is that in order to provide truly practical SSE solutions one needs to accept a certain level of leakage; therefore, the aim is to achieve an acceptable balance between leakage and performance, with formal analysis ensuring upper bounds on such leakage. Our solutions strike such a practical balance by offering performance that scales to very large data bases; supporting search in both structured and textual data with general Boolean queries; and confining leakage to access (to encrypted data) patterns and some query-term repetition only, with formal analysis defining and proving the exact boundaries of leakage. We stress that while in our solutions leakage never occurs in the form of direct exposure of plain data or searched values, when combined with side-information that the server may have (e.g., what are the most common searched words), such leakage can allow for statistical inference on plaintext data. Nonetheless, it appears that in many practical settings the benefits of search would outweigh moderate leakage (especially given the alternatives of outsourcing the plaintext data or keeping it encrypted but without the ability to run useful searches).

Our report on the characteristics and performance of our prototype points to the fact that scalability can only be achieved by low-complexity protocols which admit highly parallelizable implementations of their computational and I/O paths. Our protocols are designed to fulfill these crucial performance requirements.

There are interesting design and research challenges arising from this work. What we call "the challenge of being imperfect" calls for trade-offs between privacy and performance that can only be evaluated on the basis of a formal treatment of leakage. Understanding the limits of what is possible in this domain and providing formal lower bounds on such trade-offs appears as a non-trivial problem that deserves more attention. Some of these problems may still be unresolved even for plaintext data. The seemingly inherent difference pointed out in the introduction between the complexity of resolving conjunctions with high-frequency terms versus conjunctions with low-frequency terms, but with a similar-size result set, may be such a case. We do not know of a proven lower

bound in this case although the work of Patrascu [22], for example, may point to some relevant conjectured bounds.

Another important evaluation of leakage is what we refer to as "semantic leakage." How much can an attacker learn from the data given the formal leakage profile and side knowledge on the plaintext data? Clearly, the answer to this question is application-dependent but one may hope for some general theory in which these questions can be studied. The success of differential privacy in related domains opens some room for optimism in this direction. Demonstrating specific attacks in real-world settings is also an important direction to pursue. We note that in some settings just revealing the size of the number of documents matching a query may leak important information on the query contents (e.g., [15]). Therefore, developing masking techniques that include dummy or controlled data to obscure statistical information available to the attacker seems as an important research direction to strengthen the privacy of solutions as those developed here. ORAM-related techniques can be certainly help in this setting, especially given the progress on the practicality of these techniques in last years.

Yet another research direction is to characterize plaintext-search algorithms that lend themselves for adaptation to the encrypted setting. The s-term and x-term based search that we use is such an example: It treats data in "black-box" form that translates well to the encrypted setting. In contrast, search that looks at the data itself (e.g., sorting it) may not work in this setting or incur in significantly increased leakage (e.g., requiring order-preserving or deterministic encryption). Finally, it would be interesting to see more examples (in other two-party, or multi-party, protocols) of our approach, which is central to the design of OXT, of removing interaction from protocols by pre-computing and storing some of the protocol messages during a pre-computation phase.

Acknowledgment. Supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center (DoI / NBC) contract number D11PC20201. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

References

1. Ballard, L., Kamara, S., Monrose, F.: Achieving efficient conjunctive keyword searches over encrypted data. In: Qing, S., Mao, W., López, J., Wang, G. (eds.) ICICS 2005. LNCS, vol. 3783, pp. 414–426. Springer, Heidelberg (2005)
2. Bloom, B.H.: Space/time trade-offs in hash coding with allowable errors. Communications of the Association for Computing Machinery 13(7), 422–426 (1970)
3. Byun, J.W., Lee, D.-H., Lim, J.-I.: Efficient conjunctive keyword search on encrypted data storage system. In: Atzeni, A.S., Lioy, A. (eds.) EuroPKI 2006. LNCS, vol. 4043, pp. 184–196. Springer, Heidelberg (2006)

4. Cash, D., Jagger, J., Jarecki, S., Jutla, C., Krawczyk, H., Rosu, M.C., Steiner, M.: Dynamic searchable encryption in very-large databases: Data structures and implementation (manuscript, 2013)
5. Cash, D., Jarecki, S., Jutla, C., Krawczyk, H., Rosu, M., Steiner, M.: Highlyscalable searchable symmetric encryption with support for boolean queries. Report 2013/169, Cryptology ePrint Archive (2013), http://eprint.iacr.org/2013/169
6. Chang, Y.-C., Mitzenmacher, M.: Privacy preserving keyword searches on remote encrypted data. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 442–455. Springer, Heidelberg (2005)
7. Chase, M., Kamara, S.: Structured encryption and controlled disclosure. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 577–594. Springer, Heidelberg (2010)
8. Curtmola, R., Garay, J.A., Kamara, S., Ostrovsky, R.: Searchable symmetric encryption: improved definitions and efficient constructions. In: Juels, A., Wright, R.N., Vimercati, S. (eds.) ACM CCS 2006, pp. 79–88. ACM Press (October 2006)
9. EDRM (edrm.net). Enron data set, http://www.edrm.net/resources/data-sets/edrm-enron-email-data-set-v2
10. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Mitzenmacher, M. (ed.) 41st ACM STOC, pp. 169–178. ACM Press (May/June 2009)
11. Goh, E.-J.: Secure indexes. Cryptology ePrint Archive, Report 2003/216 (2003), http://eprint.iacr.org/
12. Goldwasser, S., Ostrovsky, R.: Invariant signatures and non-interactive zeroknowledge proofs are equivalent (extended abstract). In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 228–245. Springer, Heidelberg (1993)
13. Golle, P., Staddon, J., Waters, B.: Secure conjunctive keyword search over encrypted data. In: Jakobsson, M., Yung, M., Zhou, J. (eds.) ACNS 2004. LNCS, vol. 3089, pp. 31–45. Springer, Heidelberg (2004)
14. IARPA. Security and Privacy Assurance Research (SPAR) Program - BAA (2011), http://www.iarpa.gov/solicitations_spar.html/
15. Islam, M., Kuzu, M., Kantarcioglu, M.: Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. In: Proceedings of the Symposium on Network and Distributed Systems Security (NDSS 2012), San Diego, CA. Internet Society (February 2012)
16. Jarecki, S., Jutla, C., Krawczyk, H., Rosu, M.C., Steiner, M.: Outsourced symmetric private information retrieval (manuscript 2013)
17. Kamara, S., Lauter, K.: Cryptographic cloud storage. In: Sion, R., Curtmola, R., Dietrich, S., Kiayias, A., Miret, J.M., Sako, K., Sebé, F. (eds.) RLCPS, WECSR, and WLC 2010. LNCS, vol. 6054, pp. 136–149. Springer, Heidelberg (2010)
18. Kamara, S., Papamanthou, C., Röder, T.: CS2: A searchable cryptographic cloud storage system (2011), http://research.microsoft.com/pubs/148632/CS2.pdf
19. Kamara, S., Papamanthou, C., Roeder, T.: Dynamic searchable symmetric encryption. In: Proc. of CCS 2012 (2012)
20. Lemur Project. ClueWeb09 dataset, http://lemurproject.org/clueweb09.php/
21. Pappas, V., Vo, B., Krell, F., Choi, S.G., Kolesnikov, V., Keromytis, A., Malkin, T.: Blind Seer: A Scalable Private DBMS (manuscript, 2013)
22. Patrascu, M.: Towards polynomial lower bounds for dynamic problems. In: 42nd ACM STOC, pp. 603–610. ACM Press (2010)
23. Song, D.X., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: 2000 IEEE Symposium on Security and Privacy, pp. 44–55. IEEE Computer Society Press (May 2000)

Message-Locked Encryption for Lock-Dependent Messages

Martín Abadi[1,3], Dan Boneh[2,⋆], Ilya Mironov[1], Ananth Raghunathan[2,⋆,⋆⋆], and Gil Segev[2,⋆,⋆⋆]

[1] Microsoft Research Silicon Valley
[2] Stanford University
[3] University of California, Santa Cruz

Abstract. Motivated by the problem of avoiding duplication in storage systems, Bellare, Keelveedhi, and Ristenpart have recently put forward the notion of Message-Locked Encryption (MLE) schemes which subsumes *convergent encryption* and its variants. Such schemes do not rely on permanent secret keys, but rather encrypt messages using keys derived from the messages themselves.

We strengthen the notions of security proposed by Bellare et al. by considering plaintext distributions that may depend on the public parameters of the schemes. We refer to such inputs as *lock-dependent* messages. We construct two schemes that satisfy our new notions of security for message-locked encryption with lock-dependent messages.

Our main construction deviates from the approach of Bellare et al. by avoiding the use of ciphertext components derived deterministically from the messages. We design a fully randomized scheme that supports an equality-testing algorithm defined on the ciphertexts.

Our second construction has a deterministic ciphertext component that enables more efficient equality testing. Security for lock-dependent messages still holds under computational assumptions on the message distributions produced by the attacker.

In both of our schemes the overhead in the length of the ciphertext is only additive and independent of the message length.

Keywords: Deduplication, convergent encryption, message-locked encryption.

1 Introduction

Deduplication, which eliminates redundant copies in user-provided data, is an important space-saving technique in communications and storage (see, for

⋆ This work was supported by NSF, the DARPA PROCEED program, an AFO SR MURI award, a grant from ONR, an IARPA project provided via DoI/NBC, and by Samsung. Opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA or IARPA. Distrib. Statement "A:" Approved for Public Release, Distribution Unlimited.

⋆⋆ Part of the work was done at Microsoft Research Silicon Valley.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 374–391, 2013.

example, [28,35,26]). Storage systems that rely on deduplication typically let the server have unfettered access to the clients' data. This set-up creates an obvious confidentiality problem, since the clients must trust the server with not only storing their documents but keeping them secret too.

The first solution for balancing confidentiality and efficiency in deduplication was described by Douceur et al. [18] and called *convergent encryption*. According to this deterministic scheme, a message is encrypted under a message-derived key (a hash of the message) so that identical plaintexts are mapped to identical ciphertexts. After encrypting the message, the client uploads the ciphertext to the server, retaining the hash to allow later decryption. In the meantime, the server can recognize equal ciphertexts, storing only one copy of each: if two clients happen to upload the same file, the resulting ciphertexts will be identical and can be deduplicated. The clients need not coordinate their actions and might not even be aware of each other's existence. Implementations and variants of convergent encryption followed [3,14,25,30,33,2,1] but their precise security guarantees were never fully proven or even stated.

Message-Locked Encryption. Recently, Bellare, Keelveedhi, and Ristenpart [7] brought much needed rigor into the area, by defining a new encryption primitive, Message-Locked Encryption (MLE), and several definitions that capture various aspects of MLE security. They also constructed and analyzed several schemes in their framework.

We briefly recall the definition of MLE and two security notions of privacy and tag integrity for MLE schemes introduced by Bellare et al. An MLE scheme encapsulates a standard (possibly randomized) symmetric-key encryption scheme where the encryption algorithm accepts a message m and a key k, and outputs a ciphertext c. The decryption algorithm reverses the process, recovering m from c given k. The scheme comes with a *key derivation algorithm* that, unlike a conventional key generation algorithm, is a deterministic function from m to k. It also includes a *tag-generation algorithm* that maps the ciphertext to a tag. Identical plaintexts result in equal tags. The corresponding ciphertexts, which may be randomized, are not necessarily equal. Tag integrity means that no computationally bounded adversary can trick the server into replacing a valid encryption with a ciphertext that does not decrypt to the same plaintext.

It is apparent that MLE, with its *deterministic* tag, cannot satisfy the standard notions of confidentiality (such as semantic security). Indeed, if the plaintexts can be feasibly enumerated, the adversary may always compute their tags and test them against that of the challenge ciphertext. A meaningful security guarantee can be achieved only if the input is sufficiently unpredictable. More concretely, in a CDA game (a chosen-*distribution* attack) the challenge consists either of a uniformly distributed string of bits or an encryption of a message drawn randomly from a distribution provided by the adversary. The security level is characterized by the distinguisher's running time, its advantage over a random guess, and the min-entropy of the distribution that the adversary is allowed to specify. A lower min-entropy requirement corresponds to a stronger security guarantee. This approach—basing security of the scheme on the assumption

of unpredictability of the plaintexts—is similar to the theory of deterministic public-key encryption initiated by the work of Bellare, Boldyreva, and O'Neill [4] (see also [6,10,5,12,19,24,32,29]).

Lock-Dependent Messages. In addition to the min-entropy requirement, there is another constraint on the adversarially chosen distribution of plaintexts implicit in the definition of MLE. If the adversary is allowed to specify a distribution of plaintexts, it may use the fact that the tags are deterministic for leaking unnecessary information on the messages (e.g., select a distribution that is concentrated on messages whose tags share a particular property, such as that they all start with a zero bit, or that the first bit of the tag reveals the first bit of the message). Doing so immediately gives the adversary a constant advantage in answering the challenge (of whether the output was a random string of bits or an encryption of a message drawn from the distribution). Similar attacks can be effective against *any* deterministic encryption scheme, where the adversary tailors the distribution to the scheme's public key. The common way of sidestepping this difficulty is to require that the distribution be chosen independently of the system parameters or, in the case of deterministic encryption, of the system's public key. More formally, the adversary must commit to the distribution of plaintexts before accessing the description of the system.

Since the parameters of the scheme are supposed to be publicly available, they must be included into the view of any realistic adversary. As soon as the adversary learns the parameters of the system and may influence, however indirectly, the distribution of plaintexts, the assumption of independence becomes false, voiding the security guarantees proven under this assumption.

In this paper we ask whether security guarantees can encompass also attacks that may depend on the scheme's parameters. Identifying the public parameters of an encryption scheme with a *lock*, we can paraphrase the problem addressed in this paper as follows:

Can message-locked encryption be secure for lock-dependent messages?

1.1 Our Contributions

In this paper we put forward two approaches for resolving this question in the affirmative, and provide schemes that are secure even for lock-dependent messages in realistic and rigorously defined adversarial models.

Our first approach is to avoid using tags that are derived deterministically from the messages. To this end, we design a fully randomized scheme that supports an equality-testing algorithm defined on the ciphertexts. We show that this enables us to satisfy a strong definition of security for an extension of the MLE notion, allowing the adversary to specify the distribution of the plaintexts adaptively, with no further restrictions on the distribution other than its min-entropy. Our construction is based on standard cryptographic tools in the random oracle model [8] and on a natural variant of Canetti's entropy-based DDH assumption [13]. The ciphertext overhead is only additive and polynomial in the security parameter.

Our second approach, on the other hand, continues using deterministic tags. Security for lock-dependent messages is guaranteed by limiting the computational power of the adversarial message distributions. (This approach is inspired by the recent work of Raghunathan, Segev, and Vadhan [29] who proposed a similar adversarial model for deterministic public-key encryption.) Specifically, in the random oracle model, we consider adversaries that are allowed to choose the distribution of plaintexts adaptively, after seeing the scheme's parameters, subject to the condition that the distribution be efficiently samplable using at most q queries to the random oracle, where q is a pre-determined parameter. Our construction can be based on any semantically secure encryption scheme. Its overhead, defined as the increase in the length of the ciphertext, is additive and depends only on the security parameter.

1.2 Paper Organization

In Section 2, we give a high-level overview of the fully randomized scheme and the deterministic scheme that we construct in this paper. In Section 3, we introduce a few preliminaries required to present our results. In Section 4, we formally define our notion of message-locked encryption for lock-dependent messages. In Section 5, we present the fully randomized scheme. In Section 7, we conclude and mention several interesting directions for further research. Because of space limitations all proofs and some definitions are deferred to the full version.

2 Overview of Our Schemes

In what follows we provide a high-level overview of the main ideas that underlie our schemes. Intuitively, constructing MLE schemes requires solving two technical challenges. We must design an algorithm that encrypts messages under a key that is highly correlated (via the key derivation algorithm) with the message and still remains secure. Secondly, the part of the ciphertext that allows the equality test must not leak any information about messages sampled from an adversarially chosen min-entropy distribution even given the public parameters.

Construction 1: A Fully Randomized Scheme. An encryption of a message m in our first scheme consists of three components: a "payload" which is an encryption of m using some underlying randomized encryption scheme, a tag, and a proof of consistency showing that the payload and the tag correspond to the same message. A tag for a message m is computed as $\tau = \left(g^r, g^{r \cdot h(m)}\right)$, where g is a generator of a bilinear group, h is a sufficiently strong collision-resistant function, and r is chosen uniformly at random. Given two tags $\tau_1 = (g_1, h_1)$ and $\tau_2 = (g_2, h_2)$, the equality-testing algorithm computes the pairings $\hat{e}(g_1, h_2)$ and $\hat{e}(g_2, h_1)$, which match if the tags were derived from the same message (or if a non-trivial collision was found for h). The fact that tags do not reveal any more information than is necessary for the scheme's functionality is based on combining a variant of Canetti's entropy-based DDH assumption [13], and the concept of seed-dependent condensers, recently introduced by Dodis, Ristenpart,

and Vadhan [17]. A similar idea for equality testing (without hashing) was explored by Yang et al. [34], who designed public-key encryption schemes that support equality testing but offer a significantly weaker notion of security (only one-wayness).

As for the payload and the consistency proof, a natural approach would be to simply encrypt m using its hash $h(m)$ as a key (as in [7]), and provide a NIZK proof of consistency. This approach, however, seems to fail as we must use an encryption scheme for which it is secure to encrypt a message m under the key $h(m)$. All existing constructions satisfying this property rely on the random oracle paradigm, which rules out using NIZK proofs as the language under consideration is no longer in NP.

We can resolve this issue with a cut-and-choose protocol applied to the encryption of the message. Naïvely, such a protocol would inflate the size of the ciphertext. However, a delicate combination of a secret-sharing scheme and a cut-and-choose technique enables us to realize an encryption scheme with a ciphertext overhead that is only additive and independent of the message length.

Specifically, the payload in our ciphertext consists of a randomized encryption $\mathsf{E}_s(m; r_1)$ of m under a uniformly chosen key s, a commitment $\mathsf{Commit}(s\|t)$ to s and a uniformly chosen key t, an ElGamal encryption $\left(g^{r_2}, g^{r_2 \cdot h(m)} \cdot t\right)$ of t using $h(m)$ as a key, and a circular-secure encryption $(r_3, H(r_3\|t) + s)$ of s using t as a key. The circular-secure encryption scheme is due to Black, Rogaway, and Shrimpton [9] whose proof of security assumes a random oracle H.

The only component that requires a random oracle is the circular-secure encryption of s using t. We can use a NIZK proof for proving that all other components (including the tag) are consistent with the same message m. In addition, we use a cut-and-choose protocol (which we collapse using a random oracle to a non-interactive one) for showing that the commitment $\mathsf{Commit}(s\|t)$ is consistent with the circular-secure encryption $(r_3, H(r_3\|t) + s)$, where s is encoded with a threshold secret-sharing scheme. The commitment $\mathsf{Commit}(s\|t)$ is used in both the NIZK proof and in the cut-and-choose components, and binds the two together to yield a proof of consistency for the entire ciphertext.

To ensure that the overhead of the scheme is additive and independent of the length of the message, first observe that the length of the commitment and the encryption of s under t (and hence the cut-and-choose part of the scheme) depend only on the security parameter. To further minimize the length of ciphertexts, we use a composition of an NIZK proof system with a succinct argument system in the random oracle model, where the length of the arguments depends only on the security parameter.

Construction 2: Deterministic encryption for computationally bounded distributions. As in the previous work [18,7], our second scheme uses any semantically secure randomized encryption $\mathsf{E}_k(m; r)$. It encrypts a message m using a key $k = k_m$ and randomness $r = r_m$ that are derived from m in a deterministic manner (e.g., using a hash function modeled as a random oracle).

With lock-dependent message distributions, however, such a scheme does not satisfy a meaningful notion of security since it is completely deterministic (as discussed above).

Following Raghunathan et al. [29] we show that this approach can be made secure even for lock-dependent message distributions, subject to the condition that these distributions are efficiently samplable using at most q queries to the random oracle, where q is a pre-determined parameter. (We do not ask for an a priori bound on the number of oracle calls that are made directly by the adversary.) Concretely, we derive the key k_m and the randomness r_m as $k_m = \oplus_{i=1}^{q+1} H_1(m\|i)$ and $r_m = \oplus_{i=1}^{q+1} H_2(m\|i)$, where H_1 and H_2 are two hash functions modeled as independent random oracles.

Intuitively, the proof of security relies on the fact that k_m and r_m are pseudorandom against *both* the adversary and the sampling circuits of such "q-bounded" message distributions. Pseudorandomness against the adversary relies on the fact that m is sampled with a super-logarithmic min-entropy and that the underlying encryption scheme is secure. Pseudorandomness against the sampling circuits relies on the fact that for learning any information on k_m or r_m it is essential to query the random oracle $q+1$ times.

3 Preliminaries

Notation. For an integer $n \in \mathbb{N}$ we denote by $[n]$ the set $\{1,\ldots,n\}$, by $[a,b]$ the set $\{a,a+1,\ldots,b\}$, and by U_n the uniform distribution over the set $\{0,1\}^n$. For a random variable X we denote by $x \leftarrow X$ the process of sampling a value x according to the distribution of X. Similarly, for a finite set S we denote by $x \leftarrow S$ the process of sampling a value x according to the uniform distribution over S. We denote by $\mathbf{x}$ (and sometimes $\boldsymbol{x}$) a vector $(x_1,\ldots,x_{|\mathbf{x}|})$. We denote by $\mathbf{X} = (X_1,\ldots,X_T)$ a joint distribution of T random variables, and by $\mathbf{x} = (x_1,\ldots,x_T)$ a sample drawn from $\mathbf{X}$. For two bit-strings x and y we denote by $x\|y$ their concatenation. A non-negative function $f\colon \mathbb{N} \to \mathbb{R}$ is negligible if it vanishes faster than any inverse polynomial.

Entropy. The *min-entropy* of a random variable X is defined as $\mathbf{H}_\infty(X) = -\log(\max_x \Pr[X=x])$. A *$k$-source* is a random variable X with $\mathbf{H}_\infty(X) \geq k$. A *$(k_1,\ldots,k_T)$-source* is a random variable $\mathbf{X} = (X_1,\ldots,X_T)$ where each X_i is a k_i-source. A *(T,k)-source* is a random variable $\mathbf{X} = (X_1,\ldots,X_T)$ where for each $i \in [T]$, it holds that X_i is a k-source. A *(T,k)-block-source* is a random variable $\mathbf{X} = (X_1,\ldots,X_T)$ where for every $i \in [T]$ and $x_1,\ldots,x_{i-1}$ it holds that $X_i|_{X_1=x_1,\ldots,X_{i-1}=x_{i-1}}$ is a k-source. The *statistical distance* between two random variables X and Y over a finite domain Ω is $\mathbf{SD}(X,Y) = \frac{1}{2}\sum_{\omega\in\Omega} |\Pr[X=\omega] - \Pr[Y=\omega]|$.

The ME-DDH Assumption. We state a variant of Canetti's entropy DDH assumption [13]. The β-min-entropy DDH assumption (abbreviated as ME-DDH) states that for a group $\mathbb{G}$ equipped with a non-degenerate bilinear map $\hat{e}\colon \mathbb{G}\times\mathbb{G}\to\mathbb{G}_T$, of prime order p (where p is a λ-bit prime) for any distribution X over $\mathbb{Z}_p$ with $\mathbf{H}_\infty(X) \geq \beta$, for uniformly sampled $a,c \leftarrow \mathbb{Z}_p$ and $b \leftarrow X$, it

holds that the two distributions $\left(g, g^a, g^{ab}\right)$ and (g, g^a, g^c) are computationally indistinguishable. We make two remarks on the ME-DDH assumption:

1. We require $\beta \geq \omega(\log \lambda)$ for the assumption to be plausible. Otherwise, there exists an $x^* \leftarrow X$ such that $\Pr[X = x^*]$ is non-negligible and a distinguisher that, when given (g, g^a, g^c), checks to see whether $(g^a)^{x^*} = g^c$, succeeds in distinguishing the two distributions with non-negligible probability.
2. If X is the uniform distribution over $\mathbb{Z}_p$, then the assumption is *unconditionally true* as the two distributions ab and c are identical even given a.

4 MLE for Lock-Dependent Messages

Extending the work of Bellare et al. [7], we propose a more general notion of the primitive MLE, which we call MLE2. In MLE2, we allow tags to be randomized and consider a definition of tag correctness that introduces a new polynomial-time algorithm EQ that subsumes the functionality of deterministic tags. In addition, we introduce a new *validity-test* algorithm, denoted Valid, that allows anyone with the public parameters to check if a given ciphertext is a valid ciphertext. In the context of using MLE2 for secure deduplication, EQ allows for deduplication of ciphertexts and Valid allows the server to reject adversarially constructed ciphertexts that subvert deduplication to replace a valid ciphertext with an invalid one that does not decrypt correctly.

The main benefit of the new notion is permitting a stronger security requirement (which we denote $\mathsf{PRV\text{-}CDA2}$) that allows the adversary to see the public parameters *before* issuing oracle queries.

MLE2. A message-locked encryption scheme for lock-dependent messages is a six-tuple $\Pi = (\mathsf{PPGen}, \mathsf{KD}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{EQ}, \mathsf{Valid})$ operating over plaintext space $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda \in \mathbb{N}}$, ciphertext space $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$, and keyspace $\mathcal{K} = \{\mathcal{K}_\lambda\}_{\lambda \in \mathbb{N}}$ of polynomial-time randomized algorithms with the following properties:

- The parameter generation algorithm takes as input 1^λ and returns public parameters pp.
- The key derivation function KD takes as input public parameters pp, a message m, and outputs a message-derived key $k_m \leftarrow \mathsf{KD}_{\mathsf{pp}}(m)$.
- The encryption algorithm Enc takes as input public parameters pp, a message m, and a message-derived key k_m. It outputs a ciphertext $c \leftarrow \mathsf{Enc}_{\mathsf{pp}}(k_m, m)$.
- The decryption algorithm Dec takes as input public parameters pp, ciphertext c, and a secret key k and outputs either a message m or $\perp$.
- The (new) equality algorithm EQ takes as input public parameters pp, and two ciphertexts c_1 and c_2 and outputs 1 if both ciphertexts are generated from the same underlying message.
- The (new) validity-test algorithm Valid takes as input public parameters pp and a ciphertext c and outputs 1 if the ciphertext c is a valid ciphertext.

Bellare et al. [7] considered the notion of an equality-checking *tag* analogous to our notion of an equality algorithm EQ. A (publicly computable) tag-generation

algorithm, on input a ciphertext c, produces a tag such that if two ciphertexts c_1 and c_2 are generated from the same message, the corresponding tags are equal with high probability. Our equality algorithm is a generalization of such an equality-checking tag. Given any scheme with equality-checking tags, we can describe a simple algorithm EQ that given c_1 and c_2 derives their respective tags and outputs 1 only if the tags are equal.

The notion analogous to *tag correctness* of Bellare et al. [7] requires that for all $\lambda \in \mathbb{N}$, all public parameters $\mathsf{pp} \leftarrow \mathsf{PPGen}(1^\lambda)$, and all messages $m \in \mathcal{M}$, there is a negligible function $\nu(\lambda)$ such that for two encryptions c_1 and c_2 of m with $\mathsf{KD}_{\mathsf{pp}}(m)$ and independent random coins, it holds that $\mathsf{EQ}_{\mathsf{pp}}(c_1, c_2) = 1$ with probability at least $1 - \nu(\lambda)$, where the probability is taken over random coins of all algorithms.

The notion of correctness for the validity-test algorithm Valid requires that for all $\lambda \in \mathbb{N}$, all public parameters $\mathsf{pp} \leftarrow \mathsf{PPGen}(1^\lambda)$, and all messages $m \in \mathcal{M}$, there is a negligible function $\nu(\lambda)$ such that for a ciphertext $c \leftarrow \mathsf{Enc}_{\mathsf{pp}}(\mathsf{KD}_{\mathsf{pp}}(m), m)$, $\Pr[\mathsf{Valid}_{\mathsf{pp}}(c) = 1] \geq 1 - \nu(\lambda)$, where the probability is taken over all random coins of all algorithms.

The usual notion of correctness of the decryption algorithm Dec applies. Specifically, for all $\lambda \in \mathbb{N}$, all public parameters $\mathsf{pp} \leftarrow \mathsf{PPGen}(1^\lambda)$, and all messages $m \in \mathcal{M}$, there is a negligible function $\nu(\lambda)$ such that

$$\Pr[\mathsf{Dec}_{\mathsf{pp}}(k_m, \mathsf{Enc}_{\mathsf{pp}}(k_m, m)) = m \mid k_m \leftarrow \mathsf{KD}_{\mathsf{pp}}(m)] \geq 1 - \nu(\lambda),$$

where the probability is taken over all random coins of all algorithms.

MLE2 Adversaries. To capture a notion of security against an adversary that attacks the system by choosing messages that may depend on the public parameters, we introduce several adversary models. In what follows, we consider several parameters that are functions of the security parameter; $q = q(\lambda)$ denoting the number of random oracle queries, $k = k(\lambda)$ denoting min-entropy requirements over message sources, $T = T(\lambda)$ denoting the number of blocks in the message source, and $\Gamma = \Gamma(\lambda)$ denoting the size of a circuit that generates message sources.

In particular, inspired by recent work on deterministic encryption [29], for $\mathsf{X} \in \{(T,k)\text{-block}, (T,k)\}$ we define the class of Γ-sampling complexity X-source adversaries and a generalization to polynomial-size X-source adversaries. We stress that all algorithms are allowed polynomially many calls to the random oracle in the security definitions that follow. Additionally, in schemes that rely random oracles, we define q-query X-source adversaries. Although more restrictive, they are useful in constructing efficient and practical deterministic encryption schemes secure in the random oracle model [29]. We begin by introducing a definition of the real-or-random encryption oracle used in definitions of security.

Definition 4.1 (Real-or-Random Encryption Oracle). *The real-or-random encryption oracle,* RoR*, takes as input triplets of the form* $(\mathsf{mode}, \mathsf{pp}, \mathbf{M})$*, where* $\mathsf{mode} \in \{\mathsf{real}, \mathsf{rand}\}$*,* pp *denotes public parameters, and* $\mathbf{M}$ *is a polynomial size circuit representing a joint distribution over* T *messages. If* $\mathsf{mode} = \mathsf{real}$ *then the*

oracle samples $(m_1, \ldots, m_T) \leftarrow \mathbf{M}$*, and if* $\mathsf{mode} = \mathsf{rand}$ *then the oracle samples uniform and independent messages* $m_1, \ldots, m_T \leftarrow \mathcal{M}$*. Next, for each* $i \in [T]$*, it samples* $k_i \leftarrow \mathsf{KD}_{\mathsf{pp}}(m_i)$*, computes* $c_i \leftarrow \mathsf{Enc}_{\mathsf{pp}}(k_i, m_i)$ *and outputs the ciphertext vector* $(c_1, \ldots, c_T)$*.*

Definition 4.2 (Γ-sampling Complexity Adversary). *Consider an* X*-source where* $\mathsf{X} \in \{(T,k)\text{-}block, (T,k)\}$*. Let* $\mathcal{A}$ *be a probabilistic polynomial-time algorithm that is given as input a pair* $(1^\lambda, \mathsf{pp})$ *and oracle access to* $\mathsf{RoR}(\mathsf{mode}, \mathsf{pp}, \cdot)$ *for some* $\mathsf{mode} \in \{\mathsf{real}, \mathsf{rand}\}$*. Then,* $\mathcal{A}$ *is a* Γ-sampling complexity X-source adversary *if for each of* $\mathcal{A}$*'s* RoR *queries* $\mathbf{M}$ *it holds that* $\mathbf{M}$ *is an* X*-source that is samplable by a circuit of size at most* Γ*. In addition, for the case of* (T,k)*-source adversaries, we require that for each such query* $\mathbf{M}$ *it holds that* $M_i \neq M_j$ *for all vectors* $(M_1, \ldots, M_T)$ *in the support of* $\mathbf{M}$ *and for all* $i \neq j \in [T]$*.*

We consider a stronger adversary that has no *a-priori* bound on the sampling complexity of its queries except that they are efficiently samplable by polynomial size circuits. Such an adversary subsumes Γ-sampling complexity adversaries for all $\Gamma = \mathrm{poly}(\lambda)$.

Definition 4.3 (Polynomial-Sampling Complexity Adversary). *Let* $\mathsf{X} \in \{(T,k)\text{-}block, (T,k)\}$*, and let* $\mathcal{A}$ *be a probabilistic polynomial-time algorithm that is given as input a pair* $(1^\lambda, \mathsf{pp})$ *and oracle access to* $\mathsf{RoR}(\mathsf{mode}, \mathsf{pp}, \cdot)$ *for some* $\mathsf{mode} \in \{\mathsf{real}, \mathsf{rand}\}$*. Then,* $\mathcal{A}$ *is a* polynomial-size X-source adversary *if for each of* $\mathcal{A}$*'s* RoR*-queries* $\mathbf{M}$ *it holds that* $\mathbf{M}$ *is an* X*-source that is samplable by a circuit of (an arbitrary) polynomial size in the security parameter.*

Definition 4.4 (q-query Adversary [29]). *Consider an* X*-source where* $\mathsf{X} \in \{(T,k)\text{-}block, (T,k)\}$*. Let* $\mathcal{A}$ *be a probabilistic polynomial-time algorithm that is given as input a pair* $(1^\lambda, \mathsf{pp})$ *and oracle access to* $\mathsf{RoR}(\mathsf{mode}, \mathsf{pp}, \cdot)$ *for some* $\mathsf{mode} \in \{\mathsf{real}, \mathsf{rand}\}$*. Then,* $\mathcal{A}$ *is a* q-query k-source adversary *if for each of* $\mathcal{A}$*'s* RoR*-queries* $\mathbf{M}$ *it holds that* $\mathbf{M}$ *is an* X*-source that is samplable by a polynomial-size circuit that uses at most* q *queries to the random oracle.*

A Stronger Notion of Message Privacy: PRV-CDA2. We define the following security notion with respect to polynomial-size X-source adversaries (see Definition 4.3), which we denote X-source PRV-CDA2 security. A simple modification to the experiments in the security definition allows us to restrict our class of adversaries to Γ-sampling complexity or q-query X-source adversaries. Such notions are referred to as Γ-sampling complexity or q-query X-source PRV-CDA2.

Definition 4.5 (PRV-CDA2 Security). *An MLE2 scheme* $\Pi = (\mathsf{PPGen}, \mathsf{KD}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{EQ}, \mathsf{Valid})$ *is* X-source PRV-CDA2 secure*, for* $\mathsf{X} \in \{(T,k)\text{-}block, (T,k)\}$*, if for any probabilistic polynomial-time polynomial-size* X*-source adversary* $\mathcal{A}$*, there exists a negligible function* $\nu(\lambda)$ *such that*

$$\mathbf{Adv}^{\mathsf{PRV\text{-}CDA2}}_{\Pi,\mathcal{A}}(\lambda) \stackrel{\mathsf{def}}{=} \left| \Pr\left[\mathsf{Expt}^{\mathsf{real}}_{\Pi,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{rand}}_{\Pi,\mathcal{A}}(\lambda) = 1\right] \right| \leq \nu(\lambda),$$

PRV-CDA2 game: $\mathsf{Expt}^{\mathsf{mode}}_{\Pi,\mathcal{A}}(\lambda)$	TC2/STC2 game: $\mathsf{Expt}^{\mathsf{Z}}_{\Pi,\mathcal{A}}(\lambda)$
1. $\mathsf{pp} \leftarrow \mathsf{PPGen}(1^\lambda)$.	1. $\mathsf{pp} \leftarrow \mathsf{PPGen}(1^\lambda)$.
2. $b \leftarrow \mathcal{A}^{\mathsf{RoR}(\mathsf{mode},\mathsf{pp},\cdot)}(1^\lambda, \mathsf{pp})$.	2. $(m, c') \leftarrow \mathcal{A}(1^\lambda, \mathsf{pp})$.
3. Output b.	3. **If** $m = \perp$ **or** $\mathsf{Valid}(c') = 0$ output 0.
	4. $k \leftarrow \mathsf{KD}_{\mathsf{pp}}(m)$.
	5. $c \leftarrow (\mathsf{Enc}_{\mathsf{pp}}(k, m))$ **and** $m' \leftarrow \mathsf{Dec}_{\mathsf{pp}}(k, c')$.
	6. **If** $\mathsf{Z} = \mathsf{TC2}$, $\mathsf{EQ}(c, c') = 1$, $m \neq m'$, **and** $m' \neq \perp$, output 1.
	7. **If** $\mathsf{Z} = \mathsf{STC2}$, $\mathsf{EQ}(c, c') = 1$, **and** $m \neq m'$, output 1.
	8. **Else**, output 0.

Fig. 1. Security games for Definitions 4.5 and 4.7.

where for each $\mathsf{mode} \in \{\mathsf{real}, \mathsf{rand}\}$ *and* $\lambda \in \mathbb{N}$ *the experiment* $\mathsf{Expt}^{\mathsf{mode}}_{\Pi,\mathcal{A}}(\lambda)$ *is defined in Figure 1. In addition, such a scheme is* one-time *secure if the above holds for any adversary* $\mathcal{A}$ *that queries the* RoR *oracle at most once.*

The assumption of the plaintexts' unpredictability and support for equality testing (for use in the context of deduplication) may appear to be at odds with each other. After all, a distribution of plaintexts with sufficiently large min-entropy cannot possibly benefit from deduplication as the number of clones in a moderately sized sample is going to be negligible. However, the definition does not presuppose a particular generative model for the plaintexts. Instead, it bounds from below the amount of uncertainty that the adversary has about a particular plaintext, or in the language of Bayesian probability theory, the adversary's *prior*. In other words, Alice and Bob may share the same document that can be deduplicated on the server and will stay private as long as the server cannot guess its exact content.

Our parameter-dependent security notion enables an immediate reduction of "multi-shot" adversaries to "single-shot" ones, as is standard in public-key encryption schemes. Theorem 4.6 stated below follows via a standard hybrid argument.

Theorem 4.6 (Equivalence of PRV-CDA2 and One-time PRV-CDA2 Security). *Let* $k = k(\lambda)$, $T = T(\lambda)$, *and* $\mathsf{X} \in \{(T,k)\text{-}block, (T,k)\}$. *Then, an MLE2 scheme is* X*-source* PRV-CDA2*-secure if and only if it is* one-time X*-source* PRV-CDA2*-secure.*

Definition 4.7 (Tag Consistency). *An MLE2 scheme* $\Pi = (\mathsf{PPGen}, \mathsf{KD}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{EQ}, \mathsf{Valid})$ *is tag consistent (resp., strongly tag consistent) if for any probabilistic polynomial-time adversary* $\mathcal{A}$*, there exists a negligible function* $\nu(\lambda)$ *such that* $\mathbf{Adv}^{\mathsf{expt}}_{\Pi,\mathcal{A}}(\lambda) \stackrel{\mathsf{def}}{=} \Pr\left[\mathsf{Expt}^{\mathsf{expt}}_{\Pi,\mathcal{A}}(\lambda) = 1\right] \leq \nu(\lambda)$, *where* $\mathsf{expt} = \mathsf{TC2}$ *(resp.,* $\mathsf{expt} = \mathsf{STC2}$*) and for each* $\mathsf{Z} \in \{\mathsf{TC2}, \mathsf{STC2}\}$, $\lambda \in \mathbb{N}$ *the experiment* $\mathsf{Expt}^{\mathsf{Z}}_{\Pi,\mathcal{A}}(\lambda)$, *is defined in Figure 1.*

Block-Source Adversaries vs. Single-Message Adversaries. In the somewhat similar setting of deterministic public-key encryption, Boldyreva et al. [10] showed that for proving security against (T, k)-block-source adversaries it suffices to prove security against $(1, k)$-source adversaries. Their proof, however, does not seem to carry over to our setting, where all message distributions are required to be efficiently samplable by polynomial-sized circuits. Nevertheless, motivated by the works of Bellare et al. [7] and Brakerski and Segev [12], we now present a strengthening of our notion of PRV-CDA2 security, for which we are able to prove an equivalence between security against (T, k)-block-source adversaries and security against $(1, k)$-source adversaries.

The strengthened notion, to which we refer as aux-PRV-CDA2 security (i.e., PRV-CDA2 security with auxiliary inputs), is obtained by modifying the real-or-random encryption oracle. The modification is that its inputs are now of the form $(\mathsf{mode}, \mathsf{pp}, (\mathbf{M}, \mathsf{Aux}))$, where $(\mathbf{M}, \mathsf{Aux})$ is a joint distribution over messages and auxiliary inputs. If $\mathsf{mode} = \mathsf{real}$ then the oracle samples $(m_1, \ldots, m_T, \mathsf{aux}) \leftarrow (\mathbf{M}, \mathsf{Aux})$, and if $\mathsf{mode} = \mathsf{rand}$ then the oracle samples uniform and independent messages $m_1, \ldots, m_T \leftarrow \mathcal{M}$ and $\mathsf{aux} \leftarrow \mathsf{Aux}$, independent of the messages. Next, for each $i \in [T]$, it samples $k_i \leftarrow \mathsf{KD}_{\mathsf{pp}}(m_i)$, computes $c_i \leftarrow \mathsf{Enc}_{\mathsf{pp}}(k_i, m_i)$ and outputs the vector $(c_1, \ldots, c_T, \mathsf{aux})$. For $\mathsf{X} \in \{(T, k)\text{-block}, (T, k)\}$, we say that a probabilistic polynomial-time algorithm $\mathcal{A}$ is a *polynomial-size* X*-source adversary* if for each of $\mathcal{A}$'s RoR-queries $(\mathbf{M}, \mathsf{Aux})$ it holds that: (1) the joint distribution $(\mathbf{M}, \mathsf{Aux})$ is samplable by a circuit of polynomial size, and (2) for every auxiliary input aux in the support of Aux, it holds that $\mathbf{M}|_{\mathsf{Aux}=\mathsf{aux}}$ is an X-source. Equipped with this modification, we prove the following theorem:

Theorem 4.8 (Equivalence of (T, k)-block-source and $(1, k)$-source adversaries with auxiliary inputs). *Let $k = k(\lambda)$ and $T = T(\lambda)$ be polynomial in λ. Then, an MLE2 scheme is (T, k)-block-source* aux-PRV-CDA2*-secure if and only if it is $(1, k)$-source* aux-PRV-CDA2*-secure.*

A Comparison to the Security Notion of Bellare et al. [7]. In the security notion of MLE [7], adversaries are not given access to the public parameters pp when interacting with the RoR encryption oracle (unlike in step 2 in our definition). Adversaries receive pp only after all queries to the RoR oracle are completed. (Once pp is published, subsequent oracle queries return $\perp$.) Our security notion of MLE2 considers adversaries that are given access to the public parameters when interacting with the RoR encryption oracle. In particular, this enables adversaries to query the oracle with message distributions that depend on the public parameters pp (in a bounded manner, as described in the various adversary notions defined above).

The security notions of both MLE and MLE2 consider message distributions that are (T, k)-sources. All the MLE constructions of Bellare et al. are secure for (T, k)-sources, and our deterministic MLE2 construction for q-query adversaries is secure for (T, k)-sources as well (for any polynomial $T = T(\lambda)$). However, our fully randomized construction is secure only for k-sources (that is, for $(1, k)$-sources). This limitation seems to be inherent to our approach, which uses

seed-dependent condensers. (See the work of Dodis et al. [17] for a discussion on the limitations of seed-dependent condensers in the presence of auxiliary inputs).

The notions of TC2/STC2 tag consistency described above follow closely the definitions of Bellare et al., with small modifications to accommodate the more general notion of tags and the new algorithms EQ and Valid. In particular, the experiment outputs 1 only if $\mathsf{EQ}(c, c') = 1$, which corresponds to comparing tags in MLE. Additionally, we discard adversarially constructed ciphertexts c' that can be recognized by algorithm Valid as *invalid*.

5 A Fully Randomized Scheme

In this section, we present the scheme Π_{full}, a fully randomized MLE2 scheme. An overview of the construction is presented in Section 2.

The Scheme. Let λ denote the security parameter. Let GroupGen be a probabilistic polynomial-time algorithm that takes as input a security parameter 1^λ, and outputs $(\mathbb{G}, \mathbb{G}_{\mathrm{T}}, p, g, \hat{e})$ where $\mathbb{G}$ and $\mathbb{G}_{\mathrm{T}}$ are groups of prime order p, $\mathbb{G}$ is generated by g, p is a λ-bit prime number, and $\hat{e}\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_{\mathrm{T}}$ is a non-degenerate efficiently computable bilinear map. The scheme is parameterized by a parameter n that is polynomial in the security parameter. The MLE2 scheme Π_{full} comprises the following building blocks.

- A one-time secure symmetric-key encryption scheme $\mathcal{SE} = (\mathsf{K}, \mathsf{E}, \mathsf{D})$. As a concrete example, let $G\colon \mathcal{K} \to \mathcal{M}$ be a pseudorandom generator that takes short keys and expands them to the message space. We can use such a PRG as a one-time pad to get a simple, efficient, and one-time secure scheme $\mathsf{E}_k(m) := G(k) \oplus m \in \mathcal{M}$.
- An $(n+1)$-out-of-$(2n+1)$ secret sharing of a key $k \in \mathcal{K}$. The secret is encoded as an element of the field $\mathbb{F}_q$ for a prime q slightly larger than $|\mathcal{K}|$. The additive secret-sharing scheme we use (based on interpolating polynomials) also satisfies the additional property that given $2n+1$ shares of a secret, one can efficiently reconstruct (via Reed-Solomon decoding techniques [23]) the secret as long as at least $\lceil (3n+1)/2 \rceil$ shares are correctly computed.
- Two hash functions $\mathsf{RO}\colon \{0,1\}^* \to \mathbb{F}_q$ and $\mathsf{FS}\colon \{0,1\}^* \to \mathfrak{p}^{n,2n+1}$. The functions will be modeled in the proof of security as random oracles. RO is used to break circularity and FS denotes the random oracle required to implement Fiat-Shamir. Here $\mathfrak{p}^{n,2n+1}$ denotes the set of all subsets of $[2n+1]$ of cardinality n.
- A collection $\mathcal{H} = \{\mathcal{H}_\lambda\}_{\lambda \in \mathbb{N}}$ of collision-resistant hash functions $h\colon \mathcal{M} \to \mathbb{Z}_p$.
- A commitment scheme $\mathcal{TC} = (\mathsf{CGen}, \mathsf{Commit}, \mathsf{Reveal})$.
- A simulation-sound non-interactive extractable zero-knowledge proof system $\mathcal{ZK} = (\mathsf{ZKGen}, \mathsf{ZKProve}, \mathsf{ZKVer}, \mathsf{ZKFakeGen}, \mathsf{ZKSim}, \mathsf{ZKExt})$ for the NP language $\mathcal{L}$ defined at the end of the description of the scheme.

The scheme $\Pi_{\mathsf{full}} = (\mathsf{PPGen}, \mathsf{KD}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{EQ}, \mathsf{Valid})$ is parameterized by a parameter n that is polynomial in the security parameter and is as follows:

– **Parameter-generation algorithm:** On input 1^λ, algorithm PPGen samples $(\mathbb{G}, \mathbb{G}_\mathrm{T}, p, g, \hat{e}) \leftarrow \mathsf{GroupGen}(1^\lambda)$. It chooses a hash function $h \leftarrow \mathcal{H}$ from the family of collision-resistant hash functions, and specifies two additional hash functions RO and FS. It generates public parameters $\mathsf{pp}_{\mathsf{ZK}} \leftarrow \mathsf{ZKGen}(1^\lambda)$ and $\mathsf{pp}_{\mathsf{com}} \leftarrow \mathsf{CGen}(1^\lambda)$ and publishes $\mathsf{pp} = (\mathbb{G}, \mathbb{G}_\mathrm{T}, p, g, \hat{e}, h, \mathsf{pp}_{\mathsf{ZK}}, \mathsf{pp}_{\mathsf{com}})$.
– **Key-derivation function:** KD takes as input public parameters pp, a message m, and outputs the message-derived key $k_m = h(m)$.
– **Encryption algorithm:** Enc takes as input public parameters pp, a message m, and a message-derived key k_m. It samples $r \leftarrow \mathbb{Z}_p$ and first computes $\tau = \left(g^r, g^{r \cdot h(m)}\right) \in \mathbb{G}^2$.
Creating shares of s: The algorithm chooses a key $s \leftarrow \mathsf{K}(1^\lambda)$ for scheme $\mathcal{SE}$ and an $(n+1)$-out-of-$(2n+1)$ additive secret sharing of s denoted $\vec{s} = (s_1, \ldots, s_{2n+1}) \in \mathbb{F}_q^{2n+1}$. It computes the encrypted message $d = \mathsf{E}_s(m) \in \mathcal{C}$.
Committing to deferring elements t_i: The algorithm samples $t_1, \ldots, t_{2n+1} \leftarrow \mathbb{G}$ and lets com_i denote the commitment $\mathsf{Commit}(s_i \| t_i)$. We let $\vec{\mathsf{com}} = (\mathsf{com}_1, \ldots, \mathsf{com}_{2n+1})$ and $\vec{t} = (t_1, \ldots, t_{2n+1})$.
Encrypting deferring elements t_i: The algorithm samples random elements $u_1, \ldots, u_{2n+1} \leftarrow \mathbb{Z}_p$ and computes ElGamal encryptions of t_i with public key $g^{h(m)}$. Let $\mathsf{et}_i = \left(g^{u_i}, g^{u_i \cdot h(m)} \cdot t_i\right)$. We let $\vec{\mathsf{et}} = (\mathsf{et}_1, \ldots, \mathsf{et}_{2n+1})$ and $\vec{u} = (u_1, \ldots, u_{2n+1})$.
Encrypting shares of s: The algorithm encrypts s_i under t_i with a construction by Black et al. [9]. The algorithm samples $v_1, \ldots, v_{2n+1} \leftarrow \{0,1\}^\lambda$ and sets es_i to the ciphertext $(v_i, \mathsf{RO}(v_i \| t_i) + s_i \bmod q)$. We let $\vec{\mathsf{es}}$ denote $(\mathsf{es}_1, \ldots, \mathsf{es}_{2n+1})$.
Zero-knowledge proof: The algorithm computes a proof π using algorithm $\mathsf{ZKProve}$ that the statement $\sigma = (d, \vec{\mathsf{com}}, \vec{\mathsf{et}}, \tau)$ is in the language $\mathcal{L}$ defined below.
Cut-and-choose: The algorithm computes $X \leftarrow \mathsf{FS}(\sigma \| \pi \| \vec{\mathsf{es}})$ where $X \subset [2n+1]$ of cardinality n. The algorithm reveals commitments com_i for $i \in X$, denoted by $\vec{\mathsf{rcom}} = \{\mathsf{Reveal}(\mathsf{com}_i)\}_{i \in X}$.
The algorithm outputs:

$$c = \left(d, \vec{\mathsf{com}}, \vec{\mathsf{et}}, \vec{\mathsf{es}}, \pi, \vec{\mathsf{rcom}}, \tau\right).$$

– **Validity test:** On input a ciphertext $c = \left(d, \vec{\mathsf{com}}, \vec{\mathsf{et}}, \vec{\mathsf{es}}, \pi, \vec{\mathsf{rcom}}, \tau\right)$, algorithm Valid constructs $\sigma = \left(d, \vec{\mathsf{com}}, \vec{\mathsf{et}}, \tau\right)$. If $\mathsf{ZKVer}(\mathsf{pp}_{\mathsf{ZK}}, \sigma, \pi) = 0$, algorithm Valid outputs 0. Next, Valid computes $X = \mathsf{FS}(\sigma \| \pi \| \vec{\mathsf{es}}) \in \mathfrak{p}^{n, 2n+1}$ and verifies for revealed values $\{\mathsf{rcom}_i\}_{i \in X}$ from $\vec{\mathsf{rcom}}$ that the commitments and encryptions of $\vec{s}$, $\{\mathsf{com}_i, \mathsf{es}_i\}_{i \in X}$, are consistent with opened values $\{s_i, t_i\}_{i \in X}$. It outputs 1 if they are consistent and 0 otherwise.
– **Decryption algorithm:** On input the public parameters of the system pp, the ciphertext $c = \left(d, \vec{\mathsf{com}}, \vec{\mathsf{et}}, \vec{\mathsf{es}}, \pi, \vec{\mathsf{rcom}}, \tau\right)$, and a secret key k_m, if $\mathsf{Valid}(c) = 0$, the decryption algorithm outputs $\perp$. Else, the decryption algorithm first recovers t_i from $\mathsf{et}_i = (\alpha, \beta)$ with secret key k_m by computing $t_i = \beta / \left(\alpha^{k_m}\right)$. Next, using t_i, the algorithm recovers s_i from $\mathsf{es}_i = (\alpha, \beta)$ by

computing $s_i = \beta - \mathsf{RO}(\alpha \| t_i) \bmod q$. It reconstructs s, given the $(n+1)$-out-of-$(2n+1)$ additive secret sharing of s, $(s_1, \ldots, s_{2n+1})$. Finally, the decryption algorithm outputs $m \leftarrow \mathsf{D}_s(d)$.

– **Equality-testing algorithm:** On input two ciphertexts, c_1 and c_2, the algorithm recovers τ_1 and τ_2. Let $\tau_1 = (g_1, h_1) \in \mathbb{G}^2$ and $\tau_2 = (g_2, h_2) \in \mathbb{G}^2$. The algorithm outputs 1 if and only if $\hat{e}(g_1, h_2) = \hat{e}(g_2, h_1)$.

The Language $\mathcal{L}$ and Relation $\mathcal{R}$. Intuitively, the language $\mathcal{L}$ contains only statements $\sigma = (d, \vec{\mathsf{com}}, \vec{\mathsf{et}}, \tau)$ whose components are created with the secret values $(m, r, s, \vec{t}, \vec{u})$ in a consistent manner. More formally, we define the relation $\mathcal{R} = \{(\sigma, w)\}$ of statements σ and corresponding proof strings w below and note that $\mathcal{L} = \{\sigma \ : \ \exists\, w \text{ s.t. } (\sigma, w) \in \mathcal{R}\}$:

$$\mathcal{R} := \left\{ \left(\left(d, \vec{\mathsf{com}}, \vec{\mathsf{et}}, \tau\right), \left(\vec{s}, \vec{t}, m\right) \right) \;\middle|\; \begin{array}{r} d = \mathsf{E}_s(m) \\ \mathsf{com}_i = \mathsf{Commit}(s_i \| t_i)\ \forall\ i \in [2n+1] \\ \mathsf{et}_i = \left(g^{u_i}, g^{u_i \cdot h(m)}\right) \text{ for uniform } u_i \in \mathbb{Z}_p \\ \tau = \left(g^r, g^{r \cdot h(m)}\right) \text{ for uniform } r \in \mathbb{Z}_p \end{array} \right\}.$$

Correctness of the Scheme Π_{full}. Consider a ciphertext $c \leftarrow \mathsf{Enc}_{\mathsf{pp}}(h(m), m)$ with components $(d, \vec{\mathsf{com}}, \vec{\mathsf{et}}, \vec{\mathsf{es}}, \pi, \vec{\mathsf{rcom}}, \tau)$ and the secret key $k_m = h(m)$. If $\mathsf{et}_i = (\alpha, \beta)$, then we have $\beta / (\alpha^{k_m}) = g^{u_i \cdot h(m)} \cdot t_i / (g^{u_i})^{h(m)} = t_i$ as required. Next, if $\mathsf{es}_i = (\alpha, \beta)$, we have $\beta - \mathsf{RO}(\alpha \| t_i) = \mathsf{RO}(v_i \| t_i) + s_i - \mathsf{RO}(v_i \| t_i) = s_i \pmod q$ as required. The secret-sharing scheme correctly reconstructs s given shares $(s_1, \ldots, s_{2n+1})$ (via Reed-Solomon decoding techniques [23]) and therefore correctness of the scheme follows from correctness of the symmetric encryption scheme $\mathcal{SE}$.

Correctness of algorithm EQ follows from properties of groups equipped with bilinear maps. If $\tau_1 = (\alpha_1, \beta_1) \in \mathbb{G}^2$ and $\tau_2 = (\alpha_2, \beta_2) \in \mathbb{G}^2$ are constructed by the encryption scheme with the same underlying message m, then

$$\hat{e}\left(\alpha_1, \beta_2\right) = \hat{e}\left(g^{r_1}, g^{r_2 \cdot h(m)}\right) = \hat{e}\left(g, g\right)^{r_1 r_2 \cdot h(m)}, \text{ and}$$

$$\hat{e}\left(\alpha_2, \beta_1\right) = \hat{e}\left(g^{r_2}, g^{r_1 \cdot h(m)}\right) = \hat{e}\left(g, g\right)^{r_1 r_2 \cdot h(m)} \text{ as required.}$$

Succinct Ciphertexts. In order to shrink the ciphertexts in the scheme Π_{full} to be of length $|\mathsf{E}_s(m)| + \mathrm{poly}(\lambda)$, we replace the (long) NIZK proof π in our ciphertext with a non-interactive succinct extractable argument system whose length depends only on the security parameter. (Such argument systems are known to exist in the random oracle model—see the full version for the definition and instantiation.)

Specifically, our parameter generation algorithm outputs additionally the public parameters $\mathsf{pp}_{\mathsf{SA}}$ for the argument system. The encryption scheme first computes an NIZK proof π for the statement $\sigma = (d, \vec{\mathsf{com}}, \vec{\mathsf{et}}, \tau)$, and then uses π as a witness for asserting (with a succinct proof π_{SA}), using the succinct argument system, that there exists a proof π that is accepted by the verifier of the NIZK system for the assertion that $(\sigma, \pi) \in \mathcal{R}$. Finally, we discard the NIZK proof π

and only include the succinct argument π_{SA} in the ciphertext. The rest of the components of the ciphertext remain unchanged. Such a technique for shrinking NIZK proofs using succinct arguments was recently used, for example, in the work of Boneh, Segev, and Waters [11]. And finally, we modify the validity test by invoking the verifier SAVer of the succinct argument system on π_{SA} instead of the verifier of the NIZK proof system.

The following theorem states that the scheme Π_{full} is k-source $\mathsf{PRV\text{-}CDA2}$ secure (see Definition 4.5). A proof outline is presented in Section 2.

Theorem 5.1. *Let $\mathcal{SE}$ be a one-time secure symmetric-key encryption scheme, $\mathcal{TC}$ be a statistically-hiding commitment scheme, $\mathcal{ZK}$ be a non-interactive extractable zero-knowledge proof system, and $\mathcal{H}$ be a family of $\left(\mathrm{poly}, 2^{-\omega(\log^2 \lambda)}\right)$-collision-resistant hash functions. Then, under the $\omega(\log^2 \lambda)$-min-entropy DDH assumption and the CDH assumption in group $\mathbb{G}$, for any $k > \omega(\log^2 \lambda)$, Π_{full} is k-source $\mathsf{PRV\text{-}CDA2}$ secure with RO and FS modeled as random oracles.*

Next, we state that the scheme Π_{full} satisfies the notion of *strong* tag consistency as in Definition 4.7.

Theorem 5.2. *Let $\mathcal{TC}$ be a secure commitment scheme, $\mathcal{ZK}$ be a non-interactive extractable zero-knowledge proof system, and $\mathcal{H}$ be a family of $\left(\mathrm{poly}, 2^{-\omega(\log^2 \lambda)}\right)$-collision-resistant hash functions. Then, setting $n \geq \omega(\log \lambda)$, Π_{full} is strongly tag consistent.*

6 A Deterministic Scheme for Bounded Message Distributions

The Scheme. Our deterministic MLE2 scheme uses as a building block an IND-CPA secure symmetric-key scheme $\mathcal{SE} = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ with the same message space $\mathcal{M}$ as the MLE2 scheme, key space $\mathcal{K}$, ciphertext space $\mathcal{C}$, and randomness length ρ. It is additionally parameterized by an integer $q = q(\lambda)$. The scheme $\Pi^{(q)}_{\mathsf{det}} = (\mathsf{PPGen}, \mathsf{KD}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{EQ}, \mathsf{Valid})$ is defined as follows:

- **Parameter-generation algorithm:** On input 1^λ, the algorithm PPGen chooses two hash functions $H_1 \colon \{0,1\}^* \to \mathcal{K}$ and $H_2 \colon \{0,1\}^* \to \{0,1\}^\rho$. It outputs the public parameters $\mathsf{pp} = (H_1, H_2, q)$.
- **Key-derivation function:** The algorithm KD takes as input public parameters pp, a message m, and outputs the message-derived key $k_m = H_1(m\|1) \oplus H_1(m\|2) \oplus \cdots \oplus H_1(m\|q+1) \in \mathcal{K}$.
- **Encryption algorithm:** The algorithm Enc takes as input public parameters pp, a message m, and a message-derived key k_m. It computes $r_m = H_2(m\|1) \oplus H_2(m\|2) \oplus \cdots \oplus H_2(m\|q+1)$ and outputs $\mathsf{E}_{k_m}(m; r_m) \in \mathcal{C}$.
- **Validity test:** The algorithm Valid outputs 1 on any input $c \in \mathcal{C}$.
- **Decryption algorithm:** Dec takes as input public parameters pp, a ciphertext c, and a message-derived key k_m and outputs $m \leftarrow \mathsf{D}_{k_m}(c)$.

– **Equality algorithm:** Algorithm EQ on input public parameters pp and ciphertexts c_1 and c_2 outputs 1 if and only if $c_1 = c_2$.

The following theorem, which is analogous to the combination of Theorems 5.1 and 5.2, captures security of $\Pi^{(q)}_{\mathsf{Det}}$. However, security is established in a different, incomparable adversarial model: the source specified by the adversary is allowed to output T, possibly correlated, messages at a time as long as the sampling circuit makes no more than q random oracle queries.

Theorem 6.1. *Let $q \in \mathbb{N}$ be polynomial in the security parameter λ.*

1. *If $\mathcal{SE}$ is an* IND-CPA *secure scheme and H_1 and H_2 are modeled as random oracles, then, for any any $T = \mathrm{poly}(\lambda)$ and any $k = \omega(\log \lambda)$, $\Pi^{(q)}_{\mathsf{det}}$ is q-query (T, k)-source* PRV-CDA2*-secure.*
2. *The scheme $\Pi^{(q)}_{\mathsf{Det}}$ is strongly tag consistent.*

7 Conclusions and Open Problems

Prior definitions and schemes for message-locked encryption (MLE) admit only an adversary who is oblivious to the scheme's public parameters during the initial interaction. We explore two avenues for extending security guarantees of MLE towards a more powerful adversarial model, where the distribution of plaintexts can be correlated with the scheme's parameters (lock-dependent messages). In our first construction we augment the definition of MLE to allow fully random ciphertexts by supporting equality-testing functionality. One challenging aspect of the construction is ensuring ciphertext consistency in the presence of random oracles without inflating the length of the ciphertext. We achieve this goal via a combination of a cut-and-choose technique and NIZKs. The resulting scheme is secure against a fully adaptive adversary. Our second construction assumes a predetermined bound on the complexity of distributions specified by the adversary. It fits the original framework of deterministic MLE while satisfying a stronger security notion.

We formulate the following several directions for further research. First, we ask whether a fully adaptive randomized MLE2 can be constructed and proven secure in the standard model. Second, a randomized scheme for deduplication creates a potential leakage channel that allows one user to test whether her plaintext has already been uploaded to the system (similar to the attack described by Harnik et al. [20] where the deduplication event was observable via traffic analysis). Designing a scheme resistant to this attack, for example, by supporting server-side rerandomization of ciphertexts, constitutes an interesting research question. Note that deterministic MLEs are immune to this problem. Finally, our first scheme requires a pairwise application of the equality-testing algorithm to identify all duplicate ciphertexts, and uses computationally expensive NIZKs as a building block. We leave reducing the overhead of the scheme as an open problem.

Acknowledgments. We thank the anonymous CRYPTO 2013 reviewers for their helpful comments.

References

1. Bitcasa, http://www.bitcasa.com
2. GNUNet, http://www.gnu.org/software/GNUnet/
3. Adya, A., Bolosky, W.J., Castro, M., Cermak, G., Chaiken, R., Douceur, J.R., Howell, J., Lorch, J.R., Theimer, M., Wattenhofer, R.: FARSITE: Federated, available, and reliable storage for an incompletely trusted environment. In: Culler, Druschel [16], pp. 1–14
4. Bellare, M., Boldyreva, A., O'Neill, A.: Deterministic and efficiently searchable encryption. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 535–552. Springer, Heidelberg (2007)
5. Bellare, M., Brakerski, Z., Naor, M., Ristenpart, T., Segev, G., Shacham, H., Yilek, S.: Hedged public-key encryption: How to protect against bad randomness. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 232–249. Springer, Heidelberg (2009)
6. Bellare, M., Fischlin, M., O'Neill, A., Ristenpart, T.: Deterministic encryption: Definitional equivalences and constructions without random oracles. In: Wagner [31], pp. 360–378
7. Bellare, M., Keelveedhi, S., Ristenpart, T.: Message-locked encryption and secure deduplication. In: Johansson, Nguyen [21], pp. 296–312
8. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Denning, D.E., Pyle, R., Ganesan, R., Sandhu, R.S., Ashby, V. (eds.) ACM Conference on Computer and Communications Security, pp. 62–73. ACM (1993)
9. Black, J., Rogaway, P., Shrimpton, T.: Encryption-scheme security in the presence of key-dependent messages. In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 62–75. Springer, Heidelberg (2003)
10. Boldyreva, A., Fehr, S., O'Neill, A.: On notions of security for deterministic encryption, and efficient constructions without random oracles. In: Wagner [31], pp. 335–359
11. Boneh, D., Segev, G., Waters, B.: Targeted malleability: homomorphic encryption for restricted computations. In: Goldwasser, S. (ed.) ITCS, pp. 350–366. ACM (2012)
12. Brakerski, Z., Segev, G.: Better security for deterministic public-key encryption: The auxiliary-input setting. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 543–560. Springer, Heidelberg (2011)
13. Canetti, R.: Towards realizing random oracles: Hash functions that hide all partial information. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 455–469. Springer, Heidelberg (1997)
14. Cox, L.P., Murray, C.D., Noble, B.D.: Pastiche: Making backup cheap and easy. In: Culler, Druschel [16], pp. 285–298
15. Cramer, R. (ed.): TCC 2012. LNCS, vol. 7194. Springer, Heidelberg (2012)
16. Culler, D.E., Druschel, P. (eds.): 5th Symposium on Operating System Design and Implementation (OSDI 2002), Boston, Massachusetts, USA, December 9-11. USENIX Association (2002)
17. Dodis, Y., Ristenpart, T., Vadhan, S.P.: Randomness condensers for efficiently samplable, seed-dependent sources. In: Cramer [15], pp. 618–635
18. Douceur, J.R., Adya, A., Bolosky, W.J., Simon, D., Theimer, M.: Reclaiming space from duplicate files in a serverless distributed file system. In: ICDCS, pp. 617–624 (2002)

19. Fuller, B., O'Neill, A., Reyzin, L.: A unified approach to deterministic encryption: New constructions and a connection to computational entropy. In: Cramer [15], pp. 582–599
20. Harnik, D., Pinkas, B., Shulman-Peleg, A.: Side channels in cloud services: Deduplication in cloud storage. IEEE Security & Privacy 8(6), 40–47 (2010)
21. Johansson, T., Nguyen, P.Q. (eds.): EUROCRYPT 2013. LNCS, vol. 7881. Springer, Heidelberg (2013)
22. Kim, Y., Yurcik, W. (eds.): Proceedings of the 2008 ACM Workshop on Storage Security and Survivability, StorageSS 2008, Alexandria, VA, USA. ACM (October 31, 2008)
23. MacWilliams, F., Sloane, N.: The theory of error-correcting codes. North-Holland (1977)
24. Mironov, I., Pandey, O., Reingold, O., Segev, G.: Incremental deterministic public-key encryption. In: Pointcheval, Johansson [27], pp. 628–644
25. Mislove, A., Post, A., Reis, C., Willmann, P., Druschel, P., Wallach, D.S., Bonnaire, X., Sens, P., Busca, J.M., Arantes, L.B.: POST: A secure, resilient, cooperative messaging system. In: Jones, M.B. (ed.) HotOS, pp. 61–66. USENIX (2003)
26. Muthitacharoen, A., Chen, B., Mazières, D.: A low-bandwidth network file system. In: SOSP 2001, pp. 174–187 (2001)
27. Pointcheval, D., Johansson, T. (eds.): EUROCRYPT 2012. LNCS, vol. 7237. Springer, Heidelberg (2012)
28. Quinlan, S., Dorward, S.: Venti: A new approach to archival storage. In: Long, D.D.E. (ed.) FAST, pp. 89–101. USENIX (2002)
29. Raghunathan, A., Segev, G., Vadhan, S.P.: Deterministic public-key encryption for adaptively chosen plaintext distributions. In: Johansson, Nguyen [21], pp. 93–110
30. Storer, M.W., Greenan, K.M., Long, D.D.E., Miller, E.L.: Secure data deduplication. In: Kim, Yurcik [22], pp. 1–10
31. Wagner, D. (ed.): CRYPTO 2008. LNCS, vol. 5157. Springer, Heidelberg (2008)
32. Wee, H.: Dual projective hashing and its applications—lossy trapdoor functions and more. In: Pointcheval, Johansson [27], pp. 246–262
33. Wilcox-O'Hearn, Z., Warner, B.: Tahoe: the least-authority filesystem. In: Kim, Yurcik [22], pp. 21–26
34. Yang, G., Tan, C.H., Huang, Q., Wong, D.S.: Probabilistic public key encryption with equality test. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 119–131. Springer, Heidelberg (2010)
35. Zhu, B., Li, K., Patterson, R.H.: Avoiding the disk bottleneck in the data domain deduplication file system. In: Baker, M., Riedel, E. (eds.) FAST, pp. 269–282. USENIX (2008)

The Mix-and-Cut Shuffle: Small-Domain Encryption Secure against N Queries

Thomas Ristenpart[1] and Scott Yilek[2]

[1] University of Wisconsin–Madison
rist@cs.wisc.edu
[2] University of St. Thomas
syilek@stthomas.edu

Abstract. We provide a new shuffling algorithm, called Mix-and-Cut, that provides a provably-secure block cipher even for adversaries that can observe the encryption of all $N = 2^n$ domain points. Such fully secure ciphers are useful for format-preserving encryption, where small domains (e.g., $n = 30$) are common and databases may well include examples of almost all ciphertexts. Mix-and-Cut derives from a general framework for building fully secure pseudorandom permutations (PRPs) from fully secure pseudorandom separators (PRSs). The latter is a new primitive that we treat for the first time. Our framework was inspired by, and uses ideas from, a particular cipher due to Granboulin and Pornin. To achieve full security for Mix-and-Cut using this framework, we give a simple proof that a PRP secure for $(1 - \epsilon)N$ queries (recently achieved efficiently by Hoang, Morris, and Rogaway's Swap-or-Not cipher) yields a PRS secure for N queries.

Keywords: shuffles, small-block encryption, tweakable block ciphers.

1 Introduction

Traditional block ciphers such as AES and DES work on fixed domain sizes (e.g., $n = 64$ or 128 bits). Some applications, however, require the ability to securely encipher bit strings of smaller sizes (e.g., $n = 30$ bits). The canonical example here being format-preserving encryption (FPE) [2, 4, 6, 7], which makes use of small-block-size block ciphers to perform in-place encryption of credit card numbers, social security numbers, and other sensitive data.

In this paper, we provide a new small-domain block cipher, called Mix-and-Cut, that achieves provable security up to $q = 2^n$ queries (the most possible). It was designed using a new methodology for building ciphers secure as pseudorandom permutations (PRPs) from pseudorandom separators (PRSs). The latter is a cryptographic primitive that we treat for the first time. This methodology was inspired by, and uses ideas from, a particular cipher construction due to Granboulan and Pornin (GP) [11].

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 392–409, 2013.

Small domain encryption. Before explaining our results in more detail, we describe further the motivation and related work. FPE has become popular in settings where ciphertexts must follow a proscribed format. For example, should credit card numbers (CCNs) already be stored in a database with a limit of 16 numerical digits, then encrypting the CCN with a conventional encryption scheme would result in a ciphertext that could not be placed back in the database column. Typically, in fact, the first 6 digits (being the issuer identification number) and the last digit (a Luhn checksum digit) must also be left in the clear, and so one would like to encrypt just the remaining 9 digits. This requires a cipher with domain of $10^9 \approx 2^{30}$. In cryptographic parlance, we seek a block cipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ for some key space $\mathcal{K}$, $n = 30$, and which is indistinguishable from a random permutation.

Work on this small-space encryption problem can be traced back to, at least, Black and Rogaway [5]. They gave several approaches, the most efficient of which generalizes Luby and Rackoff's classic result [17] on balanced Feistel networks. This provides provable security, but only up to a number of encryptions below $q = 2^{n/4}$. In our example with $n = 30$, this is only $q \approx 128$ ciphertexts. Morris, Rogaway, and Stegers [19] uncover a connection between unbalanced Feistel networks and the Thorp shuffle, and use it to prove that maximally unbalanced Feistel networks achieve security up to about $q \approx 2^{n(1-\epsilon)}$ queries for a fraction ϵ inversely proportional to the the number of rounds of the construction. This approach, with similar bounds, was extended to arbitrary balanced Feistels by Hoang and Rogaway [14]. Most recently, Hoang, Morris, and Rogaway [13] introduced a new shuffling approach, called Swap-or-Not, and a cryptographic realization of it that provably achieves security up to $q \approx (1-\epsilon)2^n$.

None of the above approaches, however, provide guarantees of security should q get within a constant of $N = 2^n$. Such *full security* is desirable when large databases contain almost N encryptions. In our running example, this would mean only about 1 billion database entries, which is not particularly large considering that processors using FPE deal with hundreds of millions of transactions each month. Employing key rotation and tweaks [16] can ease this gap (by reducing the number of ciphertexts per key/tweak), but their use for this purpose is not always feasible or desirable, for example should one need to support search. What's more, choosing appropriate security parameters requires somehow predicting the number of ciphertexts an adversary obtains. A fully secure cipher, on the other hand, can often set parameters based solely on n.

There exist a handful of proven full security constructions, but they are all quite slow for moderately sized N. The shuffle popularized by Knuth [9, 10, 15] provides a cipher with full security, but requires $\mathcal{O}(N)$ computation time, as does the simple construction that uses lookup tables (c.f., [5]). Granboulan and Pornin (GP) [11], build a cipher based on a shuffle introduced by Czumaj, Kanarek, Kutylowski, and Lory [8]. It requires $\mathcal{O}(\log^3 N)$ operations and repeated samplings from the hypergeometric distribution. The Thorp shuffle was shown to provide full security [18], but also with $\mathcal{O}(\log^3 N)$ rounds. Stefanov and Shi [23] offer a variant of the GP scheme that improves performance for very small

algorithm $\mathsf{MixAndCut}(D)$:
$n \leftarrow \log |D|$
Repeat r times:
  $K \leftarrow\!\!\$\ \{0,1\}^n$
  for each pair of positions $\{X, K \oplus X\}$
    $b \leftarrow\!\!\$\ \{0,1\}$
    If $b = 1$ swap the cards at positions X and $K \oplus X$
$(D_1, D_2) \leftarrow \mathsf{Cut}(D)$
$\mathsf{MixAndCut}(D_1)$
$\mathsf{MixAndCut}(D_2)$
$\mathsf{Gather}(D_1, D_2)$

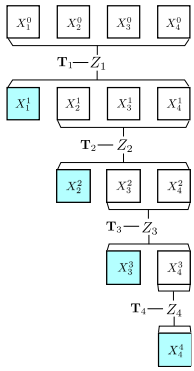


Fig. 1. (Left) The Mix-and-Cut shuffle on $N = 2^n$ cards with Swap-or-Not for the mixing. D is the size N deck of cards, $|D|$ is the number of cards in D, $\mathsf{Cut}(D)$ cuts the deck exactly in half, and $\mathsf{Gather}(D_1, D_2)$ stacks two piles on top of each other. **(Right)** Diagram of the Icicle construction with four stages ($s = 4$) and using pseudorandom separators $Z_1, \ldots, Z_4$. Each of the X_j^i have length γ bits. The leftmost γ bits (the shaded boxes) of each stage "drip" down to the final ciphertext $X_1^1 \,\|\, X_2^2 \,\|\, X_3^3 \,\|\, X_4^4$. Each $\mathbf{T}_i$ is a tweak that includes the X_j^j values output in previous stages $j < i$.

domains, but it requires $\tilde{\Theta}(N)$ time for key setup and $\tilde{\Theta}(N^{1/2})$ time and space for encryption.

Mix-and-Cut. We offer a new cipher, which, following [13,14,19] can be viewed as a card shuffle. Think of each of the 2^n domain points as a card. Our new shuffling algorithm intermingles two kinds of shuffles on these cards. The first is the recursive shuffling procedure underlying GP, in which at each stage one splits the deck of cards into two halves, then splits each half into two smaller halves for a total of four halves, and so on until one can split no further. GP uses a recursive hypergeometric sampling routine to make the split perfectly random. We instead use another shuffling routine to provide a split that is cryptographically indistinguishable from a random one. For this we use a number of rounds of the Swap-or-Not shuffle sufficient to ensure the top and bottom halves of the deck appear to have been chosen randomly (but not yet nearly enough rounds to prove a full uniform shuffling of the deck). The complete shuffling algorithm is given on the left of Figure 1. The full shuffle, then, will *mix* the deck using Swap-or-Not, then *cut* it into two, recursively shuffle each one independently, and so on.

Mix-and-Cut may appear ad-hoc, fitting together two prior shuffles in an arbitrary fashion. But in fact it was discovered using, and is just the best instantiation we could find of, a new paradigm for building ciphers that is inspired by the construction of the GP cipher. We describe this paradigm from the bottom up, starting with a new cryptographic primitive that we define, called a Γ-pseudorandom separator.

Γ-PSEUDORANDOM SEPARATORS. A Γ-PRS, from a shuffling point of view, can "split a deck" pseudorandomly into Γ piles. Cryptographically speaking, a permutation Z on $\{0,1\}^n$ is a secure Γ-PRS if no computationally bounded adversary can distinguish the first $\gamma = \log \Gamma$ bits of its output from that of a randomly chosen, regular function $\{0,1\}^n \to \{0,1\}^\gamma$. That the ideal object is regular is important: we will be often interested in n small and q large, and here the difference between a regular random function and arbitrary random function is readily apparent even to a computationally bounded adversary. For large n and small q, however, Γ-PRS security can be shown to be equivalent to security in the sense of a PRF with range size γ bits using techniques first used to analyze the PRF security of truncated PRPs [1, 12].

Unlike a PRP, a Γ-PRS provides no security guarantees about the remaining $n-\gamma$ bits of its output (when $\gamma < n$). This means that Γ-PRS security is strictly weaker than PRP security in general; when $\gamma = n$ the two notions coincide.

In fact, we focus on tweakable separators, which like tweakable block ciphers in the sense of Liskov, Rivest and Wagner [16], are families of permutations indexed by both a key and a tweak. Different tweaks should give rise to independent-looking permutations. Tweaks will be critical to our use of PRSs in building PRPs.

Separators were inspired by the use, in the GP cipher, of an algorithm for perfectly splitting a deck into two halves using a recursive sampling from the hypergeometric distribution. Our treatment here draws out the implicit cryptographic primitive underlying GP's algorithm: in our terminology, the GP hypergeometric separator is a fully secure 2-PRS.

FROM PRSS TO PRPS. Fix a value γ and target block size n, with $s = n/\gamma$. We provide a new construction, that we call an icicle, that uses a set of tweakable permutations $Z_1, \ldots, Z_s$ to build a secure PRP on n bits. See the right hand side of Figure 1. Each Z_i should be a secure $\Gamma = 2^\gamma$-PRS on $n - \gamma(i-1)$ bits. An icicle is simple: apply an n-bit Γ-PRS, output the first γ bits, apply to the remaining $n-\gamma$ bits a Γ-PRS on $n-\gamma$ bits with tweak being the output thus far, add the first γ bits of the result to the output, and so on for s stages, in order to produce a sequence of γ-bit outputs that is the ciphertext. The use of tweaks is requisite for security: they ensure that the PRSs lower in the icicle provide independent behavior for different prefixes of the ciphertext. An icicle for $\Gamma = 2$ is exactly the recursive shuffling procedure used in the GP cipher; their cipher we can view now as an icicle using hypergeometric 2-PRSs. Our proof of the icicle construction modularizes their result, and generalizes it to work with imperfect, computational PRSs for arbitrary Γ. The proof conserves full security, as well, so if the underlying PRSs are fully secure, so too is the resulting PRP.

Overall, this can be seen as a new paradigm for building PRPs. Unlike Luby-Rackoff that starts with PRFs, we go a different route, starting with PRSs. This may seem to not buy much; in particular, building an n-bit cipher using an icicle requires an n-bit permutation (the first stage) and also a γ-bit PRP (the last stage). But we only require the first γ bits of the first stage to be random-looking,

and we can arrange that γ is small enough in the last stage to make a PRP there trivial (e.g., $\gamma = 1$).

A simple but useful lemma. This begs the question of how to build PRSs, particularly ones that are faster than the GP hypergeometric 2-PRS. We show that the inverse E_K^{-1} of any cipher E_K that is a secure PRP for $(\Gamma - 1)N/\Gamma$ queries is a good Γ-PRS for all N queries. The proof is straightforward (see Section 4), and we explain it informally here for the case of $\Gamma = 2$ which we will use later. The reduction must simulate all N 2-PRS queries given only $N/2$ evaluations of E. But to do so requires only returning the first bit of each value $E_K^{-1}(X_1), \ldots, E_K^{-1}(X_N)$, and this is learnable by querying only half the domain, say by querying $E_K(0 \,\|\, y)$ for all $y \in \{0,1\}^{n-1}$. If a value X is in the set of returned points, then we know that the first bit of $E_K^{-1}(X)$ is zero and otherwise that it is one. The result extends easily to handle when the PRS adversary queries all N domain points for each of some number of tweaks.

The lemma shows that one can get a fully secure Γ-PRS using any construction that achieves only $(1 - \epsilon)N$ PRP security. In particular this implies that Swap-or-Not is a fully secure 2-PRS for a number of rounds a small fraction of that needed to make it a fully secure PRP (provably under the Hoang et al. result).

Putting it all together. The Mix-and-Cut cipher uses the icicle construction with fully secure 2-PRSs built from Swap-or-Not. Our simple lemma described above ensures that we can use the number of Swap-or-Not rounds suggested by Hoang et al. for $N/2$ queries to establish 2-PRS security for N queries. Back to the shuffling interpretation, Swap-or-Not need only shuffle enough to ensure that the "deck" can be cut into two piles with a pseudorandom assignment of cards to piles. We then cut the deck, and focus on each pile independently.

The resulting cipher provides full security, using only simple operations: Swap-or-Not can be instantiated with two AES calls per round. That said, the new cipher does require a large number of rounds. For $N = 2^{30}$ and an advantage of less than 10^{-10} we need around 10,000 rounds. By comparison, using Swap-or-Not directly under the bounds[1] of Hoang et al. requires about 126×10^9 rounds to achieve the same advantage for $N = 2^{30}$. On an Intel Core i5 with AES-NI, a full application of Mix-and-Cut should take less than a millisecond for $N = 2^{30}$. Improved analyses for Swap-or-Not or another algorithm (directly as a 2-PRS or as a PRP) can be used immediately by Mix-and-Cut in order to increase efficiency.

Adding tweaks and CCA security. It is easy to ensure that icicle produces a tweakable block cipher: just prepend the tweak T to the tweak used in each stage. We have also described all the above in terms of achieving CPA security. But CPA security and CCA security are equivalent when $q = N$ — another advantage of targeting full security.

[1] Their bound is vacuous if one sets $q = N$, but we can apply it with $q = N - 1$.

Separators for generalized domains. Largely for pedagogical reasons, we focus in this abstract on the case where domains have sizes that are powers of 2. However, many small-domain encryption settings require domains that operate over non-binary digits (e.g., base 10 for credit cards). We can generalize our results to work with radixes other than two in a straightforward way. This natural generalization leads to slightly weaker bounds than the binary case. It is also possible to treat the most general case, which uses PRSs to build tweakable ciphers for completely arbitrary domains. We give details in the full version.

Other uses of Γ-PRSs. We believe that the new Γ-PRS primitive will find application in contexts beyond our goal here of full security ciphers. As one example, we show in the full version that 2 rounds of balanced Feistel gives a good $2^{n/2}$-PRS, though with security only for $q \approx 2^{n/4}$. While therefore not directly useful for full security applications, we show that one can recast the original Luby-Rackoff (LR) result [17] that 3-rounds provides a secure PRP as a composition of any $2^{n/2}$-PRS with one round of Feistel. Combining the two results gives a bound that matches the original LR result.

Further discussion. Should one be interested in just partial security ($q \ll N$), then our approach does not provide the most efficient solution. This limitation extends as well to very large block sizes, where partial security is inherently the goal (since, e.g., $q = 2^{128}$ is unrealistic); our lemma described above scales exponentially in q. In these settings one would do best to stick with (say) Swap-or-Not up to $n = 64$ and from there use traditional block ciphers within a suitable domain extension transform (e.g., [22]).

The icicle construction, being built from any Γ-PRS, can of course be used in a multitude of ways. A two stage icicle extends the domain of any fixed-length PRP by γ bits. One can also build a full cipher using multiple different kinds of separators across different icicle stages. For example, one could use several stages of (say) Swap-or-Not before applying a different shuffle for a few stages, before then applying at the bottom of the icicle a permutation that works well for very small domains. While the resulting cipher would be more complicated than Mix-and-Cut with its homogeneous set of separators, it suggests the existence of a wide space of possible designs from which one might use proven-secure ciphers for the domain sizes for which they work best to improve overall efficiency.

Finally, we note that in practice small-block encryption uses constructions that have weak bounds or even have no security proofs entirely. The proposed FFX [3,4] standard for FPE suggests 10 rounds of Feistel for domain size around 2^{30}. The choice of rounds is based on a heuristic; no proofs providing reasonable bounds are known for this choice and in fact no proofs are likely given current techniques (see [19] for more discussion). That said, no (computationally reasonable) attacks are known, and we have no reason to believe that efficient attacks will arise. (The best is due to Patarin [21], but the round choice was made to defeat this.) But that is not proof of the absence of attacks, and so we view closing the (large) performance gap between Mix-and-Cut and ciphers such as FFX an important open research problem.

2 Preliminaries

TWEAKABLE BLOCK CIPHERS. A tweakable block cipher is a family of functions $E : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{K}$ is a non-empty, finite set called the key space, $\mathcal{T}$ is a non-empty, finite set called the tweak space, and where for every $K \in \mathcal{K}$ and $T \in \mathcal{T}$, $E_K(T,\cdot) = E(K,T,\cdot)$ gives a permutation on $\{0,1\}^n$. We let $E_K^{-1}(T,\cdot)$ denote the inverse block cipher of E. When $\mathcal{T}$ is a singleton, we have a block cipher, and write instead $E_K(\cdot) = E(K,\cdot)$ and $E_K^{-1}(\cdot) = E^{-1}(K,\cdot)$.

We target tweakable block ciphers that are secure even under chosen-ciphertext attack. This is sometimes called strong pseudorandom permutation (SPRP) security. Given tweakable block cipher $E : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ and an adversary A, the cca-advantage of A with respect to E is

$$\mathbf{Adv}^{\mathrm{cca}}_E(A) = \Pr\left[A^{E(K,\cdot,\cdot),E^{-1}(K,\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[A^{\pi(\cdot,\cdot),\pi^{-1}(\cdot,\cdot)} \Rightarrow 1\right]$$

where the first probability is over $K \xleftarrow{\$} \mathcal{K}$ and the coins used by A and the second probability is over $\pi \xleftarrow{\$} \mathsf{Perm}(\mathcal{T},n)$ and the coins used by A. Here $\mathsf{Perm}(\mathcal{T},n)$ is the set of families of n-bit permutations. That means picking π gives a family of uniformly chosen permutations, one for each $T \in \mathcal{T}$. The inverse of π is denoted π^{-1}. When $\mathcal{T} = \{0,1\}^t$ we will write $\mathsf{Perm}(t,n)$. A cpa adversary simply makes no queries to its second oracle, and for such adversaries we denote their advantage by $\mathbf{Adv}^{\mathrm{cpa}}_E(A)$ as a reminder that A makes no inverse queries. We call a cipher that is secure under cpa attack a good PRP.

We will need as well non-adaptive cpa security, which we define as follows. A non-adaptive cpa adversary A is given access to one of two different oracles to which it can query a single time a pair of vectors $(T_1,\ldots,T_q)$ and $(M_1,\ldots,M_q)$. The oracle $\boldsymbol{E}(K,(T_1,\ldots,T_q),(M_1,\ldots,M_q))$ computes $C_i = E_K^{T_i}(M_i)$ for all i and returns the resulting ciphertexts. The oracle $\boldsymbol{\pi}((T_1,\ldots,T_q),(M_1,\ldots,M_q))$ computes $C_i = \pi(T_i,M_i)$ for a random tweakable permutation π, and returns the results. We define advantage as

$$\mathbf{Adv}^{\mathrm{ncpa}}_E(A) = \Pr\left[A^{\boldsymbol{E}(K,\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[A^{\boldsymbol{\pi}(\cdot,\cdot)} \Rightarrow 1\right]$$

where the first probability is over the choice of $K \xleftarrow{\$} \mathcal{K}$ and the coins used by A and the second probability is over $\pi \xleftarrow{\$} \mathsf{Perm}(\mathcal{T},n)$ and the coins used by A.

FULL SECURITY. We target full security, meaning that cca advantage should be low even for adversaries that make $q = N = 2^n$ queries for some number w of tweaks. (Clearly $q = N - 1$ is also sufficient, but the difference matters little.) Thus, full security requires security to hold for a total of wN queries. When security holds only for $q \ll N$ we say that the tweakable cipher instead achieves only *partial security*. Partial security suffices when, as with standard block ciphers with $n = 128$, the domain is so large that no adversary could feasibly obtain, let alone compute over, anywhere remotely close to N queries. In small domain encryption settings, however, full security is important as applications may apply a cipher to most of the domain (for some set of tweaks). Another advantage of targeting full security is that cpa security and cca security are

equivalent when $q = N$. The following formalizes this fact and allows us to focus on cpa adversaries in the remainder of the paper.

Lemma 1. *Let* $E\colon \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ *and* $N = 2^n$. *Let* A *be a cca adversary making queries for* w *distinct tweaks. Then for the cpa adversary* B *specified in the proof below it holds that* $\mathbf{Adv}^{\mathrm{cca}}_E(A) \le \mathbf{Adv}^{\mathrm{cpa}}_E(B)$. *Moreover* B *makes at most* wN *queries and runs in time that of* A *plus* $\mathcal{O}(wN \log wN)$ *time.*

Proof. The adversary B runs adversary A. When A makes either a forward or inverse query on a not-before-seen tweak T, B immediately queries values $(T, X_1), \ldots, (T, X_N)$. That is, it queries the entire domain. Then B uses the resulting values $Y_1, \ldots, Y_N$ to respond to the query, and to respond to future forward or inverse oracle queries using T. ∎

Note that the variable w in the above only measures the number of distinct tweaks queried, not the total number of queries made by A. Thus, even if A makes $2^n - 1$ queries on each tweak, the bound holds as shown. Also, when we use big-O notation, i.e., $\mathcal{O}(w2^n \log w2^n)$ in the lemma above, this hides only small, fixed constants.

PRFs. Let $F\colon \mathcal{K} \times \{0,1\}^\ell \to \{0,1\}^n$ be a family of functions. For an adversary A, the prf security of F with respect to A is $\mathbf{Adv}^{\mathrm{prf}}_F(A) = \Pr\left[A^{F(K,\cdot)} \Rightarrow 1\right] - \Pr\left[A^{\rho(\cdot)} \Rightarrow 1\right]$ where the first probability is over $K \xleftarrow{\$} \mathcal{K}$ and the coins used by A and the second probability is over $\rho \xleftarrow{\$} \mathsf{Func}(\ell, n)$ and the coins used by A. Here $\mathsf{Func}(\ell, n)$ is the set of all functions $\{0,1\}^\ell \to \{0,1\}^n$. It will be convenient to speak of PRFs that accept tweaks as input in addition to messages and keys, so a map $F\colon \mathcal{K} \times \mathcal{T} \times \{0,1\}^\ell \to \{0,1\}^n$. It is easy to build such an F that is a good PRF using an untweaked PRF. We write $\mathsf{Func}(\mathcal{T}, \ell, n)$ to denote the set of all such functions and $\mathsf{Func}(t, \ell, n)$ should $\mathcal{T} = \{0,1\}^t$.

3 Pseudorandom Separators

We define a new security goal for tweakable permutations. Informally speaking, a family of permutations is a good Γ-*pseudorandom separator* or simply a good Γ-PRS if no adversary can distinguish the first $\gamma = \log \Gamma$ bits of its outputs from a random, regular function $\{0,1\}^n \to \{0,1\}^\gamma$. (Regular just means that each range point has $2^{n-\gamma}$ preimages.) The shuffling-based interpretation is that the permutation does a good job of separating the domain into Γ different piles with random domain points (cards) assigned to each pile.

More formally, a *tweakable separator* is a map $Z\colon \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ where $\mathcal{K}$ is the key space, $\mathcal{T}$ is the tweak space, and we require that for all $K \in \mathcal{K}$ and $T \in \mathcal{T}$, $Z(K, T, \cdot)$ is a permutation with inverse $Z^{-1}(K, T, \cdot)$. Let $\mathsf{RegFunc}(\mathcal{T}, n, \gamma)$ be the set of all functions $f\colon \mathcal{T} \times \{0,1\}^n \to \{0,1\}^\gamma$ where $f(T, \cdot)$ is regular for any $T \in \mathcal{T}$. Then for $\gamma \le n$, the Γ-prs advantage of an adversary A is defined by

$$\mathbf{Adv}^{\Gamma\text{-prs}}_Z(A) = \Pr\left[A^{Z(K,\cdot,\cdot)[\gamma]} \Rightarrow 1\right] - \Pr\left[A^{\hat{\rho}(\cdot,\cdot)} \Rightarrow 1\right]$$

where the first probability is over $K \xleftarrow{\$} \mathcal{K}$ and the coins used by A and the second probability is over $\hat{\rho} \xleftarrow{\$} \mathsf{RegFunc}(\mathcal{T}, n, \gamma)$ and the coins used by A. The oracle $Z(K, \cdot, \cdot)[\gamma]$ on input (T, x), returns the first γ bits of $Z(K, T, x)$.

We say, informally, that a tweakable separator Z is Γ-PRS secure if the advantage is low for all adversaries given reasonable resources. Looking ahead, we will use separators in building fully secure, small-block ciphers, and so reasonable here means $q = N$ queries can be made for each of some number ω of tweaks.

It is easy to see that any separator that is Γ-PRS secure is also Γ'-PRS secure for any $\gamma' < \gamma$. The converse is also true, for any $\gamma' < \gamma$ one can construct a Γ'-PRS that is not a Γ-PRS. (Briefly, use the hypergeometric sampler described below within the icicle construction of Section 5 for just γ' stages, and output the result without further processing of the low $n - \gamma'$ bits.)

Separators for generalized domains. In the above, and in the next few sections, we restrict attention to the case of bit-string domains. We find this to be pedagogically appealing. However, many small-domain encryption settings require domains that operate over non-binary digits (e.g., base 10 for credit cards). Generalizing the above formulation and our results in Sections 4, 5, and 6 to work over domains being strings over an arbitrary alphabet Σ is straightforward. We do caution that the natural generalization leads to slightly weaker bounds than the binary case; see the full version for the details. There we also treat the most general case, using PRSs to build tweakable ciphers for domains of any size.

PRS versus PRF. It may seem that we have, above, just tediously repeated with different language the classical PRF security notion for the special case of a truncated permutation. However, there is indeed a gap between the two notions, due to the fact that we require the random function in the PRS setting to be regular. In fact, PRF security is not helpful to us in our full security setting. Formally:

Proposition 1. *Let $\rho \xleftarrow{\$} \mathsf{Func}(\mathcal{T}, n, \gamma)$ for $n \geq \gamma$ and finite set $\mathcal{T}$. There exists an adversary A making $N = 2^n$ queries such that $\mathbf{Adv}_{\rho}^{\Gamma\text{-prs}}(A) > 1 - e/\left((\sqrt{2\pi})^{\Gamma} \cdot N^{(\Gamma-1)/2} \cdot \Gamma^{(N-\Gamma)/2}\right)$ where $\Gamma = 2^{\gamma}$.*

The attack is simple: just query all points, and determine if the function is regular by inspecting preimage sets. The probability that a random function is regular is at most the fraction shown, derived straightforwardly using Stirling's approximation. Even for $n = 2$ and $\gamma = 1$, we have the advantage of A being about 0.9.

We note, however, that for $q \ll N$ and for certain ranges of values of γ and n, there exist upper bounds showing that PRF security implies PRS security. For example, a simple birthday bound argument shows this for reasonably large γ (and so n) and relatively small q. Better bounds for some parameters by Hall, Wagner, Kelsey, and Schneier [12] as well as Bellare and Impagliazzo [1] arise in their analyses of truncated PRPs. Their results also yield analogues to Proposition 1 with improved bounds when A is allowed fewer than N queries.

PRS VERSUS PRP. When $\gamma = n$, PRS security is equivalent to PRP security. For $\gamma < n$, however, PRS is strictly weaker. The reason is that the high bits of the output of a separator are not given to the adversary, and so these may be entirely non-random. We next give an example of a secure separator that is easily distinguished from a PRP.

THE HYPERGEOMETRIC 2-PRS. As an example of a PRS, we turn to the Granboulan and Pornin [11] splitting function that we denote $Z_{\mathrm{GP}}(x)$. Their algorithm cleverly uses repeated sampling from the hypergeometric distribution. We provide a fuller description in the full version.

GP show that Z_{GP} is (in our terms) a secure 2-PRS. It is not a secure PRP. More specifically, Z_{GP} has the property that if $x < x'$ and x and x' are mapped to the same half of the output space, then $Z_{\mathrm{GP}}(x) < Z_{\mathrm{GP}}(x')$. In other words, points that are mapped to the same half of the range retain their relative ordering. Because of this, for example, $Z_{\mathrm{GP}}(0^n)$ will always be equal to either 0^n or 10^{n-1}, and so Z_{GP} is easily distinguished from a random permutation.

The GP 2-PRS is, unfortunately, slow when implemented due to the expensive floating point computations needed to perform the repeated hypergeometric samplings. We will therefore seek other ways of constructing pseudorandom separators.

4 Full PRS Security from Partial CPA Security

In this section we prove a relationship between the cpa advantage and the Γ-prs advantage for a block cipher E. In short, we show that if a block cipher E is CPA secure against $(\Gamma-1)N/\Gamma$ queries, then its inverse E^{-1} is a secure Γ-PRS against N queries. Later in the paper, we will be particularly interested in the $\Gamma = 2$ case, since there a block cipher need only be CPA secure against $N/2$ queries. The following theorem captures this result.

Theorem 1. *Fix $n \geq \gamma \geq 1$. Let $N = 2^n$ and $\Gamma = 2^\gamma$. Let $E : \{0,1\}^k \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable block cipher and $E^{-1} : \{0,1\}^k \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ its inverse. Let A be an Γ-prs adversary making queries with ω distinct tweaks. Then for the cpa adversary B specified in the proof below it holds that $\mathbf{Adv}^{\Gamma\text{-prs}}_{E^{-1}}(A) \leq \mathbf{Adv}^{\mathrm{cpa}}_{E}(B)$. Moreover, B makes exactly $\omega \cdot N \cdot (\Gamma-1)/\Gamma$ cpa queries and runs in time $\mathcal{O}(\omega N(\Gamma-1)/\Gamma \cdot \log(\omega N(\Gamma-1)/\Gamma))$.*

Proof. Adversary B runs A and answers its oracle queries as follows. On query (T, x) from B, if T is a tweak that has never been previously queried, then B queries its own oracle on (T, y) for all $y \in \{0,1\}^n$ *except* those that begin with 0^γ. If there is a y such that the oracle query (T, y) returned x to A, then B replies to A with the first γ bits of y. Otherwise, A replies with 0^γ. When A finally outputs a final bit, B outputs this same value. In short, B is able to perfectly simulate the environment for A because it makes sure to make enough queries to its oracle to determine the first γ bits of all of A's queries. ∎

The above theorem and proof are perhaps most easily understood for the $\Gamma = 2$ case. There, to answer which half of the set $\{0,1\}^n$ the point x is mapped to, B queries the second half of $\{0,1\}^n$, i.e., all points starting with a 1. If one of the returned answers is x, then B knows to return 1. If none of the returned answers is x, then x must be mapped to the other half of $\{0,1\}^n$, so A knows it can return a 0.

5 Building PRPs from PRSs: The Icicle Construction

We now show how to use PRSs to build PRPs. The route is a new construction that we call Icicle, which recasts the Permutator algorithm of Granboulan and Pornin [11] to work for arbitrary Γ-PRSs. (In the full version we provide a generalization of Permutator that works for any domain size.) Icicle can be viewed as a way to shuffle a deck of $N = 2^n$ cards. First, use a Γ-PRS to separate the full deck of cards into Γ piles pseudorandomly. Then, recursively shuffle each of the Γ piles by separating each of them pseudorandomly into Γ smaller piles, and so on. Finally collect up all the final piles (in some fixed order) back to form a single deck. It is important that in each recursive step the shuffling is independent across all the different piles.

The cryptographic interpretation can be formalized as follows. Let $\gamma, n \in \mathbb{N}$ be numbers with $1 \le \gamma \le n$ and $\gamma \mid n$. Let $\Gamma = 2^\gamma$. Let $Z_i \colon \{0,1\}^{k_i} \times \mathcal{T}_i \times \{0,1\}^{n_i} \to \{0,1\}^{n_i}$ for $i \in [1\,..\,n/\gamma]$ and where $\mathcal{T}_i = \mathcal{T} \times \{0,1\}^{\omega_i}$ for $\omega_i = (i-1)\gamma$ and $n_i = n - (i-1)\gamma$. Note that when $i = 1$, $\mathcal{T}_i = \mathcal{T}$ and $n_i = n$. Let $\mathcal{Z} = \{Z_1, \ldots, Z_s\}$. The Icicle construction $\mathrm{Ic}_{\gamma,n}(\mathcal{Z})$ builds a tweakable block cipher $E \colon \{0,1\}^{ks} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ as defined by the pseudocode in Figure 2. We use some notation there that must be defined: $X^1 \,\|\, \cdots \,\|\, X^0$ is defined to be the empty string, while $\mathrm{par}_k(\cdot)$ takes an input string of ik bits and parses it into i strings of length k bits (and similarly for par_γ).

The Icicle uses $s = n/\gamma$ stages. In the first stage, it applies to the full n bit input a Γ-PRS on n bits. This fixes the least γ bits of the final ciphertext, which "drip" down to the output. These bits specify to which of the Γ piles the input

algorithm $E(K,T,M)$:	algorithm $E^{-1}(K,T,Y)$:
$s \leftarrow n/\gamma$	$s \leftarrow n/\gamma$
$K^1, \ldots, K^s \leftarrow \mathrm{par}_k(K)$	$K^1, \ldots, K^s \leftarrow \mathrm{par}_k(K)$
$X_1^0, \ldots, X_s^0 \leftarrow \mathrm{par}_\gamma(M)$	$X_1^1, X_2^2, \ldots, X_s^s \leftarrow \mathrm{par}_\gamma(Y)$
for $i = 1$ to s:	for $i = s$ down to 1:
$\mathbf{T}^i \leftarrow T \,\|\, X_1^1 \,\|\, \cdots \,\|\, X_{i-1}^{i-1}$	$\mathbf{T}^i \leftarrow T \,\|\, X_1^1 \,\|\, \cdots \,\|\, X_{i-1}^{i-1}$
$\mathbf{X}^{i-1} \leftarrow X_i^{i-1} \,\|\, \cdots \,\|\, X_s^{i-1}$	$\mathbf{X}^i \leftarrow X_i^i \,\|\, \cdots \,\|\, X_s^i$
$X_i^i, \ldots, X_s^i \leftarrow \mathrm{par}_\gamma(Z_i(K_i, \mathbf{T}^i, \mathbf{X}^{i-1}))$	$X_i^{i-1}, \ldots, X_s^{i-1} \leftarrow \mathrm{par}_\gamma(Z_i^{-1}(K_i, \mathbf{T}^i, \mathbf{X}^i))$
$Y \leftarrow X_1^1 \,\|\, X_2^2 \,\|\, \cdots \,\|\, X_s^s$	$M \leftarrow X_1^0 \,\|\, \cdots \,\|\, X_s^0$
return Y	return M

Fig. 2. The s-stage Icicle construction $\mathrm{Ic}_{\gamma,n}(\mathcal{Z})$ using 2^γ-PRSs $\mathcal{Z} = \{Z_1, \ldots, Z_s\}$

algorithm $\mathrm{Hyb}_j(K, T, X)$:
$s \leftarrow n/\gamma$; $K^1, \ldots, K^s \leftarrow \mathrm{par}_k(K)$; $X_1^0, \ldots, X_s^0 \leftarrow \mathrm{par}_\gamma(X)$
for $i = 1$ to j:
  $T^i \leftarrow T \,\|\, X_1^1 \,\|\, \cdots \,\|\, X_{i-1}^{i-1}$; $\mathbf{X}^{i-1} \leftarrow X_i^{i-1} \,\|\, \cdots \,\|\, X_s^{i-1}$
  $X_i^i, \ldots, X_s^i \leftarrow \mathrm{par}_\gamma(Z_i(K^i, T^i, \mathbf{X}^{i-1}))$
if $j = s$ then
  $Y \leftarrow X_1^1 \,\|\, \cdots \,\|\, X_j^j$
else
  $T^{j+1} \leftarrow T \,\|\, X_1^1 \,\|\, \cdots \,\|\, X_j^j$; $\mathbf{X}^j \leftarrow X_{j+1}^j \,\|\, \cdots \,\|\, X_s^j$
  $X_{j+1}^{j+1}, \ldots, X_s^{j+1} \leftarrow \mathrm{par}_\gamma(\pi_{j+1}(T^{j+1}, \mathbf{X}^j))$
  $Y \leftarrow X_1^1 \,\|\, \cdots \,\|\, X_j^j \,\|\, X_{j+1}^{j+1} \,\|\, \cdots \,\|\, X_s^{j+1}$
return Y

Fig. 3. Hybrid oracles for $j \in [0\,..\,s]$ where $s = n/\gamma$ as used in the proof of Theorem 2. Tweakable permutation π_j is selected randomly from $\mathsf{Perm}(\mathcal{T}_{\gamma j}, n - \gamma j)$.

was mapped by the stage. The second stage processes the remaining $n - \gamma$ bits using a Γ-PRS on n bits. The tweak includes the γ bits output from the first stage. The resulting output fixes the second γ bits of the final output, and so on. In the last stage, only γ bits remain, and these are handled by the last Γ-PRS. A pictorial diagram is shown on the right-hand-side of Figure 1 in the introduction.

Security. We now turn to analyzing the security of the Icicle construction. The following theorem shows its security assuming the underlying separators enjoy Γ-PRS security.

Theorem 2. *Fix $n \geq \gamma \geq 1$ with $\gamma \,|\, n$, let $s = n/\gamma$ and let $\mathcal{Z} = \{Z_i\}_{i=1}^s$ be an appropriate collection of permutations for $\mathrm{Ic}_{\gamma,n}(\mathcal{Z})$. Let A be a cpa adversary making q queries. Then for the Γ-prs adversaries $B_1, \ldots, B_s$ specified in the proof below it holds that $\mathbf{Adv}^{\mathrm{cpa}}_{\mathrm{Ic}_{\gamma,n}(\mathcal{Z})}(A) \leq \sum_{j=1}^s \mathbf{Adv}^{\Gamma\text{-prs}}_{Z_i}(B_s)$. Each adversary B_j makes q oracle queries, runs in time at most that of A plus $\mathcal{O}(jq \cdot \mathsf{Time}(\mathcal{Z}) + (s-j)q \log(s-j)q)$.*

Proof. We introduce a sequence of hybrid oracles $\mathrm{Hyb}_0, \ldots, \mathrm{Hyb}_s$ as defined in Figure 3. The j^{th} hybrid Hyb_j implements $\mathrm{Ic}_{\gamma,n}(\mathcal{Z})$ but with the last $s - j$ stages replaced by an appropriately sized, tweakable random permutation. By construction Hyb_0 implements a random permutation while Hyb_s implements $\mathrm{Ic}_{\gamma,n}(\mathcal{Z})$. Then

$$\mathbf{Adv}^{\mathrm{cpa}}_{\mathrm{Ic}_{\gamma,n}(\mathcal{Z})}(A) \leq \sum_{j=0}^{s-1} \left| \Pr\left[A^{\mathrm{Hyb}_{j+1}} \Rightarrow 1 \right] - \Pr\left[A^{\mathrm{Hyb}_j} \Rightarrow 1 \right] \right| \qquad (1)$$

We will now upper bound each individual difference in the sum, by way of adversaries $B_1, \ldots, B_s$ that attack the Γ-PRS security of the s underlying permutations in $\mathcal{Z}$. The adversaries are defined in Figure 4. In it, the adversary runs

adversary B_j^{Sep}:
$\mathtt{V}$ everywhere zero ; $\mathtt{Z}$ everywhere $\perp$
$K^1, \ldots, K^s \leftarrow\!\!\$\ \{0,1\}^{ks}$
Run A^{Enc}, answering queries (T, M) by:
  $X_1^0, \ldots, X_s^0 \leftarrow \mathrm{par}_\gamma(M)$
  for $i = 1$ to j:
    $T^i \leftarrow T \,\|\, X_1^1 \,\|\, \cdots \,\|\, X_{i-1}^{i-1}$; $\mathbf{X}^{i-1} \leftarrow X_i^{i-1} \,\|\, \cdots \,\|\, X_s^{i-1}$
    if $i < j$ then $X_i^i, \ldots, X_s^i \leftarrow \mathrm{par}_\gamma(Z_i(K_i, T^i, \mathbf{X}^{i-1}))$
    else $X_j^j, \ldots, X_s^j \leftarrow \mathrm{par}_\gamma(\mathrm{SepSim}(T^j, \mathbf{X}^{j-1}))$
  if $j = s$ then
    $Y \leftarrow X_1^1 \,\|\, \cdots \,\|\, X_j^j$
  else
    $T^{j+1} \leftarrow T \,\|\, X_1^1 \,\|\, \cdots \,\|\, X_j^j$; $\mathbf{X}^j \leftarrow X_{j+1}^j \,\|\, \cdots \,\|\, X_s^j$
    $X_{j+1}^{j+1}, \ldots, X_s^{j+1} \leftarrow \mathrm{par}_\gamma(\pi_{j+1}(T^{j+1}, \mathbf{X}^j))$
    $Y \leftarrow X_1^1 \,\|\, \cdots \,\|\, X_j^j \,\|\, X_{j+1}^{j+1} \,\|\, \cdots \,\|\, X_s^{j+1}$
  return Y
A outputs b'
return b'

subroutine $\mathrm{SepSim}(T', \mathbf{X})$
if $\mathtt{Z}[T', \mathbf{X}] = \perp$ then
  $X_j^j \leftarrow \mathrm{Sep}(T', \mathbf{X})$
  $\mathtt{Z}[T', \mathbf{X}] \leftarrow X_j^j \,\|\, \langle \mathtt{V}[T', X_j^j] \rangle_{n-j\gamma}$
  $\mathtt{V}[T', X_j^j] \leftarrow \mathtt{V}[T', X_j^j] + 1$
return $\mathtt{Z}[T', \mathbf{X}]$

Fig. 4. Adversaries for the proof of Theorem 2

for the first $j-1$ stages as in $\mathrm{Ic}_{\gamma,n}(\mathcal{Z})$ and then queries its own Γ-PRS oracle for the j^{th} stage. Note that the oracle only gives back γ bits.

The adversary therefore simulates the behavior of Z_j using the procedure SepSim, defined as follows. On input T', X The first γ bits X_j are set by querying $X_j^j \leftarrow \mathrm{Sep}(T', \mathbf{X}^{j-1})$. The remaining $n-\gamma$ bits are set to an arbitrary value so that the $\mathrm{SepSim}(T', \cdot)$ implements a permutation for each T'. Specifically, we set the last $n-\gamma$ bits to be equal to the value $\mathtt{V}[T, X_j^j]$, which is then incremented. (Note that $\mathtt{V}$ is set everywhere to zero initially.) This process ensures that we build a simulation of $Z_j(T', \cdot)$ that is a permutation yet agrees on the first γ bits with the output of SepSim.

The output of the first j stages, the γ-bit value returned by the PRS oracle, and the output of π_{j+1} (should $j < s$) combines to give the oracle response. We will now argue that

$$\Pr\left[A^{\mathrm{Hyb}_j} \Rightarrow 1\right] = \Pr\left[B_j^{Z_j(K,\cdot,\cdot)[\gamma]} \Rightarrow 1\right] \quad \text{and} \tag{2}$$

$$\Pr\left[A^{\mathrm{Hyb}_{j+1}} \Rightarrow 1\right] = \Pr\left[B_j^{\rho_j(\cdot,\cdot)} \Rightarrow 1\right]. \tag{3}$$

where $\rho_j \xleftarrow{\$} \mathsf{RegFunc}(\mathcal{T}_j, n_j, \gamma)$ for $\mathcal{T}_j = \mathcal{T} \times \{0,1\}^{(j-1)\gamma}$ and $n_j = n - (j-1)\gamma$. The first equation may seem to be immediate, but in fact is not because B_j does not simulate exactly Z_j for A when $j < s$. (When $j = s$ the simulation is indeed exact.) Nevertheless, for $j < s$ the distribution of values observed by A when Sep returns $Z_j(T', \mathbf{X})[\gamma]$ is distributed identically to when A's oracle uses Z_j directly. This is because the discrepancy between SepSim's outputs and what would have been computed by Z_j are hidden by π_{j+1}, and SepSim implements a permutation. This justifies (2).

To show the second part, we must argue that, when SepSim uses an oracle ρ_j, the result is that B_j implements Hyb_{j+1}. Here, we require that the composition of SepSim and π_{j+1} in B_j's simulation yields a random tweakable permutation. For each tweak $T' = T \,\|\, X_1^1 \,\|\, \cdots \,\|\, X_{j-1}^{j-1}$, gives rise to a different random function $\rho_j(T', \cdot)$ and permutation $\pi_{j+1}(T', \cdot)$. Moreover, $\mathrm{SepSim}(T', \cdot)$ implements a permutation and so π_{j+1} ends up mapping counter values 1-1 to random values. This justifies (3).

Combining (2) and (3) with (1) and the definition of Γ-prs advantage yields the advantage statement given in the theorem. ∎

Discussion. It is easy to generalize the construction above to work for heterogeneous mixes of separators, meaning that γ varies across stages. Moreover, we have above used a different separator with its own key for each stage, but we can also extend our results to use a variable-input-length separator that can be securely used with a single key on inputs of differing sizes. Finally we note that a corollary of Theorem 2 is that the icicle construction with $s = 2$ gives a way to extend the domain of a PRP on m-bits to one on $m + \gamma$ bits given a Γ-PRS on $m + \gamma$ bits.

6 The Mix-and-Cut Cipher

We now use the results of the prior sections to construct a new, full-security tweakable block cipher that we call Mix-and-Cut. To do so, we show that the Swap-or-Not cipher [13] is a secure 2-PRS for N queries. We then apply the Icicle construction with $\Gamma = 2$, and use Swap-or-Not for each of the n stages.

The Swap-or-Not cipher. Let $F : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^n$ be a PRF family. The Swap-or-Not tweakable block cipher $E_{\mathrm{SN}} : \{0,1\}^k \times \{0,1\}^t \times \{0,1\}^n \to \{0,1\}^n$ built from F is described on the left side of Figure 5. We emphasize that each round of Swap-or-Not results in two calls to F, one to generate the round key and the other to decide whether or not to swap. If F is implemented using CBC-MAC with AES, then a single call to F may result in multiple AES calls, depending on the lengths of the PRF inputs and how they are encoded. (One must also take care to use F that is secure for variable-length inputs.)

Hoang, Morris, and Rogaway [13] analyzed an ideal, untweaked version of this cipher with K_j independently random and the swap decision in each round made by a call to a random function G_j (right side Figure 5). We use the following

algorithm $E_{\mathrm{SN}}(K, T, M)$:
for $j = 1$ to r:
  $K_j \leftarrow F(K, \mathsf{roundkey} \,\|\, j \,\|\, T)$
  $M' \leftarrow K_j \oplus M$
  $\hat{X} \leftarrow \max(M, M')$
  If $F(K, \mathsf{decision} \,\|\, j \,\|\, T \,\|\, \hat{X})[0] = 1$
    $M \leftarrow M'$
Return M

algorithm $E_{\mathrm{SN}^*}(KG, M)$:
$(K_1, \ldots, K_r) \leftarrow \mathrm{par}_n(KG)$
for $j = 1$ to r:
  $M' \leftarrow K_j \oplus M$
  $\hat{X} \leftarrow \max(M, M')$
  If $G_j(\hat{X}) = 1$
    $M \leftarrow M'$
Return M

Fig. 5. (Left) The Swap-or-Not algorithm [13] and **(Right)** its ideal counterpart

restatement of [13, Th. 2], which gives a bound on the non-adaptive cpa security of E_{SN^*}.

Theorem 3 (Hoang-Morris-Rogaway). *Let $N = 2^n$ and let E_{SN^*} be the ideal version Swap-or-Not block cipher with r rounds as defined above. Let A be a non-adaptive cpa adversary making q queries. Then* $\mathbf{Adv}^{\mathrm{ncpa}}_{E_{\mathrm{SN}^*}}(A) \leq \frac{2 \cdot N^{3/2}}{r+2} \cdot \left(\frac{q+N}{2N}\right)^{r/2+1}$.

Unfortunately, the bound above is not very useful when $q \geq N - 1$. We will instead use it with $q = N/2$ in order to help us prove the following theorem, which establishes that E_{SN} is a secure 2-PRS for wN queries for some number w of tweaks.

Theorem 4. *Fix γ, n with $\gamma \,|\, n$ and let $\Gamma = 2^\gamma$, $N = 2^n$. Let E_{SN} be the Swap-or-Not block cipher with r rounds as defined above and using a function F. Let A be a Γ-PRS adversary making N queries across ω distinct tweaks. Then there exists an explicit prf-adversary B for which it holds that*

$$\mathbf{Adv}^{\text{2-prs}}_{E_{\mathrm{SN}}}(A) \leq \frac{2 \cdot \omega \cdot N^{3/2}}{r+2} \cdot \left(\frac{3}{4}\right)^{r/2+1} + \mathbf{Adv}^{\mathrm{prf}}_{F}(B) \,.$$

The adversary B makes at most $2rq\omega$ queries and runs in time that of A plus $\mathcal{O}(2r\omega N)$.

Proof. By Theorem 1 the advantage of A is upper bounded by the cpa advantage of an adversary B' against E_{SN} which makes $N/2$ fixed queries for each (adaptively chosen) tweak T. (Note that $E_{\mathrm{SN}} = E^{-1}_{\mathrm{SN}}$.) Thus

$$\mathbf{Adv}^{\text{2-prs}}_{E_{\mathrm{SN}}}(A) \leq \mathbf{Adv}^{\mathrm{cpa}}_{E_{\mathrm{SN}}}(B')$$

for an adversary B' explicitly specified in the proof of Theorem 1. We now move to a setting where E_{SN} uses, instead of the function F, a random function. This is via a standard reduction yielding that

$$\mathbf{Adv}^{\text{2-prs}}_{E_{\mathrm{SN}}}(A) \leq \mathbf{Adv}^{\mathrm{cpa}}_{E_{\mathrm{SN}}[\rho]}(B') + \mathbf{Adv}^{\mathrm{prf}}_{F}(B)$$

where $E_{\mathrm{SN}}[\rho]$ is E_{SN} except with $F(K, \cdot)$ replaced by a random function ρ. Note that $E_{\mathrm{SN}}[\rho]$ is not yet E_{SN^*} since the latter does not take tweaks. However $E_{\mathrm{SN}}[\rho]$

is equivalent to using E_{SN^*} with a fresh key KG for every tweak T queried by B'. Note also that the set of messages queried to each instance of E_{SN^*} is fixed. We can therefore use a hybrid argument in which one repeatedly applies the ncpa advantage for each independent instance of E_{SN^*} to get that

$$\mathbf{Adv}^{\mathrm{cpa}}_{E_{\mathrm{SN}}[\rho]}(B') \le \sum_{i=1}^{\omega} \mathbf{Adv}^{\mathrm{ncpa}}_{E_{\mathrm{SN}^*}}(B_i) \le \frac{2\cdot\omega\cdot N^{3/2}}{r+2} \cdot \left(\frac{3}{4}\right)^{r/2+1} .$$

The last inequality uses Theorem 3 for $q = N/2$ (the number of queries used by each B_i). ∎

Icicle applied to Swap-or-Not. We now explore the security of the icicle construction when E_{SN} is used as the underlying 2-PRS. Let $N = 2^n$ and let $\mathcal{Z}_{\mathrm{SN}} = \{E^i_{\mathrm{SN}}\}_{i=1}^n$ be a family of SN block ciphers where $E^i_{\mathrm{SN}} : \{0,1\}^k \times \mathcal{T}_i \times \{0,1\}^{n_i} \to \{0,1\}^{n_i}$ denotes the Swap-or-Not cipher as defined above where $n_i = n - (i-1)$ and $\mathcal{T}_i = \mathcal{T} \times \{0,1\}^{i-1}$. Given this, let $\mathrm{Ic}_{1,n}(\mathcal{Z}_{\mathrm{SN}}) : \{0,1\}^{kn} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ denote the block cipher on n bits with tweak space $\mathcal{T}$ constructed by using the icicle construction with $\mathcal{Z}_{\mathrm{SN}}$. For simplicity, we use this construction with the same number of rounds r of Swap-or-Not at each stage (meaning nr rounds of Swap-or-Not for the entire icicle construction). We refer to the resulting construction as the Mix-and-Cut cipher. We can prove the following:

Theorem 5. *Let $N = 2^n$ and $\mathrm{Ic}_{1,n}(\mathcal{Z}_{\mathrm{SN}})$ be the construction described above using r rounds for each Swap-or-Not cipher E^i_{SN}. Each cipher uses the same PRF F (with distinct keys). Let A be a cpa adversary making N queries with w distinct tweaks. Then for the prf adversary B given in the proof below it holds that*

$$\mathbf{Adv}^{\mathrm{cpa}}_{\mathrm{Ic}_{1,n}(\mathcal{Z}_{\mathrm{SN}})}(A) < \frac{7\cdot w\cdot N^{3/2}}{r+2} \cdot \left(\frac{3}{4}\right)^{r/2+1} + n\cdot \mathbf{Adv}^{\mathrm{prf}}_F(B) .$$

B makes $2rnNw$ queries and runs in time that of A plus $\mathcal{O}(2rnNw)$.

Proof. Let $\alpha = (r+2)^{-1}\cdot(3/4)^{r/2+1}$. We first apply Theorem 2 and Theorem 4 to get that

$$\begin{aligned}\mathbf{Adv}^{\mathrm{cpa}}_{\mathrm{Ic}_{1,n}(\mathcal{Z}_{\mathrm{SN}})}(A) &\le \sum_{i=1}^{n} \mathbf{Adv}^{\text{2-prs}}_{E^i_{\mathrm{SN}}}(B_i) \\ &\le \sum_{i=1}^{n} 2\cdot w\cdot 2^{i-1}\cdot N_i^{3/2}\alpha + \sum_{i=1}^{n} \mathbf{Adv}^{\mathrm{prf}}_F(B_i) \qquad (4)\end{aligned}$$

where $N_i = 2^{n-(i-1)}$ and we have used that the number of tweaks queried against the i^{th} stage E^i_{SN} is $w\cdot 2^{i-1}$. First, a standard hybrid argument shows that the prf adversary B that chooses $j \leftarrow\!\!\$\ [1\,..\,n]$ and behaves as B_j is such that $\sum_{i=1}^n \mathbf{Adv}^{\mathrm{prf}}_F(B_i) \le n\cdot\mathbf{Adv}^{\mathrm{prf}}_F(B)$. Turning to analyze the first sum in the right hand side, we first have that

$$\sum_{i=1}^{n} 2^{i-1} \cdot N_i^{3/2} = \sum_{i=1}^{n} 2^{i-1} \cdot 2^{n-(i-1)} \cdot 2^{(n-(i-1))/2} = 2^n \sum_{i=1}^{n} 2^{i/2}$$

$$= 2^n \cdot \frac{2^{(n+1)/2} - \sqrt{2}}{\sqrt{2} - 1}$$

$$< 3.5N^{3/2} .$$

Plugging back into (4) yields the advantage statement of the theorem. ∎

Shuffling interpretation. We can view the cipher above as a shuffle by replacing all uses of the PRF with fresh random coin tosses. We refer to this as the Mix-and-Cut shuffle, and it is given in the left of Figure 1 in the introduction. To shuffle using Mix-and-Cut, "lightly" mix the entire deck in any fashion (e.g., using the Swap-or-Not shuffle for r rounds). Then, cut the deck in half. Lightly mix each half and then cut the halves, yielding four total piles. This process is repeated until all of the cards are in their own piles. At this point, the cards are simply gathered together to form one deck. This shuffle is oblivious assuming the mixing step is oblivious (in the sense of [20]).

Acknowledgements. The authors thank Phillip Rogaway for comments on an earlier draft of this paper as well as the anonymous Crypto 2013 reviewers for their valuable feedback. Ristenpart was supported in part by NSF grant CNS-1065134 and generous gifts from Microsoft and RSA Labs.

References

1. Bellare, M., Impagliazzo, R.: A tool for obtaining tighter security analyses of pseudorandom function based constructions, with applications to prp to prf conversion. Cryptology ePrint Archive, Report 1999/024 (1999), `http://eprint.iacr.org/`
2. Bellare, M., Ristenpart, T., Rogaway, P., Stegers, T.: Format-preserving encryption. In: Jacobson Jr., M.J., Rijmen, V., Safavi-Naini, R. (eds.) SAC 2009. LNCS, vol. 5867, pp. 295–312. Springer, Heidelberg (2009)
3. Bellare, M., Rogaway, P., Spies, T.: Addendum to "the FFX mode of operation for format preserving encryption". Submission to NIST (September 2010)
4. Bellare, M., Rogaway, P., Spies, T.: The FFX mode of operation for format-preserving encryption. Submission to NIST (February 2010)
5. Black, J.A., Rogaway, P.: Ciphers with arbitrary finite domains. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 114–130. Springer, Heidelberg (2002)
6. Brier, E., Peyrin, T., Stern, J.: BPS: a format-preserving encryption proposal. Submission to NIST, `http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/proposedmodes/bps/bps-spec.pdf`
7. Brightwell, M., Smith, H.: Using datatype-preserving encryption to enhance data warehouse security. In: National Information Systems Security Conference, NISSC (1997)

8. Czumaj, A., Kanarek, P., Kutylowski, M., Lorys, K.: Fast generation of random permutations via networks simulation. In: European Symposium on Algorithms, pp. 246–260 (1996)
9. Durstenfeld, R.: Algorithm 235: Random permutation. Communications of the ACM 7(7), 420 (1964)
10. Fisher, R., Yates, F.: Statistical tables for biological, agricultural and medical research. Oliver & Boyd (1938)
11. Granboulan, L., Pornin, T.: Perfect block ciphers with small blocks. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 452–465. Springer, Heidelberg (2007)
12. Hall, C., Wagner, D., Kelsey, J., Schneier, B.: Building PRFs from PRPs. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 370–389. Springer, Heidelberg (1998)
13. Hoang, V.T., Morris, B., Rogaway, P.: An enciphering scheme based on a card shuffle. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 1–13. Springer, Heidelberg (2012)
14. Hoang, V.T., Rogaway, P.: On generalized feistel networks. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 613–630. Springer, Heidelberg (2010)
15. Knuth, D.: The Art of Computer Programming, 3rd edn., vol. 2. Addison-Wesley (1997)
16. Liskov, M., Rivest, R.L., Wagner, D.: Tweakable block ciphers. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 31–46. Springer, Heidelberg (2002)
17. Luby, M., Rackoff, C.: How to construct pseudorandom permutations from pseudorandom functions. SIAM Journal on Computing 17(2) (1988)
18. Morris, B.: Improved mixing time bounds for the Thorp shuffle. arXiv Technical Report 0912.2759 (2009), `http://arxiv.org/abs/0912.2759`
19. Morris, B., Rogaway, P., Stegers, T.: How to encipher messages on a small domain. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 286–302. Springer, Heidelberg (2009)
20. Naor, M., Reingold, O.: On the construction of pseudorandom permutations: Luby-Rackoff revisited. Journal of Cryptology 12(1), 29–66 (1999)
21. Patarin, J.: Generic attacks on feistel schemes. Cryptology ePrint Archive, Report 2008/036 (2008), `http://eprint.iacr.org/2008/036`
22. Ristenpart, T., Rogaway, P.: How to enrich the message space of a cipher. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 101–118. Springer, Heidelberg (2007)
23. Stefanov, E., Shi, E.: Fastprp: Fast pseudo-random permutations for small domains. Cryptology ePrint Archive, Report 2012/254 (2012), `http://eprint.iacr.org/`

Key Homomorphic PRFs and Their Applications

Dan Boneh, Kevin Lewi, Hart Montgomery, and Ananth Raghunathan

Computer Science Department,
Stanford University, Stanford, CA 94305

Abstract. A pseudorandom function $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is said to be key homomorphic if given $F(k_1, x)$ and $F(k_2, x)$ there is an efficient algorithm to compute $F(k_1 \oplus k_2, x)$, where $\oplus$ denotes a group operation on k_1 and k_2 such as `xor`. Key homomorphic PRFs are natural objects to study and have a number of interesting applications: they can simplify the process of rotating encryption keys for encrypted data stored in the cloud, they give one round distributed PRFs, and they can be the basis of a symmetric-key proxy re-encryption scheme. Until now all known constructions for key homomorphic PRFs were only proven secure in the random oracle model. We construct the first provably secure key homomorphic PRFs in the standard model. Our main construction is based on the learning with errors (LWE) problem. We also give a construction based on the decision linear assumption in groups with an ℓ-linear map. We leave as an open problem the question of constructing standard model key homomorphic PRFs from more general assumptions.

Keywords: Pseudorandom functions, Key homomorphism, Learning with errors.

1 Introduction

Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be a secure Pseudorandom Function (PRF) and suppose that the key space $\mathcal{K}$ has a group structure where $\oplus$ denotes the group action. We say that F is *key homomorphic* if given $F(k_1, x)$ and $F(k_2, x)$ there is an efficient procedure that outputs $F\big(k_1 \oplus k_2, x\big)$. That is, the PRF is homomorphic with respect to its key. We show below that key homomorphic PRFs have several important applications that are practically motivated.

Constructing key homomorphic PRFs in the random oracle model is straightforward. Let $\mathbb{G}$ be a finite cyclic group of prime order q and let $H_1 : \mathcal{X} \to \mathbb{G}$ be a hash function modeled as a random oracle. Define the function $F_{\text{DDH}} : \mathbb{Z}_q \times \mathcal{X} \to \mathbb{G}$ as

$$F_{\text{DDH}}(k, x) \leftarrow H_1(x)^k,$$

and observe that $F_{\text{DDH}}(k_1 + k_2, x) = F_{\text{DDH}}(k_1, x) \cdot F_{\text{DDH}}(k_2, x)$. Naor, Pinkas, and Reingold [28] showed that F_{DDH} is a secure PRF in the random oracle model assuming the Decision Diffie-Hellman assumption holds in $\mathbb{G}$. This PRF is clearly key homomorphic.

Similarly, we can construct random oracle key homomorphic PRFs from hard lattice problems. Let $p < q$ be two primes and let $H_2 : \mathcal{X} \to \mathbb{Z}_q^n$ be a hash function modeled as a random oracle. Define the function $F_{\text{LWR}} : \mathbb{Z}_q^n \times \mathcal{X} \to \mathbb{Z}_p$ as

$$F_{\text{LWR}}(\mathbf{k}, x) \leftarrow \lceil \langle H_2(x), \mathbf{k} \rangle \rfloor_p ,$$

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 410–428, 2013.

where $\lceil x \rfloor_p$ denotes rounding an element $x \in \mathbb{Z}_q$ to $\mathbb{Z}_p$ by multiplying it by (p/q) and rounding the result (as defined in Section 2), and $\langle \cdot, \cdot \rangle$ denotes inner product. The function F_{LWR} can be easily shown to be a secure PRF in the random oracle model whenever the Learning with Rounding (LWR) assumption [8] holds. Because rounding is not linear (i.e. it can happen that $\lceil a+b \rfloor_p \neq \lceil a \rfloor_p + \lceil b \rfloor_p$) the function F_{LWR} is not key homomorphic. However, it comes very close and is sufficiently homomorphic for most of our applications. In particular, F_{LWR} is "almost" key homomorphic in the sense that

$$F_{\text{LWR}}(\mathbf{k}_1 + \mathbf{k}_2, x) = F_{\text{LWR}}(\mathbf{k}_1, x) + F_{\text{LWR}}(\mathbf{k}_2, x) + e \tag{1.1}$$

where e is short; namely, $e \in [-1, 1]$.

1.1 Our Contributions

Key Homomorphic PRFs in the Standard Model. We construct the first (almost) key homomorphic PRFs *without* using random oracles. Our main construction, given in Section 5, is a lattice-based almost key homomorphic PRF based on the Learning with Errors (LWE) assumption [33]. The PRF uses two public matrices $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{m \times m}$ where the entries of these matrices are sampled uniformly at random from $\{0,1\}$. The dimension m is derived from the security parameter. The key for the PRF is a single vector $\mathbf{k} \in \mathbb{Z}_q^m$ and its domain is $\{0,1\}^\ell$. The PRF at the point $x = x_1 \cdots x_\ell \in \{0,1\}^\ell$ is defined as

$$F_{\text{LWE}}(\mathbf{k}, x) = \left\lceil \prod_{i=1}^{\ell} \mathbf{A}_{x_i} \cdot \mathbf{k} \right\rfloor_p \in \mathbb{Z}_p^m \,. \tag{1.2}$$

This function satisfies $F_{\text{LWE}}(\mathbf{k}_1 + \mathbf{k}_2, x) = F_{\text{LWE}}(\mathbf{k}_1, x) + F_{\text{LWE}}(\mathbf{k}_2, x) + \mathbf{e}$ where the error term $\mathbf{e} \in [-1, 1]^m$. Therefore this function is almost key homomorphic in the same sense as F_{LWR}, which is sufficient for most of our applications. We prove that F_{LWE} is a secure PRF based on the LWE assumption in the standard model.

The construction in Eq. (1.2) is closely related to an elegant non-key homomorphic PRF due to Banerjee, Peikert, and Rosen [8], but is technically quite different from it. The secret key in [8] is a collection of ℓ matrices while our secret key is only a single vector $\mathbf{k} \in \mathbb{Z}_q^m$. The public parameters in [8] consist of one matrix while our public parameters consist of two matrices. An important step in our proof of security requires that the two public matrices $\mathbf{A}_0, \mathbf{A}_1$ used in our PRF be low-norm matrices (e.g. binary) and this poses a challenge in proving security from the standard LWE assumption.

To prove security we define a variant of LWE called the *non-uniform* LWE problem and show that it is at least as hard as the standard LWE problem. Recall that the standard LWE assumption states that for a random $\mathbf{s} \in \mathbb{Z}_q^n$, the following two oracles are indistinguishable:

$$\mathcal{O}_{\text{LWE}} : \ \left(\mathbf{v}_i \xleftarrow{\text{R}} \mathbb{Z}_q^n, \quad \langle \mathbf{v}_i, \mathbf{s} \rangle + \chi_i \right) \qquad \text{and} \qquad \mathcal{O}_{\$} : \ \left(\mathbf{v}_i \xleftarrow{\text{R}} \mathbb{Z}_q^n, \quad x_i \xleftarrow{\text{R}} \mathbb{Z}_q \right)$$

where χ_i is sampled from a suitable low-norm noise distribution. We show that the LWE assumption implies that these two oracles are indistinguishable even when the vectors $\mathbf{v}_i$ are sampled from certain distributions of *low norm* vectors in $\mathbb{Z}_q^n$ or even

as binary vectors in $\{0,1\}^n$. However, the dimension n must be increased—in general, the lower the norm of each $\mathbf{v}_i$, the larger n needs to be. While this low norm version of the LWE assumption is precisely what we need to prove security of the PRF F_{LWE}, this assumption may be of independent interest and useful in other settings.

Key-Homomorphic PRFs from ℓ-Linear Maps. In the full version, we present an algebraic ℓ-bit key homomorphic PRF built from ℓ-linear maps $\hat{e} : \mathbb{G}^\ell \to \mathbb{G}_\ell$, where $\mathbb{G}_\ell$ is the ℓ^{th} target group. PRF security is based on the ℓ-decision linear assumption [34, 23] in $\mathbb{G}$. For a generator $g \in \mathbb{G}$, the public parameters for the PRF are $\mathsf{pp} = \left(g^{\mathbf{A}_0}, g^{\mathbf{A}_1}\right)$ where $\mathbf{A}_0, \mathbf{A}_1$ are two matrices in $\mathbb{Z}_p^{\ell \times \ell}$ generated at random (here, the notation $g^{\mathbf{A}_0}$ denotes component-wise exponentiation). The secret key for the PRF is a single vector $\mathbf{k} \in \mathbb{Z}_p^\ell$. The PRF at the point $x = x_1 \ldots x_\ell \in \{0,1\}^\ell$ is defined as

$$F_{\text{DLIN}}(\mathbf{k}, x) = (g_\ell)^{\mathbf{w}} \quad \in (\mathbb{G}_\ell)^\ell \qquad \text{where} \qquad \mathbf{w} = \mathbf{A}_{x_1} \cdots \mathbf{A}_{x_\ell} \cdot \mathbf{k} \quad \in \mathbb{Z}_p^\ell \quad (1.3)$$

where g_ℓ is a generator of $\mathbb{G}_\ell$. Evaluating the PRF at the point x given the public parameters $\mathsf{pp} = \left(g^{\mathbf{A}_0}, g^{\mathbf{A}_1}\right)$ and key $\mathbf{k}$ can be done using a graded ℓ-linear map as explained in the full version. The PRF $F_{\text{DLIN}}(\mathbf{k}, x)$ is clearly homomorphic with respect to the secret key $\mathbf{k}$.

This PRF is related to the Naor-Reingold DDH-based PRF [29], but since the DDH assumption is false in groups with an ℓ-linear map, the relation is closer to the Lewko-Waters [25] variant which is proven secure under the ℓ-decision linear assumption in $\mathbb{G}$. The secret key in the Lewko-Waters PRF consists of ℓ secret matrices while in construction (1.3) the secret key is only a single vector in $\mathbf{k} \in \mathbb{Z}_p^\ell$ and this enables the key-homomorphic property. However, our construction inherently requires an ℓ-linear map in $\mathbb{G}$ whereas [25] did not. Unfortunately, we cannot use the ℓ-linear map candidate of Garg, Gentry, and Halevi [20] to instantiate the construction because the ℓ-decision linear problem is easy for that proposal. We therefore view our lattice-based construction as our primary key homomorphic PRF and await other ℓ-linear map candidates to instantiate our second scheme.

1.2 Key Homomorphic PRFs: Applications

Our interest in key homomorphic PRFs stems from a number of real-world applications for such functions. We describe them briefly here and discuss some of these applications in more detail in Section 6. Due to space constraints the remaining applications have been deferred to the full version.

Distributed PRFs. In a one-time password system such as RSA SecurID, users are given a small cryptographic token containing a PRF secret key. The token displays PRF outputs that are used as one-time passwords. An authentication server verifies a given one-time password by comparing it to its own computation of the PRF value using the same PRF secret key. Since the server knows the secret PRF keys for all users, these authentication servers have become a prime target for attacks [17]. In response, RSA introduced *Distributed Credential Protection* where PRF keys are shared among two or more key servers and all servers have to be compromised to recover the keys. Currently

this design does not provide true key splitting since the PRF in use is AES for which there is no known simple key splitting mechanism.

Key homomorphic PRFs give a clean solution to distributing PRF keys using a simple communication pattern: a client who wants to evaluate the PRF at a point x sends a single short message to each key server and receives a single response back from each key server. No interaction between the key servers is needed. For an n-out-of-n sharing, key server i stores a random key k_i and the overall PRF key is $k = k_1 \oplus \cdots \oplus k_n$, where $\oplus$ is the group action over the key space. To evaluate $F(k, x)$ the client sends x to all key servers and each server responds with $y_i = F(k_i, x)$. The client combines the results to obtain $F(k, x)$ using the key homomorphism property. To provide t-out-of-n sharing, the client first "multiplies" the responses from the key servers by the appropriate Lagrange coefficients and then "adds" the results using the key homomorphism property. We give the details in Section 6.1. This application still works with an *almost* key homomorphic PRF as long as the PRF range is sufficiently larger than the homomorphism error term. The output is then defined as only the high order bits of the computed value $F(k, x)$ so as to eliminate the homomorphism error.

Symmetric-Key Proxy Re-encryption. Key homomorphic PRFs provide the symmetric-key analogue of public-key proxy re-encryption [12, 7, 15, 6, 26]. Given a message from a client encrypted under one symmetric key, a proxy can translate that ciphertext to a different symmetric key (associated with another client) without knowledge of either key. To do so, the proxy is provided with a short re-encryption token Δ that enables it to transform the symmetric encryption of the data m from key k to key k' without knowing either key.

A key homomorphic PRF directly gives a symmetric-key proxy re-encryption scheme. To see how, let $F(k, m)$ be a key homomorphic PRF satisfying $F(k \oplus k', x) = F(k, x) \otimes F(k', x)$ where both $\oplus$ and $\otimes$ are group operations. Suppose the data m is encrypted using randomized counter mode based on F—that is, the j^{th} block of m is encrypted as $c_j \leftarrow m_j \otimes F(k, N + j)$ where N is an encryption nonce. Now, to re-encrypt from key k to key k', the client sends the re-encryption token $\Delta = -k \oplus k'$ to the proxy. The proxy computes the following on every ciphertext block:

$$c'_j \leftarrow c_j \otimes F(\Delta, N + j).$$

By the key homomorphism property, $c'_j = m_j \otimes F(k, N + j) \otimes F(\Delta, N + j) = m_j \otimes F(k \oplus \Delta, N + j) = m_j \otimes F(k', N + j)$ and therefore c'_j is the encryption of m_j under key k', as required. We discuss the security of this construction in Section 6.

This application works equally well with an *almost* key homomorphic PRF except that we need to pad each message block m_j with a constant number of zeros on the right to ensure that the small additive homomorphic error term $\mathbf{e}$ does not affect the encrypted plaintext after several re-encryptions.

We note that basic symmetric-key proxy re-encryption can also be done using a seed-homomorphic pseudorandom generator (PRG)—that is, a PRG $G : \mathcal{S} \to \mathcal{Y}$ such that $G(s_2)$ can be efficiently computed from $G(s_1)$ and $\Delta = -s_1 \oplus s_2$. We give examples of such PRGs in Section 3.1. However, encrypting with a PRG is only one-time secure, whereas randomized counter-mode using a PRF provides security against chosen-plaintext attacks, thereby enabling a single key to encrypt multiple messages. We also

note that the encryption scheme can be made to provide integrity without disrupting the key homomorphism property by using "MAC then encrypt with counter-mode," which is known to provide secure authenticated encryption (see e.g., [24]).

Updatable Encryption. Symmetric-key proxy re-encryption built from key homomorphic PRFs elegantly solves a common problem facing companies who store encrypted data in the cloud. Let m be some data and suppose the company stores the symmetric encryption of m under key k in the cloud. Any employee who knows k has access to m. As employees leave the company there is a need to rotate the encryption key (i.e. re-encrypt m under a new key k') to ensure that ex-employees lose access to the data. Often key rotation happens at fixed time intervals (e.g. once a month).

One naïve approach is to download the entire ciphertext from the cloud, re-encrypt under a new key, and upload the new ciphertext to the cloud. If the cloud provider is trusted to delete the old ciphertext then this ensures that employees who leave the company lose access even if they are able to access the current data stored in the cloud. Unfortunately, downloading all the data from the cloud just for the purpose of key rotation results in considerable wasted bandwidth and cost. A better solution is to encrypt the data m using a symmetric-key proxy re-encryption scheme and use the cloud as the proxy holding the company's encrypted data. Now, by simply sending to the cloud the re-encryption token $\Delta = -k \oplus k'$, the cloud can translate the ciphertext from key k to key k' in place without doing any large data transfers. As before, if the cloud is trusted to delete the old re-encryption tokens Δ and the old versions of the ciphertext, then employees who leave the company lose access to m even if they can access the current data stored with the cloud.

We note that key-rotation in the cloud can potentially be done using ad-hoc solutions such as nested encryption, but these solutions result in increased storage needs or increased decryption time or do not fully prevent a revoked employee from decrypting cloud data. A fast key homomorphic PRF would provide a clean solution that does not increase storage requirements and has no impact on encryption or decryption time.

PRFs Secure against Related-Key Attacks. A related-key attack (RKA) on a PRF models a situation where an adversary is able to manipulate the secret key used in the PRF. Bellare and Cash [9] construct RKA-secure PRFs under the Decision Diffie-Hellman assumption and also under the decision linear assumption. One important ingredient in their constructions is a PRF that satisfies "key malleability"—informally, an adversary can transform an output of the PRF on a secret key $k \in \mathcal{K}$ and input $x \in \mathcal{X}$ into an output of the PRF on a related key $\phi(k)$ (where $\phi : \mathcal{K} \to \mathcal{K}$ is a member of a class of functions Φ) and input x *without having access to* k. For PRFs, key homomorphism implies key malleability with respect to the class $\Phi_\oplus = \{\phi(k) = k \oplus k'\}_{k' \in \mathcal{K}}$, where $\oplus$ represents the group action over $\mathcal{K}$. However, the converse does not hold in general— key malleability over $\Phi_\oplus$ is not known to imply key homomorphism.

Bellare and Cash give several constructions of RKA-secure PRFs based on various standard assumptions which are secure for certain restricted classes Φ of related-key deriving functions. We show in the full version that any key homomorphic PRF that

satisfies an additional syntactic property can be used to construct an RKA-secure PRF for a larger class Φ than in [9].

1.3 Related Work

Distributed PRFs were first considered by Naor and Pinkas [28] for the purpose of distributing the Kerberos key distribution center. They proposed the simple random oracle construction F_{DDH} which is key homomorphic. Nielsen [30] and D'Arco and Stinson [18] constructed robust distributed PRFs, but evaluation requires interaction between the key servers and multiple rounds of communication. Dodis [19] constructs distributed PRFs and VRFs under a strong assumption and also requires multiple rounds of communication to evaluate the PRF.

Much recent work has focused on preventing related key attacks [10, 9, 11]. Key homomorphic PRFs are on the other end of the spectrum where key homomorphism is encouraged in support of specific applications.

Syalim, Nishide, and Sakurai [37] describe a symmetric proxy re-encryption scheme based on the all or nothing transform (AONT) in the random oracle model. Cook and Keromytis [16] propose to use double encryption to provide one-hop symmetric proxy re-encryption.

2 Preliminaries

Rounding. We define $\lceil \cdot \rfloor$ to round a real number to the nearest integer. For integers q and p where $q \geq p \geq 2$, we define the function $\lceil \cdot \rfloor_p : \mathbb{Z}_q \to \mathbb{Z}_p$ as $\lceil x \rfloor_p = \lceil (p/q) \cdot \bar{x} \rfloor \pmod p$ where $\bar{x} \in \{0, 1, \ldots, q-1\}$ is the integer congruent to $x \pmod q$. For a vector $\mathbf{v} \in \mathbb{Z}_q^m$, we define $\lceil \mathbf{v} \rfloor_p$ as the vector in $\mathbb{Z}_p^m$ obtained by rounding each coordinate of the vector individually. A probability distribution χ over $\mathbb{R}$ is said to be *B-bounded* if it holds that $\Pr_{x \leftarrow \chi}[|x| > B]$ is negligible in the security parameter.

PRFs and PRGs. Recall that a *pseudorandom generator* (PRG) is an efficiently computable function $G : \mathcal{S} \to \mathcal{R}$ such that for uniform s in $\mathcal{S}$ and uniform r in $\mathcal{R}$, the distribution $\{G(s)\}$ is computationally indistinguishable from the distribution $\{r\}$. A *pseudorandom function* (PRF) [22] is an efficiently computable function $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ such that for a uniform k in $\mathcal{K}$ and a uniform function $f : \mathcal{X} \to \mathcal{Y}$, an oracle for $F(k, \cdot)$ is computationally indistinguishable from an oracle for $f(\cdot)$. In this paper, we allow our PRFs and PRGs to be further paramaterized by an additional public parameter pp, which will be clear from context. We let $\mathbf{Adv}^{\mathsf{PRF}}[F, \mathcal{A}]$ denote the advantage of adversary $\mathcal{A}$ in distinguishing the PRF F (along with its public parameters) from a random function $f : \mathcal{X} \to \mathcal{Y}$.

Learning With Errors (LWE) Assumption. The LWE problem was introduced by Regev [33] who showed that solving the LWE problem *on average* is as hard as (quantumly) solving several standard lattice problems *in the worst case*.

Definition 2.1 (Learning With Errors). *For integers $q = q(n) \geq 2$ and a noise distribution $\chi = \chi(n)$ over $\mathbb{Z}_q$, the learning with errors problem $(\mathbb{Z}_q, n, \chi)$-LWE is to distinguish between the following pairs of distributions:*

$$\{\mathbf{A}, \mathbf{A}^\intercal \mathbf{s} + \boldsymbol{\chi}\} \qquad \textit{and} \qquad \{\mathbf{A}, \mathbf{u}\},$$

where $m = \text{poly}(n)$, $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\boldsymbol{\chi} \leftarrow \chi^m$, *and* $\mathbf{u} \leftarrow \mathbb{Z}_q^m$. *We refer to the* m *columns of the matrix* $\mathbf{A}$ *as the* LWE *sample points.*

Regev [33] shows that for a certain noise distribution $\chi = \overline{\Psi}_\alpha$[1], for n polynomial in λ and $q > 2\sqrt{n}/\alpha$, the LWE problem is as hard as the worst-case SIVP and GapSVP under a quantum reduction (see also [31, 13]). These results have been extended to show that $\mathbf{s}$ can be sampled from a low norm distribution (in particular, from the noise distribution $\boldsymbol{\chi}$) and the resulting problem is as hard as the basic LWE problem [5]. Similarly, the noise distribution $\boldsymbol{\chi}$ can be a simple low-norm distribution [27].

3 Key Homomorphic PRFs and Seed Homomorphic PRGs

In this section, we define key homomorphic PRFs and "almost" key homomorphic PRFs. We also introduce the concept of seed homomorphic PRGs and give example instantiations from standard assumptions.

Definition 3.1 (Key homomorphic PRF). *Consider an efficiently computable function* $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ *such that* $(\mathcal{K}, \oplus)$ *and* $(\mathcal{Y}, \otimes)$ *are both groups. We say that the tuple* $(F, \oplus, \otimes)$ *is a key homomorphic PRF (KHPRF) if the following two properties hold:*

1. F *is a secure pseudorandom function.*
2. *For every* $k_1, k_2 \in \mathcal{K}$ *and every* $x \in \mathcal{X}$, $F(k_1, x) \otimes F(k_2, x) = F(k_1 \oplus k_2, x)$.

In Section 1 we gave an example of a key homomorphic PRF in the random oracle model due to Naor, Pinkas, and Reingold [28]. While property 2 is very desirable, it is helpful to also modify the homomorphic requirement to only being approximately correct when $\mathcal{Y} = \mathbb{Z}_p^m$. We call this variant an *almost* key homomorphic PRF (AKHPRF). An AKHPRF has a parameter γ that reflects the amount of error allowed in the homomorphism.

Definition 3.2 (γ-Almost key homomorphic PRF). *Let* $F : \mathcal{K} \times \mathcal{X} \to \mathbb{Z}_p^m$ *be an efficiently computable function such that* $(\mathcal{K}, \oplus)$ *is a group. We say that the tuple* $(F, \oplus)$ *is a* γ*-almost key homomorphic PRF (*γ*-AKHPRF) if the following two properties hold:*

1. F *is a secure pseudorandom function.*
2. *For every* $k_1, k_2 \in \mathcal{K}$ *and every* $x \in \mathcal{X}$, *there exists a vector* $\mathbf{v} \in [-\gamma, \gamma]^m$ *such that*
$$F(k_1, x) + F(k_2, x) = F(k_1 \oplus k_2, x) + \mathbf{v} \pmod{p}.$$

For example, the function F_{LWR} from Section 1 is 1-almost key homomorphic. Such γ-almost key homomorphic functions for small γ are sufficient for the applications we have in mind.

[1] For an $\alpha \in (0, 1)$ and a prime q, let $\overline{\Psi}_\alpha$ denote the distribution over $\mathbb{Z}_q$ of the random variable $\lceil qX \rfloor \pmod q$ where X is a normal random variable with mean 0 and standard deviation $\alpha/\sqrt{2\pi}$.

3.1 Seed Homomorphic Pseudorandom Generators

To explain our PRF construction it is instructive to first consider pseudorandom generators (PRGs) that are homomorphic with respect to their seed.

Definition 3.3 (Seed homomorphic PRG). *An efficiently computable function $G : \mathcal{X} \to \mathcal{Y}$, where $(\mathcal{X}, \oplus)$ and $(\mathcal{Y}, \otimes)$ are groups, is is said to be seed homomorphic if the following two properties hold:*

1. *G is a secure PRG.*
2. *For every $s_1, s_2 \in \mathcal{X}$ we have that $G(s_1) \otimes G(s_2) = G(s_1 \oplus s_2)$.*

3.2 Examples of Seed Homomorphic PRGs

We give two example seed homomorphic PRGs, one based on the Decision Diffie-Hellman (DDH) assumption and the other based on lattices.

Seed Homomorphic PRGs from DDH. Let $\mathbb{G}$ be a group of order p in which the DDH assumption holds. Consider the following PRG $G_{\text{DDH}} : \mathbb{Z}_p \to \mathbb{G} \times \mathbb{G}$ and public parameters pp being a pair $g, h \xleftarrow{R} \mathbb{G}$:

$$G_{\text{DDH}}(s) = (g^s, h^s)$$

Security of this PRG follows immediately from the DDH assumption: when s is uniform in $\mathbb{Z}_p$ then $G_{\text{DDH}}(s)$ is indistinguishable from a random sample in $\mathbb{G} \times \mathbb{G}$. It should also be clear that this PRG is seed homomorphic since for all $s_1, s_2 \in \mathbb{Z}_p$, $G_{\text{DDH}}(s_1 + s_2) = G_{\text{DDH}}(s_1) \cdot G_{\text{DDH}}(s_2)$ where $\cdot$ is component-wise multiplication.

Almost Seed Homomorphic PRGs from LWR. Let $p < q$ and $n < m$ be parameters. Then the following PRG $G_{\text{LWR}} : \mathbb{Z}_q^n \to \mathbb{Z}_p^m$, with public parameters pp being a random matrix $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}$, is secure assuming the Learning With Rounding (LWR) problem is hard for the given parameters p, q, n, m:

$$G_{\text{LWR}}(\mathbf{s}) = \lceil \mathbf{A}^\intercal \cdot \mathbf{s} \rfloor_p$$

While this PRG is not seed homomorphic, it is close to seed homomorphic in the following sense: $G_{\text{LWR}}(\mathbf{s}_1 + \mathbf{s}_2) = G_{\text{LWR}}(\mathbf{s}_1) + G_{\text{LWR}}(\mathbf{s}_2) + \mathbf{e}$ where $\mathbf{e} \in [-1, 1]^m$. An almost seed homomorphic PRG can be used for updatable encryption by appending a few zeros to each plaintext block before encryption so that low-order bits can be dropped during decryption thereby eliminating any errors resulting from the *almost* homomorphic property.

Key Homomorphic PRFs from the GGM Construction. Consider a seed homomorphic PRG $G : \mathcal{X} \to \mathcal{X} \times \mathcal{X}$ where $(\mathcal{X}, \oplus)$ is a group. Since the output of $G(s)$ is in $\mathcal{X} \times \mathcal{X}$ let us write $G_0(s)$ for the left half of $G(s)$ and write $G_1(s)$ for the right half. We can now construct the GGM PRF [22] with key space $\mathcal{X}$ and input space $\{0,1\}^\ell$ as follows:

$$F_{\text{GGM}}(k,\ x = x_1 \cdots x_\ell) = G_{x_\ell}\big(\, G_{x_{\ell-1}}\big(\cdots G_{x_2}(\, G_{x_1}(k)\,)\cdots\big)\big) \tag{3.1}$$

The standard GGM proof shows that if G is a secure PRG then F is a secure PRF. Now suppose further that the input and output homomorphisms of G are *compatible*—that is, for all $s_1, s_2 \in \mathcal{X}$ and $b \in \{0,1\}$ we have that $G_b(s_1 \oplus s_2) = G_b(s_1) \oplus G_b(s_2)$. Then it is not difficult to see that F_{GGM} is key homomorphic.

Unfortunately, G_{DDH} and G_{LWR} defined in Section 3.2 above cannot be used directly in construction (3.1). The problem is that these generators do not compose as needed for construction (3.1). In the next few sections we show how to overcome these difficulties while preserving the key homomorphic or almost key homomorphic property of the resulting PRFs.

4 Learning With Errors with Low-Norm Samples

Before we construct our lattice-based key homomorphic PRF, we first present a variant of the learning with errors assumption needed to prove security. Recall that the basic LWE assumption [33] reviewed in Section 2 states that the distribution $\{\mathbf{A},\ \mathbf{A}^\intercal\mathbf{s} + \boldsymbol{\chi}\}$ is indistinguishable from the distribution $\{\mathbf{A},\ \mathbf{u}\}$ where the columns of $\mathbf{A}$ are sampled uniformly in $\mathbb{Z}_q^n$ and $\mathbf{s}$ is uniform in $\mathbb{Z}_q^n$.

In this section we introduce a variant of the learning with errors (LWE) problem in which the the columns of $\mathbf{A}$ (i.e., the LWE sample points) are sampled from a *non-uniform* distribution $\boldsymbol{\eta}$ over $\mathbb{Z}_q^n$. We call this variant *Non-uniform Learning with Errors*, or NLWE for short, and show that for suitable parameters it is as hard as the basic LWE problem. In what follows we let k denote the dimension of the NLWE problem and let n denote the dimension of the LWE problem. We also write $\boldsymbol{\eta}^m$ to denote m independent samples from the distribution $\boldsymbol{\eta}$.

Definition 4.1 (Non-uniform Learning with Errors). *For an integer $q = q(k) \geq 2$, a noise distribution $\chi = \chi(k)$ over $\mathbb{Z}_q$, and a distribution $\boldsymbol{\eta}$ over $\mathbb{Z}_q^k$, the* **non-uniform learning with errors problem** $(\mathbb{Z}_q, k, \chi, \boldsymbol{\eta})$-NLWE *is to distinguish between the two distributions:*

$$\{\mathbf{A},\ \mathbf{A}^\intercal\mathbf{s} + \boldsymbol{\chi}\} \qquad \textit{and} \qquad \{\mathbf{A},\ \mathbf{u}\},$$

where $m = poly(k)$, $\mathbf{A} \leftarrow \boldsymbol{\eta}^m$ (so that $\mathbf{A} \in \mathbb{Z}_q^{k \times m}$), $\mathbf{s} \leftarrow \mathbb{Z}_q^k$, $\boldsymbol{\chi} \leftarrow \chi^m$, and $\mathbf{u} \leftarrow \mathbb{Z}_q^m$.

We show that for certain choices of the distribution $\boldsymbol{\eta}$ there is a reduction from $(\mathbb{Z}_q, n, \chi)$-LWE to $(\mathbb{Z}_q, k, \chi, \boldsymbol{\eta})$-NLWE for some $k \geq n$. Consequently, the NLWE problem is at least as hard as the LWE problem. In particular, we show that for suitable parameters, NLWE is as hard as LWE for the following distributions $\boldsymbol{\eta}$:

- $\boldsymbol{\eta}_{\mathsf{Bin}(k)}$: the uniform distribution on $\{0,1\}^k$ for sufficiently large k,
- $\mathcal{D}_{\mathbb{Z}^k,\sigma}$: a discrete Gaussian on $\mathbb{Z}^k$ with a sufficiently large k and standard deviation σ,
- $\boldsymbol{\eta}_V$: a uniform distribution over a sufficiently large linear subspace V of $\mathbb{Z}_q^k$.

More generally, we show that NLWE is as hard as LWE for any distribution $\boldsymbol{\eta}$ which is *coset sampleable* as defined next.

Definition 4.2 (Coset Sampleable Distributions). *For integers $q = q(k)$ and $n = n(k)$ we say that a distribution $\boldsymbol{\eta} = \boldsymbol{\eta}(k)$ over $\mathbb{Z}_q^k$ is* n-coset sampleable *if there are two PPT algorithms* $(\mathsf{MatrixGen}, \mathsf{SamplePre})$ *such that:*

- $\mathsf{MatrixGen}(1^k, n, q)$ *outputs a matrix* $\mathbf{M} \in \mathbb{Z}_q^{n \times k}$ *and auxiliary data* $\mathbf{T}$*,*
- $\mathsf{SamplePre}(\mathbf{z} \in \mathbb{Z}_q^n,\ \mathbf{T})$ *outputs a* $\mathbf{y} \in \mathbb{Z}_q^k$ *satisfying* $\mathbf{My} = \mathbf{z}$*. Moreover, if* $\mathbf{z}$ *is distributed* uniformly *in* $\mathbb{Z}_q^n$ *then the output of* $\mathsf{SamplePre}(\mathbf{z}, \mathbf{T})$ *is distributed statistically close to* $\boldsymbol{\eta}$*.*

The following theorem shows that $(\mathbb{Z}_q, k, \chi, \boldsymbol{\eta})$-NLWE is as hard as $(\mathbb{Z}_q, n, \chi)$-LWE for any n-coset sampleable distribution $\boldsymbol{\eta}$.

Theorem 4.3. *Let $\boldsymbol{\eta} = \boldsymbol{\eta}(k)$ be an* n-coset sampleable distribution*. Suppose there is a PPT algorithm $\mathcal{A}$ that decides the $(\mathbb{Z}_q, k, \chi, \boldsymbol{\eta})$-*NLWE *problem with advantage $\varepsilon(k)$. Then there is a PPT algorithm $\mathcal{B}$ that decides the $(\mathbb{Z}_q, n, \chi)$-*LWE *problem with the same advantage $\varepsilon(k)$.*

Proof. Algorithm $\mathcal{B}$ takes an LWE instance $(\mathbf{A},\ \mathbf{v})$ as input and needs to decide whether this input is sampled from $\{\mathbf{A},\ \mathbf{A}^\intercal\mathbf{s}+\boldsymbol{\chi}\}$ or from $\{\mathbf{A},\ \mathbf{u}\}$ where $\mathbf{A}$ is uniform in $\mathbb{Z}_q^{n \times m}$ and $\mathbf{u}$ is uniform in $\mathbb{Z}_q^m$. Algorithm $\mathcal{B}$ translates $(\mathbf{A},\ \mathbf{v})$ into an NLWE instance $(\mathbf{B},\ \mathbf{v}')$ and then runs $\mathcal{A}$ on $(\mathbf{B},\ \mathbf{v}')$. It works as follows:

1. Choose a random $\mathbf{r} \leftarrow \mathbb{Z}_q^k$ and run $\mathsf{MatrixGen}(1^k, n, q)$ to obtain a matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times k}$ and $\mathbf{T}$.
2. For each column $\mathbf{a} \in \mathbb{Z}_q^n$ of $\mathbf{A}$ run $\mathsf{SamplePre}(\mathbf{a}, \mathbf{T})$ to obtain $\mathbf{b} \in \mathbb{Z}_q^k$ such that $\mathbf{Mb} = \mathbf{a}$. Assemble all such $\mathbf{b}$ into a matrix $\mathbf{B} \in \mathbb{Z}_q^{k \times m}$. Then $\mathbf{MB} = \mathbf{A}$.
3. Set $\mathbf{v}' \leftarrow \mathbf{v} + \mathbf{B}^\intercal\mathbf{r} \in \mathbb{Z}_q^m$.
4. Run $\mathcal{A}$ on input $(\mathbf{B},\ \mathbf{v}')$ and output whatever $\mathcal{A}$ outputs.

It remains to show that $(\mathbf{B},\ \mathbf{v}')$ is properly distributed as a $(\mathbb{Z}_q, k, \chi, \boldsymbol{\eta})$-NLWE problem instance. First, by the definition of $\mathsf{SamplePre}$, since the columns of $\mathbf{A}$ are uniform in $\mathbb{Z}_q^n$, the columns of $\mathbf{B}$ are statistically close to $\boldsymbol{\eta}$. Second, if the input $\mathbf{v}$ is uniform in $\mathbb{Z}_q^m$ then clearly $\mathbf{v}' = \mathbf{v} + \mathbf{B}^\intercal\mathbf{r}$ is uniform in $\mathbb{Z}_q^m$. Third, if the input $\mathbf{v}$ satisfies $\mathbf{v} = \mathbf{A}^\intercal\mathbf{s} + \boldsymbol{\chi}$ then $\mathbf{v}'$ satisfies $\mathbf{v}' = \mathbf{B}^\intercal(\mathbf{M}^\intercal\mathbf{s} + \mathbf{r}) + \boldsymbol{\chi}$ because $\mathbf{A}^\intercal = \mathbf{B}^\intercal\mathbf{M}^\intercal$ and

$$\mathbf{v}' = \mathbf{v} + \mathbf{B}^\intercal\mathbf{r} = \mathbf{A}^\intercal\mathbf{s} + \boldsymbol{\chi} + \mathbf{B}^\intercal\mathbf{r} = \mathbf{B}^\intercal\mathbf{M}^\intercal\mathbf{s} + \boldsymbol{\chi} + \mathbf{B}^\intercal\mathbf{r} = \mathbf{B}^\intercal(\mathbf{M}^\intercal\mathbf{s} + \mathbf{r}) + \boldsymbol{\chi}\,.$$

Therefore $(\mathbf{B},\ \mathbf{v}')$ is a proper NLWE instance where the secret vector is $\mathbf{s}' = \mathbf{M}^\intercal\mathbf{s} + \mathbf{r}$ which is clearly uniform in $\mathbb{Z}_q^k$. It follows that $\mathcal{B}$ decides NLWE with the same advantage as $\mathcal{A}$ decides LWE. □

Remark 4.4. We note that while our definition of the NLWE problem requires that the secret vector $\mathbf{s} \in \mathbb{Z}_q^k$ be uniform in $\mathbb{Z}_q^k$, the proof of Theorem 4.3 can be adapted to show that NLWE problem is hard even when $\mathbf{s}$ is non-uniform and in particular distributed as $\{\mathbf{M}^\intercal\mathbf{s}'\}$ where $\mathbf{s}'$ is distributed as the secret vector in the LWE problem (e.g., uniform in $\mathbb{Z}_q^n$). The proof is adapted to a non-uniform $\mathbf{s}$ by eliminating the randomization vector $\mathbf{r}$.

Next we show specific distributions $\boldsymbol{\eta}$ for which the corresponding NLWE problem is as hard as LWE.

NLWE with Uniform Samples in $\{0,1\}^k$. Let $\boldsymbol{\eta}_{\mathsf{Bin}(k)}$ be the uniform distribution on $\{0,1\}^k$. We show that if the $(\mathbb{Z}_q, n, \chi)$-LWE problem is hard then so is the non-uniform LWE problem where the columns of $\mathbf{A}$ are random *binary* vectors and the dimension is increased from n to $n\lceil \log_2 q \rceil$. The proof uses bit decomposition as in [14] and also in [2, 13].

Corollary 4.5. *Let $q = q(n)$ be an integer such that $\frac{2^{\lceil \log q \rceil} - q}{q}$ is negligible (i.e., q is close to a power of 2). Let $k = n\lceil \log_2 q \rceil$. Then the $(\mathbb{Z}_q,\ n\lceil \log_2 q \rceil,\ \chi,\ \boldsymbol{\eta}_{\mathsf{Bin}(k)})$-NLWE problem is at least as hard as the $(\mathbb{Z}_q, n, \chi)$-LWE problem.*

Proof. By Theorem 4.3 it suffices to show that $\boldsymbol{\eta}_{\mathsf{Bin}(k)}$ is coset sampleable. Let $\mathbf{m} \in \mathbb{Z}_q^{\lceil \log q \rceil}$ be the vector $(1, 2, 2^2, \ldots, 2^{\lceil \log q \rceil - 1})$, and let $\mathbf{M} \in \mathbb{Z}_q^{n \times k}$ be the matrix $\mathbf{M} = \mathbf{m} \otimes \mathbf{I}_n$, where $\otimes$ denotes the tensor product. Algorithms $\mathsf{MatrixGen}$ and $\mathsf{SamplePre}$ are defined as follows:

- $\mathsf{MatrixGen}(1^k, n, q)$ simply outputs $\mathbf{M} = \mathbf{m} \otimes \mathbf{I}_n \in \mathbb{Z}_q^{n \times k}$ and $\mathbf{T} = (n, q)$,
- $\mathsf{SamplePre}(\mathbf{z} \in \mathbb{Z}_q^n,\ \mathbf{T})$ outputs a vector $\mathbf{y}$ in $\{0,1\}^k$ by setting the entry in position $(i + j\lceil \log_2 q \rceil)$ of $\mathbf{y}$ to the i^{th} bit of the j^{th} entry of the input vector $\mathbf{z}$, for $j = 0, \ldots, n-1$ and $i = 0, \ldots, \lceil \log_2 q \rceil - 1$.

By construction $\mathbf{My} = \mathbf{z}$. Moreover, if $\mathbf{z}$ is uniformly distributed in $\mathbb{Z}_q^n$ then a standard calculation shows that the statistical distance between the distribution $\mathsf{SamplePre}(\mathbf{z}, \mathbf{T})$ and $\boldsymbol{\eta}_{\mathsf{Bin}(k)}$ is bounded from above by $n\frac{2^{\lceil \log q \rceil} - q}{q}$, which is negligible for our choice of q and n, as required. □

By Remark 4.4 the NLWE problem using $\boldsymbol{\eta}_{\mathsf{Bin}(k)}$ remains as hard as LWE when $\mathbf{s} \in \mathbb{Z}_q^k$ is distributed as $\{\mathbf{r} \otimes (1, 2, 2^2, \ldots, 2^{\lceil \log q \rceil - 1})\}$ where $\mathbf{r}$ is uniform in $\mathbb{Z}_q^n$.

NLWE with Samples from a Discrete Gaussian. Next, we show that when the columns of $\mathbf{A}$ in LWE are sampled from a discrete Gaussian with a sufficiently large σ then the resulting problem is as hard as LWE. Let $\mathcal{D}_{\mathbb{Z}^k,\sigma}$ denote the Gaussian distribution with standard deviation σ *restricted* to $\mathbb{Z}^k$. We need the following two facts [4, 21]:

- There is an efficient randomized algorithm $\mathsf{TrapGen}(1^n, k, q)$ that given integers $n,\ q \geq 2$ and $k \geq 6n \log q$, outputs a matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times k}$ and a 'trapdoor' $\mathbf{T_M} \in \mathbb{Z}^{k \times k}$ such that $\mathbf{M}$ is $\mathrm{negl}(n)$-close to uniform.
- There is an efficient randomized algorithm $\mathsf{SampleD}(\mathbf{M}, \mathbf{T_M}, \mathbf{u}, \sigma)$ that given $\mathbf{u} \in \mathbb{Z}_q^n$, sufficiently large $\sigma = \Omega(\sqrt{n \log q})$, and the trapdoor $\mathbf{T_M}$ outputs a vector $\mathbf{e} \in \mathbb{Z}_q^k$ such that $\mathbf{Me} = \mathbf{u}$. Moreover, when $\mathbf{u}$ is uniform in $\mathbb{Z}_q^n$, the output of $\mathsf{SampleD}(\mathbf{M}, \mathbf{T_M}, \mathbf{u}, \sigma)$ is distributed as $\mathcal{D}_{\mathbb{Z}^k,\sigma}$.

Corollary 4.6. *Let $q = q(k)$ be an integer and $\sigma = \Omega(\sqrt{n \log q})$. Then the problem $(\mathbb{Z}_q, k, \chi, \mathcal{D}_{\mathbb{Z}^k,\sigma})$-NLWE is at least as hard as $(\mathbb{Z}_q, \lfloor k/(6 \log_2 q) \rfloor, \chi)$-LWE.*

Proof. By Theorem 4.3 it suffices to show that $\mathcal{D}_{\mathbb{Z}^k,\sigma}$ is coset sampleable. Algorithms $\mathsf{MatrixGen}$ and $\mathsf{SamplePre}$ are defined as follows:

- $\mathsf{MatrixGen}(1^k, n, q)$ runs $\mathsf{TrapGen}(1^n, k, q)$ and outputs $\mathbf{M}$ and $\mathbf{T} = (\mathbf{M}, \mathbf{T}_{\mathbf{M}})$.
- $\mathsf{SamplePre}(\mathbf{z} \in \mathbb{Z}_q^n,\ \mathbf{T})$ outputs $\mathsf{SampleD}(\mathbf{M}, \mathbf{T}_{\mathbf{M}}, \mathbf{z}, \sigma)$.

By construction, these algorithms show that $\mathcal{D}_{\mathbb{Z}^k,\sigma}$ is coset sampleable. □

NLWE with Samples in a Linear Subspace. Our last example which was also studied by Pietrzak [32] shows that when the columns of $\mathbf{A}$ in LWE are sampled uniformly from a linear subspace of sufficient dimension then the resulting problem is as hard as LWE.

Corollary 4.7. *Let $q = q(k)$ be an integer and let V be a linear subspace of $\mathbb{Z}_q^k$ of dimension at least n. Let $\boldsymbol{\eta}_V$ be the uniform distribution on V. Then the problem $(\mathbb{Z}_q, k, \chi, \boldsymbol{\eta}_V)$-NLWE is at least as hard as $(\mathbb{Z}_q, n, \chi)$-LWE.*

Proof Sketch. By Theorem 4.3 it suffices to show that $\boldsymbol{\eta}_V$ is coset sampleable and this follows by elementary linear algebra. □

5 An LWE-Based almost Key Homomorphic PRF

In this section, we construct a 1-almost key homomorphic PRF in the standard model based on the LWE assumption. Our new (almost key homomorphic) PRF has parameters comparable to those of [8]. Despite being *almost* key homomorphic, our PRF can still be used for the applications discussed in the introduction.

Construction. Let q, n, and m be integers such that $m = n\lceil \log q \rceil$. Recall the definition of the rounding function $\lceil \cdot \rfloor_p$ and the noise distribution $\overline{\Psi}_\alpha$ from Section 2, and the definition of $\boldsymbol{\eta}_{\mathsf{Bin}(m)}$ from Lemma 4.5. Let public parameters pp be a pair of matrices of the form $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{m \times m}$ where each row of $\mathbf{A}_0$ and $\mathbf{A}_1$ is sampled from $\boldsymbol{\eta}_{\mathsf{Bin}(m)}$ such that both matrices are full rank. The secret key $\mathbf{k}$ is a vector in $\mathbb{Z}_q^m$. Define $F_{\mathrm{LWE}} : \mathbb{Z}_q^m \times \{0,1\}^\ell \to \mathbb{Z}_p^m$ as follows:

$$F_{\mathrm{LWE}}(\mathbf{k}, x) = \left\lfloor \prod_{i=1}^{\ell} \mathbf{A}_{x_i} \cdot \mathbf{k} \right\rceil_p . \tag{5.1}$$

Theorem 5.1. *The function F_{LWE} is pseudorandom under the $(\mathbb{Z}_q, n, \overline{\Psi}_\alpha)$-LWE assumption for parameter choices such that $\alpha \cdot m^\ell \cdot p \leq 2^{-\omega(\log n)}$.*

The parameters α, m, ℓ, and p must be chosen carefully, as α determines the choice of the underlying lattice hardness assumption used for security. In the interest of space, we provide a proof overview of the pseudorandomness of F_{LWE} and leave the rest of the details to the full version.

Proof Overview of Theorem 5.1. In proving the pseudorandomness of F_{LWE}, we follow the outline of the standard GGM construction [22]. The proof uses $\ell + 1$ hybrid experiments, where in each experiment Expt_i for $i \in [\ell + 1]$, we successively ignore additional bits of the input in computing the product of $\mathbf{A}_{x_i}$'s while replacing this product with consistent random values. As usual, experiment Expt_1 corresponds to the case where all bits of x are ignored, and instead of the product of $\mathbf{A}_{x_i}$'s, the experiment returns random values. Experiment $\mathsf{Expt}_{\ell+1}$ honestly evaluates the PRF. Therefore, it suffices to show the indistinguishability of Expt_j and Expt_{j+1} for all $j \in [\ell+1]$, which is shown as follows.

Let $\mathbf{P} = \prod_{i=1}^{j-1} \mathbf{A}_{x_i}$. When the adversary queries its PRF oracle at a point $x \in \{0,1\}^\ell$, the resulting PRF evaluation given to the adversary in Expt_j and Expt_{j+1} looks like $\lceil \mathbf{P} \cdot \mathbf{r} \rfloor_p$ and $\left\lceil \mathbf{P}\mathbf{A}_{x_j} \cdot \mathbf{r} \right\rfloor_p$, where $\mathbf{r}$ is chosen uniformly in $\mathbb{Z}_q^m$ and is kept consistent across the adversary's queries using a lookup table indexed by the low order bits of the query x. An LWE challenge, either of the form $(\mathbf{A}, \mathbf{A}\mathbf{s} + \boldsymbol{\delta})$ or $(\mathbf{A}, \mathbf{r})$, cannot immediately be used to simulate the above evaluations. Instead, we move to an intermediate hybrid where the evaluations given to the adversary look like $\left\lceil \mathbf{P}\mathbf{A}_{x_j}\mathbf{s} + \mathbf{P}\boldsymbol{\delta} \right\rfloor_p = \left\lceil \mathbf{P}(\mathbf{A}_{x_j}\mathbf{s} + \boldsymbol{\delta}) \right\rfloor_p$, where $\mathbf{s}$ and $\boldsymbol{\delta}$ are kept consistent across the adversary's queries as was done previously. Now, it remains to show that

$$\text{(a)} \quad \left\lceil \mathbf{P}\mathbf{A}_{x_j}\mathbf{s} + \mathbf{P}\boldsymbol{\delta} \right\rfloor_p \approx \left\lceil \mathbf{P}\mathbf{A}_{x_j} \cdot \mathbf{r} \right\rfloor_p \qquad \text{and} \qquad \text{(b)} \quad \left\lceil \mathbf{P}(\mathbf{A}_{x_j}\mathbf{s} + \boldsymbol{\delta}) \right\rfloor_p \approx \lceil \mathbf{P} \cdot \mathbf{r} \rfloor_p,$$

Together these show that Expt_j is indistinguishable from Expt_{j+1}.

Statistical indistinguishability in (a) follows from a probabilistic argument by showing that for appropriate choices of parameters, the additive term $\mathbf{P}\boldsymbol{\delta}$ has no impact on the rounding. Although $\boldsymbol{\delta}$ is low-norm, if $\mathbf{P}$ were distributed uniformly, as is the usual case, this argument would fail. This explains why we must sample $\mathbf{A}_0$ and $\mathbf{A}_1$ using $\boldsymbol{\eta}_{\mathsf{Bin}(m)}$—it ensures that $\mathbf{P}$ is a low norm matrix so that $\mathbf{P}\boldsymbol{\delta}$ is a low norm vector.

The two terms in (b) are of the more familiar form of an LWE challenge, but with one important distinction. In such a challenge, the matrix $\mathbf{A}$ is low-norm, which is precisely modeled by the non-uniform learning with errors problem, introduced in Section 4. From Theorem 4.3, we can show computational indistinguishability in (b) under the *standard* LWE assumption. □

With a simple argument about the rounding of values from $\mathbb{Z}_q$ to $\mathbb{Z}_p$ (for which the details are left to the full version), for every $\mathbf{k}_1, \mathbf{k}_2 \in \mathbb{Z}_q^m$ and every $x \in \{0,1\}^\ell$, there exists an $\mathbf{e} \in [-1, 1]^m$ such that $F_{\text{LWE}}(\mathbf{k}_1, x) + F_{\text{LWE}}(\mathbf{k}_2, x) = F_{\text{LWE}}(\mathbf{k}_1 + \mathbf{k}_2, x) + \mathbf{e}$. Thus, we can state the following theorem.

Theorem 5.2. *The tuple $(F_{\text{LWE}}, +)$ is a 1-almost key homomorphic PRF, where $+$ is addition over $\mathbb{Z}_q^m$.*

6 Applications of (Almost) Key Homomorphic PRFs

In this section, we construct one-round distributed PRFs [28] and symmetric proxy re-encryption schemes from γ-almost key homomorphic PRFs.

6.1 Distributed PRFs

Definition. To define distributed PRFs we follow the exposition of Naor, Pinkas, and Reingold [28]. The model comprises of N servers $S_1, \ldots, S_N$ and a client C that is connected to at least t servers.

A distributed PRF is a tuple of algorithms $\Pi = (\mathsf{Setup}, \mathsf{Share}, \mathsf{F}, \mathsf{G}, \mathsf{f})$ with the following properties. Algorithm Setup takes the security parameter λ and outputs public parameters pp. The *key-sharing* algorithm $\mathsf{Share} : \mathcal{K} \to \mathcal{K}^N$ takes as input a master secret key $k_0 \in \mathcal{K}$ and outputs a tuple $(k_1, \ldots, k_N) \in \mathcal{K}^N$, where $k_1, \ldots, k_N$ represent the key-shares of k_0 from a (t, N)-threshold secret sharing scheme. The *partial evaluation* function $\mathsf{F}\colon \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ computes a partial evaluation of the function f when given a key-share and an input point. The *combine* algorithm $G\colon 2^{[N]} \times \mathcal{Y}^t \to \mathcal{Y}$ takes as input a subset $W \subset [N]$ of size t and the t partial evaluations on key-shares in the set W and outputs a value in $\mathcal{Y}$. The *evaluation* function $\mathsf{f}\colon \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ maps a key and an input to the space of outputs.

The distributed PRF is initialized by a trusted third party that runs $\mathsf{Setup}(1^\lambda)$ to obtain the public parameters pp, samples a master secret key k_0 uniformly from $\mathcal{K}$, and runs $\mathsf{Share}(k_0)$ to obtain a tuple $(k_1, \ldots, k_n)$. The key-share k_i is distributed as the secret key for each server S_i along with public parameters pp. A client C that wants to compute the evaluation function f under key k_0 on input x sends x to t servers $S_{i_1}, \ldots, S_{i_t}$. Each server S_i responds to the client with $\mathsf{F}(k_i, x)$. Then, the client *locally* computes $\mathsf{f}(k_0, x)$ by computing $\mathsf{G}\left(W, \mathsf{F}(k_{i_1}, x), \ldots, \mathsf{F}(k_{i_t}, x)\right)$.

Consistency. Let pp be the output of $\mathsf{Setup}(1^\lambda)$, k_0 be sampled uniformly from $\mathcal{K}$, and $(k_1, \ldots, k_N)$ be the key-shares output by $\mathsf{Share}(k_0)$. For every subset $W = \{i_1, \ldots, i_t\} \subset [N]$ of size t, and for every input x, a distributed PRF Π is *consistent* if $\mathsf{f}(k_0, x) = \mathsf{G}\left(W, \mathsf{F}(k_{i_1}, x), \ldots, \mathsf{F}(k_{i_t}, x)\right)$.

Pseudorandomness. Intuitively, the evaluation function f should remain pseudorandom even when the adversary is given $t-1$ key shares $k_{i_1}, \ldots, k_{i_{t-1}}$ for indices $\{i_1, \ldots, i_{t-1}\}$ of its choice. The adversary is also given an oracle $\mathcal{O}$ that performs arbitrary partial evaluations: it takes (i, x) as input and returns $\mathsf{F}(k_i, x)$. The adversary should be unable to distinguish the function from random at points x where it did not query the oracle $\mathcal{O}$. We formalize this intuition by providing a concrete experiment-based security definition in the full version.

Distributed PRFs from Key Homomorphic PRFs. Let $F : \mathbb{F} \times \mathcal{X} \to \mathbb{F}$ be a key homomorphic PRF where $\mathbb{F}$ is a field. We consider a (t, N)-threshold secret sharing scheme [35] over a secret k_0 in $\mathbb{F}$, which constructs key-shares by sampling a uniformly random polynomial $p(z) \in \mathbb{F}[Z]$ of degree $t-1$ such that $p(0) = k_0$, and the remaining coefficients are sampled uniformly at random from $\mathbb{F}$. We then define share $k_i = p(i)$ for $i \in [1, N]$. For secret shares constructed in this manner, it holds that for any $W = \{i_1, \ldots, i_t\} \subset [N]$ of size t we have that $k_0 = p(0) = \sum_{i \in W} \Lambda_{i,W} \cdot k_i$, where $\Lambda_{i,W} \in \mathbb{F}$ are the Lagrange coefficients that depend only on W. We construct a distributed PRF scheme $\Pi_{\mathsf{dPRF}} = (\mathsf{Setup}, \mathsf{Share}, \mathsf{F}, \mathsf{G}, \mathsf{f})$ as follows.

- $\mathsf{Setup}(1^\lambda)$ outputs public parameters pp used by the key homomorphic PRF F.
- $\mathsf{Share}(k_0)$ samples a uniformly random polynomial $p(z)$ of degree $t-1$ such that $p(0) = k_0$ and outputs $(p(1), \ldots, p(N))$.

- $\diamond$ $\mathsf{F}(k_i, x)$ returns the output of the key homomorphic PRF $F(k_i, x)$.
- $\diamond$ $\mathsf{G}(W, y_1, \ldots, y_t)$ computes and returns $\sum_{i \in W} \Lambda_{i,W} \cdot y_i$.
- $\diamond$ $\mathsf{f}(k_0, x)$ returns the output of $F(k_0, x)$.

Let pp be the output of $\mathsf{Setup}(1^\lambda)$, k_0 be chosen uniformly from $\mathbb{F}$, $(k_1, \ldots, k_n)$ be the output of $\mathsf{Share}(k_0)$, $W = \{i_1, \ldots, i_t\} \subset [N]$, and $x \in \mathcal{X}$. The combine algorithm computes $\mathsf{G}\left(W, \mathsf{F}(k_{i_1}, x), \ldots, \mathsf{F}(k_{i_t}, x)\right) = \sum_{i \in W} \Lambda_{i,W} \cdot F(k_i, x)$. By the key homomorphism property of F, this quantity is equal to $\sum_{i \in W} F\left(\Lambda_{i,W} \cdot k_i, x\right) = \mathsf{F}(k_0, x)$ as required. If F is a key homomorphic pseudorandom function, then the following theorem shows that Π_{dPRF} is a secure distributed PRF. The complete proof is given in the full version.

Theorem 6.1. *If F is a key homomorphic PRF, then Π_{dPRF} is a secure distributed PRF.*

Constructing dPRFs from almost Key Homomorphic PRFs. The construction described above for a distributed PRF can be adapted to PRFs that are γ-almost key homomorphic with output space $\mathbb{Z}_p$ for some prime p. Unfortunately, when Lagrange coefficients are interpreted as elements in $\mathbb{Z}_p$, they can be arbitrarily large which breaks correctness of the combine algorithm. To overcome this difficulty, we use the technique of "clearing the denominator" [36, 1]. Note that for every Lagrange coefficient $\Lambda_{i,W}$, it holds that $N! \cdot \Lambda_{i,W} \in \mathbb{Z}$, and so the combine algorithm now uses $N! \cdot \Lambda_{i,W} \in \mathbb{Z}$ to reconstruct the output of the PRF. The complete details of the construction and its security proof are included in the full version.

6.2 Symmetric Proxy Re-encryption

Another natural application of key homomorphic PRFs is in constructing symmetric proxy re-encryption schemes. In a proxy re-encryption scheme, a proxy is given re-encryption information that enables the proxy to translate an encryption of any message from one key to an encryption of the same message under another key without revealing the underlying message. A symmetric proxy re-encryption scheme is a tuple of algorithms $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{ReKeyGen}, \mathsf{Enc}, \mathsf{ReEnc}, \mathsf{Dec})$ with the following properties.

- $\diamond$ $\mathsf{Setup}(1^\lambda) \rightarrow \mathsf{pp}$. On input the security parameter λ, the setup algorithm Setup outputs the public parameters pp used for the scheme.
- $\diamond$ $\mathsf{KeyGen}(1^\lambda) \rightarrow \mathsf{sk}$. On input the security parameter λ, the key generation algorithm KeyGen outputs a secret key sk.
- $\diamond$ $\mathsf{ReKeyGen}(\mathsf{sk}_1, \mathsf{sk}_2) \rightarrow \mathsf{rk}_{1,2}$. On input two secret keys sk_1 and sk_2, the re-encryption key generation algorithm ReKeyGen outputs a bidirectional re-encryption key $\mathsf{rk}_{1,2}$.
- $\diamond$ $\mathsf{Enc}(\mathsf{sk}, m) \rightarrow C$. On input a secret key sk and message m, the encryption algorithm Enc outputs a ciphertext C.
- $\diamond$ $\mathsf{ReEnc}(\mathsf{rk}_{1,2}, C_1) \rightarrow C_2$. On input a re-encryption key $\mathsf{rk}_{1,2}$ and a ciphertext C_1, the re-encryption algorithm ReEnc outputs a second ciphertext C_2 or $\perp$.
- $\diamond$ $\mathsf{Dec}(\mathsf{sk}, C) \rightarrow m$. On input a secret key sk and a ciphertext C, the decryption algorithm Dec outputs a message m or the error symbol $\perp$.

Correctness. A symmetric proxy re-encryption scheme Π is *T-time correct* if, under public parameters $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$ and for all $m_i \in \mathcal{M}$, $\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{sk}, m)) = m$, and for any sequence of secret keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_T$ output by $\mathsf{KeyGen}(1^\lambda)$ and re-encryption keys $\mathsf{rk}_{i,i+1}$ output by $\mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{sk}_{i+1})$ for $i \in [1, T-1]$, for all messages $m \in \mathcal{M}$ and all ciphertexts C output by $\mathsf{Enc}(\mathsf{sk}_1, m)$, it holds that $\mathsf{Dec}(\mathsf{sk}_T, \mathsf{ReEnc}(\mathsf{rk}_{T-1,T}, \cdots \mathsf{ReEnc}(\mathsf{rk}_{1,2}, C) \cdots)) = m$.

Security. The security model defines the notion of semantic security for *symmetric* proxy re-encryption. We adapt the public-key model of Canetti and Hohenberger [15] to the symmetric-key settings by providing access to an additional encryption oracle. The model is described in detail in the full version.

Symmetric Proxy Re-encryption from Key Homomorphic PRFs. We show how to achieve a symmetric proxy re-encryption scheme with T-time correctness for all $T > 1$ under our security model using key homomorphic PRFs. Let $F : \mathcal{K} \times \mathcal{X} \rightarrow \mathcal{Y}$ be a key homomorphic PRF with superpolynomial input set size ($|\mathcal{X}| = \omega(\mathrm{poly}(\lambda))$). Let $\Pi_{\mathsf{proxy}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{ReKeyGen}, \mathsf{Enc}, \mathsf{ReEnc}, \mathsf{Dec})$ be defined as follows.

- $\mathsf{Setup}(1^\lambda)$ samples and outputs the public parameters pp used by F.
- $\mathsf{KeyGen}(1^\lambda)$ outputs a uniform secret key sk from the key space $\mathcal{K}$.
- $\mathsf{ReKeyGen}(\mathsf{sk}_1, \mathsf{sk}_2)$ returns $\mathsf{rk}_{1,2} = \mathsf{sk}_2 - \mathsf{sk}_1$.
- $\mathsf{Enc}(\mathsf{sk}, m)$ chooses a random $r \leftarrow \mathcal{X}$ and outputs $(r, m + F(\mathsf{sk}, r))$.
- $\mathsf{ReEnc}(\mathsf{rk}_{1,2}, (r, C))$ outputs $(r, C + F(\mathsf{rk}_{1,2}, r))$.
- $\mathsf{Dec}(\mathsf{sk}, (r, C))$ outputs $C - F(\mathsf{sk}, r)$.

If F is a key homomorphic pseudorandom function with superpolynomial input set size, then the following theorem shows that Π_{proxy} is a secure symmetric proxy re-encryption scheme. The complete proof is given in the full version.

Theorem 6.2. *If F is a key homomorphic PRF, then Π_{proxy} is a secure symmetric proxy re-encryption scheme.*

Using almost Key Homomorphic PRFs. In the full version, we show how to construct a symmetric proxy re-encryption scheme with T-time correctness (where T must be an additional input to Setup) from almost key homomorphic PRFs. The proof of correctness is straightforward and the proof of security remains the same.

7 Conclusions and Open Problems

We explored the concept of key-homomorphic PRFs and discussed its application to key rotation, one-round distributed PRFs, and symmetric-key proxy re-encryption. Our construction of lattice-based key-homomorphic PRFs in the standard model relies on a non-uniform variant of the learning with errors assumption that we show is equivalent to the standard LWE assumption.

We leave as an open problem the question of constructing key-homomorphic PRFs from other standard assumptions and in particular constructions using bilinear maps. Another interesting area of research is to construct key-homomorphic PRFs whose performance is comparable to real-world block ciphers such as AES.

It would be interesting to improve the tightness of the analysis in our lattice-based PRF, perhaps using techniques from [3]. Another useful improvement would be to reduce the hardness of worst-case lattice problems directly to our non-uniform LWE variant to improve its hardness parameters.

Acknowledgments. We thank Zvika Brakerski, Daniele Miccancio, and Chris Peikert for helpful comments about this work. We also thank Gregory Roth for suggesting the application to re-keying in the cloud. This work was supported by NSF, the DARPA PROCEED program, an AFO SR MURI award, a grant from ONR, and by Samsung. Opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA. Distrib. Statement "A": Approved for Public Release, Distribution Unlimited.

References

[1] Agrawal, S., Boyen, X., Vaikuntanathan, V., Voulgaris, P., Wee, H.: Functional encryption for threshold functions (or fuzzy IBE) from lattices. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 280–297. Springer, Heidelberg (2012)

[2] Agrawal, S., Freeman, D.M., Vaikuntanathan, V.: Functional encryption for inner product predicates from learning with errors. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 21–40. Springer, Heidelberg (2011)

[3] Alwen, J., Krenn, S., Pietrzak, K., Wichs, D.: Learning with rounding, revisited: New reduction, properties and applications. IACR Cryptology ePrint Archive, 2013:98 (2013)

[4] Alwen, J., Peikert, C.: Generating shorter bases for hard random lattices. In: STACS, pp. 75–86 (2009)

[5] Applebaum, B., Cash, D., Peikert, C., Sahai, A.: Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 595–618. Springer, Heidelberg (2009)

[6] Ateniese, G., Benson, K., Hohenberger, S.: Key-private proxy re-encryption. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 279–294. Springer, Heidelberg (2009)

[7] Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Trans. Inf. Syst. Secur. 9(1), 1–30 (2006)

[8] Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 719–737. Springer, Heidelberg (2012)

[9] Bellare, M., Cash, D.: Pseudorandom functions and permutations provably secure against related-key attacks. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 666–684. Springer, Heidelberg (2010)

[10] Bellare, M., Kohno, T.: A theoretical treatment of related-key attacks: Rka-prps, rka-prfs, and applications. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 491–506. Springer, Heidelberg (2003)

[11] Bellare, M., Paterson, K.G., Thomson, S.: RKA security beyond the linear barrier: IBE, encryption and signatures. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 331–348. Springer, Heidelberg (2012)
[12] Blaze, M., Bleumer, G., Strauss, M.J.: Divertible protocols and atomic proxy cryptography. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer, Heidelberg (1998)
[13] Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: STOC, pp. 575–584 (2013)
[14] Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: FOCS, pp. 97–106 (2011)
[15] Canetti, R., Hohenberger, S.: Chosen-ciphertext secure proxy re-encryption. In: ACM Conference on Computer and Communications Security, pp. 185–194 (2007)
[16] Cook, D.L., Keromytis, A.D.: Conversion and proxy functions for symmetric key ciphers. In: ITCC (1), pp. 662–667 (2005)
[17] Coviello, A.: Open letter to rsa customers (2012), `http://www.rsa.com/node.aspx?id=3872`
[18] D'Arco, P., Stinson, D.R.: On unconditionally secure robust distributed key distribution centers. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 346–363. Springer, Heidelberg (2002)
[19] Dodis, Y.: Efficient construction of (Distributed) verifiable random functions. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 1–17. Springer, Heidelberg (2002)
[20] Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices and applications. Cryptology ePrint Archive, Report 2012/610 (2012)
[21] Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC 2008, pp. 197–206. ACM (2008)
[22] Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. J. ACM 34(4), 792–807 (1986)
[23] Hofheinz, D., Kiltz, E.: Secure hybrid encryption from weakened key encapsulation. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 553–571. Springer, Heidelberg (2007)
[24] Krawczyk, H.: The order of encryption and authentication for protecting communications (or: How secure is SSL?). In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 310–331. Springer, Heidelberg (2001)
[25] Lewko, A., Waters, B.: Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In: ACM CCS, pp. 112–120 (2009)
[26] Libert, B., Vergnaud, D.: Unidirectional chosen-ciphertext secure proxy re-encryption. IEEE Transactions on Information Theory 57(3), 1786–1802 (2011)
[27] Micciancio, D., Peikert, C.: Hardness of SIS and LWE with small parameters. IACR Cryptology ePrint Archive (2013)
[28] Naor, M., Pinkas, B., Reingold, O.: Distributed pseudo-random functions and kDCs. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 327–346. Springer, Heidelberg (1999)
[29] Naor, M., Reingold, O.: Number-theoretic constructions of efficient pseudo-random functions. In: FOCS 1997, pp. 458–467 (1997)
[30] Nielsen, J.B.: A threshold pseudorandom function construction and its applications. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 401–416. Springer, Heidelberg (2002)
[31] Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: 41st Annual ACM Symposium on Theory of Computing (STOC 2009), pp. 333–342. ACM (2009)

[32] Pietrzak, K.: Subspace LWE. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 548–563. Springer, Heidelberg (2012)
[33] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC 2005, pp. 84–93. ACM (2005)
[34] Shacham, H.: A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants. IACR Cryptology ePrint Archive, 2007:74 (2007)
[35] Shamir, A.: How to share a secret. Communications of the ACM 22(11), 612–613 (1979)
[36] Shoup, V.: Practical threshold signatures. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 207–220. Springer, Heidelberg (2000)
[37] Syalim, A., Nishide, T., Sakurai, K.: Realizing proxy re-encryption in the symmetric world. In: Abd Manaf, A., Zeki, A., Zamani, M., Chuprat, S., El-Qawasmeh, E. (eds.) ICIEIS 2011, Part I. CCIS, vol. 251, pp. 259–274. Springer, Heidelberg (2011)

On the Security of the TLS Protocol: A Systematic Analysis

Hugo Krawczyk[1], Kenneth G. Paterson[2,⋆], and Hoeteck Wee[3,⋆⋆]

[1] IBM Research
[2] Royal Holloway, University of London
[3] George Washington University

Abstract. TLS is the most widely-used cryptographic protocol on the Internet. It comprises the TLS Handshake Protocol, responsible for authentication and key establishment, and the TLS Record Protocol, which takes care of subsequent use of those keys to protect bulk data. In this paper, we present the most complete analysis to date of the TLS Handshake protocol and its application to data encryption (in the Record Protocol). We show how to extract a key-encapsulation mechanism (KEM) from the TLS Handshake Protocol, and how the security of the entire TLS protocol follows from security properties of this KEM when composed with a secure authenticated encryption scheme in the Record Protocol. The security notion we achieve is a variant of the ACCE notion recently introduced by Jager et al. (Crypto '12). Our approach enables us to analyse multiple different key establishment methods in a modular fashion, including the *first proof* of the most common deployment mode that is based on RSA PKCS #1v1.5 encryption, as well as Diffie-Hellman modes. Our results can be applied to settings where mutual authentication is provided and to the more common situation where only server authentication is applied.

1 Introduction

TLS is the mostly widely used cryptographic protocol for secure communications on the Internet. The main purpose of TLS is to provide end-to-end security against an active, man-in-the-middle attacker. Originally deployed (as SSL) in web browsers for `https` connections, TLS is now used as a general purpose provider of secure communications to all kinds of applications: e-commerce transactions, virtual private networks (VPN), Android and iOS mobile apps [20, 21], as well as related procotols like DTLS [31, 37]. In short, TLS is one of the most important real-world deployments of cryptography.

TLS in a Nutshell. We begin with an informal high-level overview of the TLS protocol; a more detailed treatment is given in Section 2. The TLS protocol is executed between a client and a server. It has two main constituents: the Handshake Protocol, which is responsible for key establishment and authentication; and the Record Protocol, which provides a secure channel for handling the delivery of data. The Handshake Protocol establishes the application keys, which are in turn used to encrypt application

⋆ Supported by EPSRC Leadership Fellowship EP/H005455/1.
⋆⋆ Supported by NSF CAREER Award CNS-1237429.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 429–448, 2013.

data in the Record Protocol. The TLS specification offers multiple options for key establishment mechanisms in the Handshake Protocol and for symmetric key encryption schemes in the Record Protocol. The most common configuration, to which we refer as TLS-RSA, relies on RSA PKCS #1v1.5 encryption in the Handshake Protocol. Other configurations include TLS-DH and TLS-DHE, which rely on Diffie-Hellman key exchange (the first uses a static server's DH key and an ephemeral client's key while in the latter both parties contribute ephemeral DH keys). All these configurations provide server authentication, with optional client authentication in settings where clients possess public keys as well.

Prior Work on TLS. In view of its importance, TLS has long been the subject of intense research analysis, including, in chronological order, [41, 12, 36, 28, 25, 40, 16, 27, 5, 32, 22, 6, 33, 18, 35, 9, 2, 24, 30, 19, 13, 3, 10, 4]. The main, twin thrusts of this research have been to establish to what extent the TLS Handshake Protocol and the TLS Record Protocol are secure, for the respective tasks of key establishment and authentication and for providing a secure channel for delivery of data.

We now have a fairly complete understanding of the underlying cryptography for the Record Protocol, as studied in the works of Krawczyk [28] as well as Paterson, Ristenpart and Shrimpton [35]. These works demonstrated that, when carefully implemented to avoid timing and other attacks like those in [40, 16, 3], the stream-cipher and CBC encryption modes in the TLS Record Layer achieve the security notion of authenticated encryption; in fact, [35] puts forth and achieves a strengthening there-of, known as *stateful, length-hiding authenticated encryption* (sLHAE).

On the other hand, a complete analysis of the TLS Handshake Protocol remains elusive. A main obstacle is that the design of TLS violates the basic cryptographic principles of key indistinguishability and separation of key exchange and secure channels. This arises because the TLS Record Protocol overlaps with the TLS Handshake Protocol, and the application key is used to encrypt the last two messages of the Handshake Protocol (known as the `Finished` messages). As such, the TLS Handshake Protocol is deemed insecure by the existing security models for key exchange, initiated in the work of Bellare and Rogaway [8].

Several prior works [33, 25] circumvented this issue by analyzing *variants* of the TLS protocol (e.g. with a different message ordering, unencrypted `Finished` messages, or RSA-OAEP encryption). In particular, Morrissey, Smart and Warinschi [33] analyze the "Truncated TLS Handshake Protocol", where the `Finished` messages are not encrypted by the application key. An important feature of [33] is the modularity of the approach. This conceptually simplifies the protocol and the security proofs, and points the way forward for subsequent analysis. However, the end result applies to truncated TLS and not to the real protocol. In addition, Morrissey et al. [33] model TLS-RSA under the assumption that RSA encryption is replaced with CCA-secure encryption which is provably false for RSA PKCS #1v1.5 encryption as used in TLS-RSA. The modularity theme from [33] is developed further in recent work by Brzuska et al. [13], who analyze the TLS protocol using a game-based framework that is designed to enable compositional results to be be proved, but their analysis of TLS-RSA assumes IND-CCA security for the RSA encryption, which, again, is known not to hold. Thus,

unfortunately, given the high sensitivity of key exchange protocols in general (and TLS in particular) to small details, these results tell us little about TLS as used in practice.

In recent work, Jager et al. [24] put forth a new security notion — Authenticated and Confidential Channel Establishment (ACCE) security — which captures the desired security guarantees when the TLS Handshake and Record Protocols are used in tandem. (This circumvents the barrier pertaining to the separation of key exchange and secure channels.) In addition, they showed that the cryptographic core of the TLS-DHE protocol when both server and client authentication are applied satisfies ACCE security. Informally, this means the TLS Record Protocol when used with TLS-DHE as the Handshake Protocol constitutes a secure channel and guarantees authentication and privacy for data delivery between the server and the client. While this work constitutes a significant step forward in terms of realistic modeling and analysis of TLS, the TLS-DHE protocol is (currently) seldom used in practice, and client-side authentication via signatures is very rarely done.

Additional literature on analyzing the TLS Handshake Protocol include works on symbolic models, e.g. [36, 22, 9] and on security analysis of a TLS implementation via type-checking [10]. Works on simulation-based definitions and designs for key agreement and secure channel protocols include [39, 14, 15].

TLS-RSA. As noted earlier, the most commonly deployed mode of TLS, namely TLS-RSA, uses RSA PKCS #1v1.5 encryption [26]. In 1998, Bleichenbacher discovered a devastating man-in-the-middle attack on SSL, the predecessor of TLS. Specifically, Bleichenbacher presented a chosen-ciphertext attack on RSA PKCS #1v1.5 encryption [12], which in turn allows a man-in-the-middle adversary against SSL to recover the pre-master secret and thence the application keys. In fact, the attack only requires a ciphertext validity oracle. TLS, the successor to SSL, incorporates an *ad hoc fix* to thwart Bleichenbacher's attack: decryption failures are hidden from the adversary, including via some defences against timing attacks, thereby removing access to the ciphertext validity oracle.

For over a decade, the TLS Handshake Protocol (and in particular TLS-RSA) has largely resisted attacks; however, that in itself does not rule out the possibility of an attack being discovered in future. The folklore belief is that TLS-RSA is secure if we replace RSA PKCS #1v1.5 with RSA-OAEP or any other CCA-secure encryption scheme; unfortunately, only RSA PKCS #1v1.5 is standardised in TLS and used in practice. This begs the question:

Is TLS-RSA with RSA PKCS #1v1.5 encryption ACCE secure?

A partial answer to the above question was provided in the work of Jonsson and Kaliski Jr. [25]: they showed that RSA PKCS #1v1.5 encryption when augmented with the *unencrypted* TLS client `Finished` message is CCA-secure. However, their analysis was not extended to either the TLS Handshake Protocol or the full TLS protocol; furthermore, in TLS the client `Finished` message is actually encrypted with the application key. We stress that RSA PKCS #1v1.5 encryption when augmented with the *encrypted* TLS client `Finished` message is not even a CPA-secure key-encapsulation mechanism (KEM), for the same reason that the TLS Handshake Protocol violates key indistinguishability.

Proving Security of TLS-RSA and Beyond. We provide an affirmative answer to the above question, namely, we provide the first proof of security for the *unmodified* TLS-RSA protocol with RSA PKCS #1v1.5 encryption in the commonly deployed setting of server-only authentication. More generally, we provide a systematic and modular analysis of the different modes of TLS, which include TLS-RSA, TLS-DH and TLS-DHE, in both the common setting of server-only authentication as well as with combined client and server authentication. We also validate the folklore belief that TLS Handshake with RSA replaced with any CCA-secure public-key encryption scheme (e.g., RSA-OAEP) is secure. We refer to such an instantiation as TLS-CCA. Following Jager et al. [24], we focus on the *cryptographic core* of TLS (see the full version [29] for a discussion of what we omit). We concentrate on achieving ACCE security with appropriate modifications to handle server-only authentication (in which case we speak of SACCE security). We next present an overview of our framework, summarized in Figure 1.

1.1 Systematic Analysis of All TLS Handshake Modes

Our Framework. We build an abstraction of the TLS Handshake Protocol via a generic representation using a *key-encapsulation mechanism (KEM)* (see Figure 2). Each of the TLS modes is then fully defined via a specific instantiation of the KEM. The goal is to find sufficient conditions on the KEM so that any instantiation satisfying these conditions immediately leads to a secure protocol in the sense of ACCE security (as discussed above). This approach has its roots in the work of Jonsson and Kaliski [25] that studied the underlying KEM in TLS-RSA.

We formalize this statement using the existing notion of constrained CCA (CCCA) security, introduced by Hofheinz and Kiltz [23] in the context of hybrid encryption. In the CCCA security game, the adversary is provided with a "constrained decryption oracle" that takes as input a pair (C, T) where C is a ciphertext and T is some auxiliary information; the oracle returns the decryption K of C if C is different from the challenge ciphertext and (K, T) satisfies some specified predicate, and $\perp$ otherwise. In particular, if the oracle returns $\perp$, the adversary does not learn whether it is because K is $\perp$ or because (K, T) fails to satisfy the predicate. In our framework, we consider CCCA security where T is an encrypted TLS client `Finished` message, and the predicate enforces validity of T. Now, if the constrained decryption oracle returns $\perp$ on query (C, T), the adversary does not learn whether it is because C is an invalid ciphertext or because T is an invalid `Finished` message – this precisely captures the intention of the TLS fix for thwarting Bleichenbacher's attack! We note that the challenge ciphertext in the CCCA security experiment is not accompanied by the corresponding `Finished` message; this asymmetry between the challenge ciphertext and the oracle queries allows us to bypass the key indistinguishability barrier in TLS.

ACCE Security from CCCA Security. Our first result says that if the key encapsulation mechanism in the TLS Handshake Protocol satisfies CCCA security and the encryption scheme used in the TLS Record Protocol is sLHAE-secure, then TLS is ACCE secure, in the server-only authentication setting. We stress that this result is in the standard model. Importantly, the CCCA security game is conceptually and technically

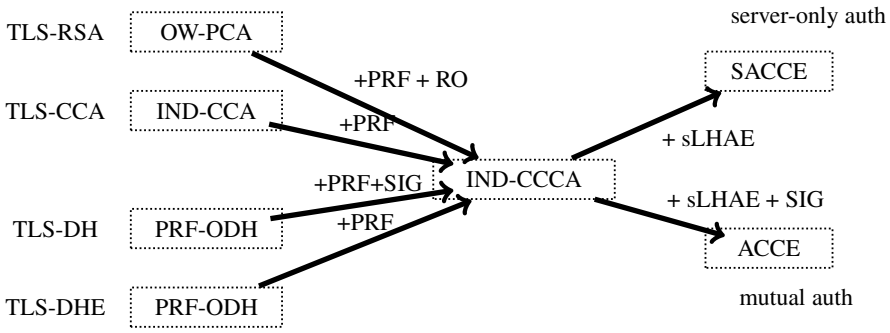


Fig. 1. Summary of our results

much simpler to analyze than the whole TLS protocol, as we do not have to worry about multiple sessions, nonces, or the multiple message flows in the full protocol.

To establish ACCE security, we need to achieve security against a (concurrent) man-in-the-middle adversary communicating with multiple honest clients and multiple honest servers. Roughly speaking, we will rely on the constrained decryption oracle in CCCA security to simulate the honest servers. The main technical difficulty in establishing this result arises when a man-in-the-middle adversary plays a relaying strategy between an honest server and client and then mauls the client's encrypted `Finished` message. Here, we cannot rely on the constrained decryption oracle to simulate the honest server's response because the adversary is using the challenge ciphertext. Moreover, we cannot immediately appeal to the non-malleability of the sLHAE-secure scheme used to encrypt the `Finished` message since the protocol messages leak information about the application key. To solve this problem, we exploit the fact that the CCCA security game provides us with a real-or-random key K^*, which we may use to decrypt and verify the client's encrypted `Finished` message for this specific adversarial strategy. We stress that this techical difficulty goes away if the client's `Finished` message is unencrypted, because the prior transcript uniquely determines an accepting `Finished` message.

CCCA Security in the TLS Handshake. Our second set of results says that the key encapsulation mechanisms underlying the TLS-RSA, TLS-CCA, TLS-DH, and TLS-DHE variants of the TLS Handshake Protocol all satisfy CCCA security. Combined with our first result, this yields ACCE security of TLS-RSA, TLS-CCA, TLS-DH and TLS-DHE (see Figure 1).

ACCE Security of TLS with Mutual Authentication. We extend the above results, developed for the case of server-only authentication, to the case of mutual authentication, namely, when the client authenticates itself via a digital signature. We show that also in this setting, CCCA security of the underlying KEM implies ACCE security with both server and client authentication. The extension is relatively straightforward (a positive feature!) requiring minor changes to the server-authentication-only proofs of server authentication and channel security, and the addition of a client authentication

proof. The resultant analysis is generic and independent of the different underlying KEM instantiations, thus it directly applies to TLS-RSA, TLS-CCA, TLS-DH and TLS-DHE (demonstrating the power of our modular analysis).

1.2 Summary of Results

As a result of the above methodology we obtain proofs of ACCE security for the TLS handshake protocol for *all* of the above TLS modes, both in the common setting of server-only authentication as well as with mutual authentication. These results are depicted in Figure 1 and are enumerated here with the assumptions used in each case. In all cases we assume a secure PRF and the TLS Record Protocol encryption implemented with an sLHAE encryption scheme. For the case of ACCE security with mutual authentication a secure client signature is also assumed. Certificates for both servers and clients are assumed to be provided by a minimally trusted CA that faithfully checks identities before issuing certificates. No other checks from the CA (such as proofs of possession, uniqueness of public keys, etc.) are assumed.

TLS-RSA. We obtain the first proof of security of TLS-RSA as deployed in practice, with RSA PKCS #1v1.5 and server-only authentication, in the random oracle model and under the assumption that RSA PKCS #1v1.5 is OW-PCA secure. The latter assumption, formalized in Section 5, states that inverting the encryption function is hard even given an oracle that on input a plaintext-ciphertext pair (K, ψ) checks whether the decryption of ψ equals K (for $K \neq \perp$). The OW-PCA security of RSA PKCS #1v1.5 can be proven under an RSA-like assumption, known as "partial-domain RSA with decision oracle", introduced by Jonsson and Kaliski in [25] and which we present in Section 5.2. We refer to [25] for a discussion on why this assumption is reasonable for typical parameters used in TLS; to the best of our knowledge no weakness in this assumption has been discovered since its introduction in [25]. When clients authenticate in TLS-RSA using digital signatures then full ACCE (i.e. with mutual authentication) is proven assuming a secure signature scheme. We stress that TLS-RSA is the only TLS mode whose proof is in the random oracle model; we prove all other modes in the standard model.

TLS-CCA. We prove that when instantiated with a CCA-secure public-key encryption scheme (instead of RSA PKCS #1v1.5), TLS is ACCE secure in the standard model. While no such schemes are currently standardised for TLS, this result confirms the intuition that IND-CCA security is the "right" target for the public key encryption scheme used in TLS. It also means that, should the current RSA-based encryption scheme used in TLS ever be replaced by a CCA-secure one, for example RSA-OAEP, then our analysis will immediately provide strong security guarantees for the protocol.

TLS-DH and TLS-DHE. We prove ACCE security (with and without client authentication) of TLS-DH in the standard model under the PRF-ODH assumption introduced in [24].[1] The PRF-ODH assumption rules out some potential related-key attacks on the

[1] The assumption is a variant of the ODH assumption from [1] where the oracle is implemented via a PRF rather than by a hash function. In the proof of TLS-DH we require security against multiple oracle queries while for TLS-DHE a single query suffices, as was the case in [24].

Kdf function that would render the protocol insecure. In the full version [29] we show this assumption to be *provably necessary* for the security of TLS-DH, showing attacks on the protocol with PRFs for which the assumption does not hold. We note that we can also prove TLS-DH in the random oracle model under the Strong DH assumption. Finally, we obtain security for TLS-DHE as a corollary of our results for TLS-DH security, under the PRF-ODH assumption as well as secure signatures for servers (and clients in the case of mutual authentication). Note that our results for TLS-DHE do not encompass forward security, but this is guaranteed by the results of [24].

2 The TLS Handshake Protocol with Server-Only Authentication

In this section, we present our model of the TLS Handshake Protocol when no client authentication takes place. As noted in the introduction, this includes TLS-RSA, the most common usage of the TLS protocol. The parties to the protocol are a client C and a server S. Each maintains an internal state variable ST and $\Lambda \in \{\emptyset, \mathtt{acc}, \mathtt{rej}\}$. The protocol makes use of a number of cryptographic components: a key derivation function (KDF) Kdf, a pseudorandom function PRF, a stateful authenticated encryption with associated data (AEAD) scheme $\mathsf{stE} = (\mathsf{stE.Gen}, \mathsf{stE.Init}, \mathsf{stE.Enc}, \mathsf{stE.Dec})$, and a KEM $(\mathsf{KeyGen}, \mathsf{F}_C, \mathsf{F}_S)$. The protocol is shown schematically in Figure 2. We also describe below how the keys established by this protocol are subsequently used by the TLS Record Protocol.

The model is derived from the current TLS specification [17], and we believe that our model captures the *cryptographic core* of TLS. It has a comparable level of accuracy to the model of TLS-DHE used in [24]. We highlight several salient properties of our model, and defer a detailed justification and discussion to the full version [29]:

- We assume that the ciphersuites, KDF, PRF and the stateful AEAD scheme, are fixed once and for all. We do not model ciphersuite negotiation/renegotiation, nor session resumption. In particular, this means that, while our treatment covers multiple ciphersuites (such as those based on RSA key transport and various Diffie-Hellman (DH) ciphersuites) in a modular fashion, our analysis currently does not treat the case where different protocols runs may negotiate different ciphersuites. This requires the application of a suitable composability framework that is beyond the immediate scope of this paper.
- In the case of TLS-RSA, $(\mathsf{KeyGen}, \mathsf{F}_C, \mathsf{F}_S)$ represents the algorithms of the RSA PKCS#1v1.5 encryption scheme (c.f. Section 5.2). The specifics of this encoding were analysed in detail in [12, 25]. For this mode, we assume that the outcome of processing CRES at the server end is completely hidden from the adversary. Such an assumption is necessary; otherwise, TLS-RSA is susceptible to Bleichenbacher's attack [12]. Formally, we model this by treating $\text{CRES}\|\text{CFIN}$ as a monolithic message in the proof of security.
- Our generic description includes the TLS-DH mode, where the server has a certificate on a static DH key PK_S and DH key exchange is used to establish PMS. Here $(\text{CRES}, \text{PMS}) \leftarrow \mathsf{F}_C(\text{PK}_S)$ denotes the client's computation of an ephemeral DH value (CRES) and the pre-master secret (PMS); $\text{PMS} \leftarrow \mathsf{F}_S(\text{SK}_S, \text{CRES})$ denotes the corresponding computation on the server side. In this situation, we may

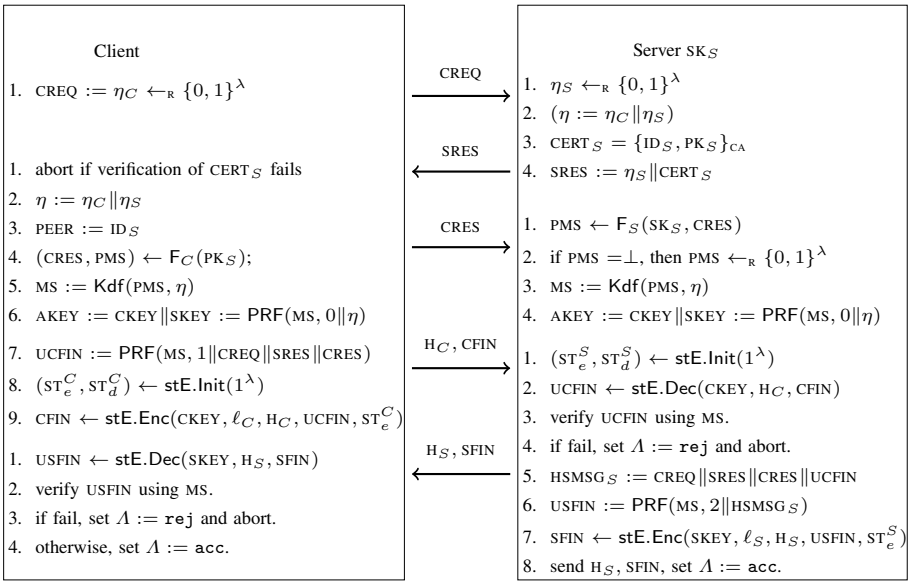


Fig. 2. Basic Generic TLS Handshake Protocol Parameterized by $(\mathsf{KeyGen}, \mathsf{F}_C, \mathsf{F}_S)$

alternatively think of $(\mathsf{KeyGen}, \mathsf{F}_C, \mathsf{F}_S)$ as being the algorithms of a Diffie-Hellman (or Elgamal-type) KEM based on public key PK_S. See Section 6. Specifically, with appropriate choices of Diffie-Hellman groups, our analysis covers the DH_DSS, DH_RSA, ECDH_ECDSA, and ECDH_RSA key exchange methods from [17, 11]; here the suffix DSS/RSA/ECDSA has no meaning since the server does not sign in this mode.

- By suitably extending CERT_S to include the server's signature on its choice of ephemeral DH value, our description captures the TLS-DHE mode, where now PK_S is a signature verification key and PMS is the result of a DH key exchange based on the ephemeral values chosen by client and server. This then covers the DHE_DSS, DHE_RSA, ECDHE_ECDSA, and ECDHE_RSA key exchange methods from [17, 11], where the suffix DSS/RSA/ECDSA refers to the signature scheme used by the server.
- Our description also captures TLS-CCA, where $(\mathsf{KeyGen}, \mathsf{F}_C, \mathsf{F}_S)$ represents the algorithms of an IND-CCA-secure encryption scheme, such as RSA-OAEP.
- For notational simplicity, we use the same symbol λ to denote the security parameter, as well as the bit length of PMS, MS, CKEY and SKEY.

TLS Record Protocol. A party that concludes the TLS Handshake protocol successfully continues to use the application key AKEY $=$ CKEY$\|$SKEY in the TLS Record Protocol. Specifically, the client uses CKEY to encrypt messages to the server, and the server uses the same key CKEY to decrypt these messages. Similarly, the server uses SKEY to encrypt messages to the client, and the client uses the same key SKEY to

decrypt these messages. As noted above, the client and server `Finished` messages are already encrypted in this way. As in [24], we model the TLS Record Protocol via a stateful AEAD scheme stE which we will assume to be sLHAE-secure in the sense of [35]. For further details, see the full version [29].

3 Authenticated and Confidential Channel Establishment (ACCE)

We begin with the definition of *authenticated and confidential channel establishment (ACCE)* from [24, 8]. We will describe the syntax for a general ACCE protocol but for security, we consider a specialization to the setting where only the server is authenticated – we call this *server-only authenticated and confidential channel establishment (SACCE)*.

ACCE Protocol. An ACCE protocol is a protocol executed between two parties, a client and a server. In the original description [24], an ACCE protocol has two distinct phases, called the 'pre-accept' phase and the 'post-accept' phase, corresponding to whether a party has accepted a session key in a particular session or not. We dispense with this distinction (though it is still expressed in our security model by making the queries that are available to the adversary depend on an oracle's acceptance state). The parties in the protocol first compute as the session key an application key AKEY. Then, encrypted and authenticated data is transmitted using a symmetric encryption scheme with the application key AKEY. More specifically, AKEY is parsed as $\text{CKEY}||\text{SKEY}$, the client uses CKEY in a stateful AEAD scheme stE to send data to the server, and the server uses SKEY in stE to protect data sent to the client. Henceforth, we will only refer to application keys and not to session keys. Parties also maintain internal state Λ, and clients keep an additional PEER variable. We assume that an ACCE protocol is such that, when a party reaches the state $\Lambda = \mathtt{acc}$, it has already computed an application key AKEY and executed $\mathsf{stE}.\mathsf{Init}$. TLS meets this requirement.

3.1 Execution Environment

Protocol Entities. Following [24, 8], we consider a set of parties $\mathcal{P} = \mathcal{S} \cup \mathcal{C}$, where $\mathcal{S}$ and $\mathcal{C}$ are disjoint and each party $P \in \mathcal{P}$ is a (potential) protocol participant. Moreover, each $P \in \mathcal{S}$ (the servers) have a *unique* key pair $(\text{PK}_P, \text{SK}_P)$, an identity $\text{ID}_P \in \{0,1\}^\lambda$ along with a certificate $\text{CERT}_P := (\text{ID}_P, \text{PK}_P)_{\text{CA}}$ signed by a certification authority CA. We also assume that all the parties in $\mathcal{S}$ have distinct identities.

Session Oracles. To model several sequential and parallel executions of the protocols and sessions, each party P maintains a collection of oracles $\{\pi_1^P, \pi_2^P, \ldots\}$. The oracle π_i^P models party P executing a single instance of a protocol in "session" i. We stress that the session numbers i are just an artefact of our security game – they are designed to provide a means for the adversary to deliver messages to different sessions at different parties. In particular, the protocols and oracles need not even be "aware" of what their session numbers are (i.e. those numbers need not form part of the state).

Each oracle π_i^P maintains as internal state a set of variables comprising:

- $\Lambda \in \{\emptyset, \mathtt{acc}, \mathtt{rej}\}$;
- $\text{AKEY} = \text{CKEY} \| \text{SKEY} \in \{0,1\}^{2\lambda}$, where $\{0,1\}^{2\lambda}$ is the application key space of the protocol;
- if $P \in \mathcal{C}$, then it has an additional PEER variable to denote the intended partner (only client oracles have the PEER variable because only servers have identities);
- if $P \in \mathcal{S}$, then π_i^P also knows the party identity ID_P.

The internal state of each oracle is initialized to $(\Lambda, \text{AKEY}, \text{PEER}) = (\emptyset, \emptyset, \emptyset)$, where $\emptyset$ denotes undefined.

Adversarial Queries. The adversary interacts with the oracles via the following queries:

$\mathsf{Send}(\pi_i^P, m)$: the adversary uses this query to send a message m to oracle π_i^P; the oracle will respond with an outgoing message according to the protocol specification and its internal state. If π_i^P has reached state $\Lambda = \mathtt{acc}$, then it replies with $\perp$. When the attacker asks the first Send-query to an oracle π_i^C where $C \in \mathcal{C}$, the oracle checks whether m is a special "Initiate client session" symbol $\top$, and if so, responds with the first protocol message (which will be a fresh client nonce). The variables Λ, AKEY are also set according to the protocol specification.

$\mathsf{Reveal}(\pi_i^P)$: the oracle π_i^P responds with the contents of the application key AKEY. Note that this query can be issued to π_i^P before it has reached state $\Lambda = \mathtt{acc}$.

$\mathsf{Encrypt}(\pi_i^P, \ell, \text{H}, m_0, m_1)$: if π_i^P has *not* reached state $\Lambda = \mathtt{acc}$, this oracle returns $\perp$. Otherwise, prior to reaching state acc, π_i^P has (by assumption) computed AKEY and run the $\mathsf{stE.Init}$ algorithm of a stateful AEAD scheme $\mathsf{stE} = (\mathsf{stE.Gen}, \mathsf{stE.Init}, \mathsf{stE.Enc}, \mathsf{stE.Dec})$ to define states ST_e, ST_d specific to the oracle π_i^P; the game also samples a random bit b_i^P at this point, parses AKEY as $\text{CKEY} \| \text{SKEY}$, and adds the 5-tuple $(\text{ST}_e, \text{ST}_d, b_i^P, \text{CKEY}, \text{SKEY})$ to the oracle state ST. Now, when receiving the Encrypt query, the message $m_{b_i^P}$ is encrypted along with header data H using algorithm $\mathsf{stE.Enc}$ and key $K = \text{CKEY}$ (if $P \in \mathcal{C}$) or key $K = \text{SKEY}$ (if $P \in \mathcal{S}$) to form a ciphertext of length ℓ, and to update the encryption state ST_e. The resulting ciphertext is returned to the adversary. For details, see Figure 3.

$\mathsf{Decrypt}(\pi_i^P, \text{H}, c)$: this query is intended to allow the adversary to decrypt ciphertexts that would be processed by the communication partner of the oracle π_i^P (a server S if $P \in \mathcal{C}$, and a client C if $P \in \mathcal{S}$). When $b_i^P = 0$, the response is always $\perp$; when $b_i^P = 1$, this query involves the decryption of H and c using algorithm $\mathsf{stE.Dec}$ and the appropriate key K obtained from ST at the oracle: if $P \in \mathcal{C}$, this will be CKEY, and if $P \in \mathcal{S}$, then it will be SKEY. The resulting message (or failure symbol $\perp$) is returned if the query is "out-of-sync". For details, see Figure 3.

Certificate Authority. We assume that there is a single certificate authority (CA), which uses a secure signature scheme casig and its public key is distributed to all the clients. For each $S \in \mathcal{S}$ with public key PK_S, the CA signs the pair $(\text{ID}_S, \text{PK}_S)$ to

$\mathsf{Encrypt}(\pi_i^P, \ell, \mathrm{H}, m_0, m_1)$:
$u \leftarrow u + 1$
$(c^0, \mathrm{ST}_e^0) \leftarrow_{\mathrm{R}} \mathsf{stE.Enc}(K, \ell, \mathrm{H}, m_0, \mathrm{ST}_e)$
$(c^1, \mathrm{ST}_e^1) \leftarrow_{\mathrm{R}} \mathsf{stE.Enc}(K, \ell, \mathrm{H}, m_1, \mathrm{ST}_e)$
If $c_i^0 = \perp$ or $c_i^1 = \perp$ then return $\perp$
Set $c_u = c^{b_i^P}$, $\mathrm{H}_u = \mathrm{H}$ and $\mathrm{ST}_e = \mathrm{ST}_e^{b_i^P}$
Ret c_u

$\mathsf{Decrypt}(\pi_i^P, \mathrm{H}, c)$:
If $b_i^P = 0$ then Ret $\perp$
$v \leftarrow v + 1$
$(m, \mathrm{ST}_d) \leftarrow \mathsf{stE.Dec}(K, \mathrm{H}, c, \mathrm{ST}_d)$
If $v > u$ or $c \neq c_v$ or $\mathrm{H} \neq \mathrm{H}_v$ then $\mathsf{phase} \leftarrow 1$
If $\mathsf{phase} = 1$ then Ret m
Ret $\perp$

Fig. 3. The Encrypt and Decrypt oracles in the ACCE security game

provide a certificate $\mathrm{CERT}_S := (\mathrm{ID}_S, \mathrm{PK}_S)_{\mathrm{CA}}$. We also allow the adversary access to the CA to register any number of parties, not in the set $\mathcal{S}$, with any public key of the adversary's choice.

Matching Conversations. We consider a definition of matching conversation which is specific to TLS (and differs from the one in [24]):

Definition 1 (Matching conversations). *We say that π_i^P has a* matching conversation *with $\pi_j^{P'}$ if (i) either $P \in \mathcal{C}$ and $P' \in \mathcal{S}$, or $P \in \mathcal{S}$ and $P' \in \mathcal{C}$; and (ii) π_i^P accepts; and (iii) the transcripts at both π_i^P and $\pi_j^{P'}$ begin with the same three messages* $(\mathrm{CREQ}, \mathrm{SRES}, \mathrm{CRES})$.

Remark 1. Defining matching conversations as above means that we may treat the vector $(\mathrm{CREQ}, \mathrm{SRES}, \mathrm{CRES})$ as a post-specified session identifer. Observe that these three messages uniquely determine the parties' nonces and server's identity as well as the key PMS which in turn determines the application keys. In addition, these three messages determine the client's `Finished` message, as well as the server's `Finished` message if the server reaches the accept state.

3.2 Correctness and Security

Correctness. For every honest $C \in \mathcal{C}$ and $S \in \mathcal{S}$, if two sessions π_i^C, π_j^S have matching conversations with each other, then we require that they have the same application key AKEY and π_i^C has its PEER variable set to ID_S. We also require that the encryption scheme stE used to model the secure channel is correct.

SACCE Security. Security of an ACCE protocol with server-only authentication (SACCE) is defined by requiring that (i) the protocol provides server authentication (but with no guarantee of client authentication, and that (ii) the subsequent use of the application keys in the stateful AEAD scheme stE provides stateful Length Hiding Authenticated Encryption (sLHAE), as per [35]. We consider a game played between the adversary $\mathcal{A}$ and a challenger. This game is obtained by adapting [24, Definition 7] to our setting. At the beginning of the game, the challenger generates the long-term key-pair $(\mathrm{PK}_S, \mathrm{SK}_S)$ along with the certificate $\mathrm{CERT}_S := (\mathrm{ID}_S, \mathrm{PK}_S)_{\mathrm{CA}}$ for all $S \in \mathcal{S}$ and gives all the certificates to $\mathcal{A}$ as input. Now the adversary issues a sequence of queries

defined before. The challenger answers all queries to π_i^C by running the honest client protocol, and all queries to π_j^S by running the honest server protocol using the key SK_S. The challenger will also provide certificates along with signatures to the adversary for any identities outside the set $\{\mathrm{ID}_S : S \in \mathcal{S}\}$.

Advantage Measures. We associate to an adversary $\mathcal{A}$ against an ACCE protocol Π two advantage measures:

- (server authentication, i.e. client accepts $\Rightarrow$ matching conversations.)
  $\mathbf{Adv}_{\Pi}^{\mathrm{sacce-sa}}(\mathcal{A})$ is the probability that when $\mathcal{A}$ terminates, there is a (honest) client C and oracle π_i^C that reaches an accept state with honest $\mathrm{PEER} = \mathrm{ID}_S$, but there is no unique oracle π_j^S for which π_i^C has had a matching conversation with π_j^S.
- (channel security.)
  $\mathbf{Adv}_{\Pi}^{\mathrm{sacce-ae}}(\mathcal{A})$ is defined to be $p - 1/2$, where p is the probability that $\mathcal{A}$ outputs (P, i, b') such that $b' = b_i^P$ where b_i^P is set during the $\mathsf{Encrypt}(\pi_i^P, \ldots)$ query and we define b' to be $\perp$ unless the following conditions hold: (i) π_i^P reaches an accept state; (ii) π_i^P is not the subject of a Reveal query, and if there is an oracle $\pi_j^{P'}$ with which π_i^P has a matching conversation then $\pi_j^{P'}$ is not the subject of a Reveal query either; and (iii) $P \in \mathcal{C}$.

Definition 2 (SACCE-secure). *We say that an ACCE protocol Π is* SACCE-secure *if Π satisfies correctness, and for all PPT adversaries $\mathcal{A}$, both $\mathbf{Adv}_{\Pi}^{\mathrm{sacce-sa}}(\mathcal{A})$ and $\mathbf{Adv}_{\Pi}^{\mathrm{sacce-ae}}(\mathcal{A})$ are a negligible function of the security parameter λ.*

4 From CCCA KEM Security to SACCE Security of TLS

In this section we state the following theorem which is our core intermediate result for proving ACCE security of all TLS modes. It uses the notion of CCCA security and the definition of the TLS KEM tlskem introduced below in Sections 4.1 and 4.2, respectively.

Theorem 1. *If* tlskem *is IND-CCCA secure,* casig *is an existentially unforgeable signature scheme and* stE *is sLHAE-secure then TLS is SACCE-secure.*

The IND-CCCA security of the KEMs arising from all TLS modes (and hence the SACCE security of these modes) is shown in the subsequent sections.

4.1 IND-CCCA Security

We consider a variant of IND-CCCA security from [23]:

Definition 3 (IND-CCCA). *For a stateful adversary $\mathcal{A}$, an LKEM* lkem *and a predicate* pred*, we define the advantage function*

$$\mathbf{Adv}_{\mathsf{lkem},\mathsf{pred}}^{\mathrm{ind-ccca}}(\mathcal{A}) := \Pr\left[b = b' : \begin{array}{l} (\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{KeyGen}(1^\lambda); \\ L^* \leftarrow \mathcal{A}^{\mathsf{CDec}(\mathrm{SK},\cdot,\cdot,\cdot)}(\mathrm{PK}); \\ (C^*, K^*) \leftarrow \mathsf{Enc}(\mathrm{PK}, L^*); \\ K_0 := K^*; K_1 \leftarrow_{\mathrm{R}} \{0,1\}^\lambda; b \leftarrow_{\mathrm{R}} \{0,1\}; \\ b' \leftarrow \mathcal{A}^{\mathsf{CDec}(\mathrm{SK},\cdot,\cdot,\cdot)}(C^*, K_b) \end{array} \right] - \frac{1}{2}$$

with the restriction that (1) L^* must be different from all previously queried L, and (2) the restriction on the decryption oracle for queries after getting the challenge ciphertext is $(L, C) \neq (L^*, C^*)$; and where the "constrained" decryption oracle CDec *is given by:*

$\mathsf{CDec}(\mathrm{SK}, L, C, T)$:
$K \leftarrow_{\mathrm{R}} \mathsf{Dec}(\mathrm{SK}, L, C)$
if $K = \perp$ *or* $\mathsf{pred}(K, T) = 0$ *then return* $\perp$
else return K

A LKEM lkem *is said to be* IND-CCCA-secure *if for all PPT adversaries* $\mathcal{A}$*, the advantage* $\mathbf{Adv}^{\mathrm{ind-ccca}}_{\mathsf{lkem},\mathsf{pred}}(\mathcal{A})$ *is a negligible function in* λ*.*

Remark 2 (comparison with [23]). We point out the differences between our formulation and that in [23]. First, we consider a setting with labels. Second, in [23], the predicate is specified by the adversary via a circuit. Here, we consider a fixed predicate that takes an additional input T. To capture the prior formulation, the predicate would be circuit evaluation and T would be a circuit. Third, by fixing the predicate, we avoid having to explicitly consider plaintext uncertainty.

4.2 The TLS Labeled KEM

Following [25], we describe a labeled KEM which is extracted from the TLS protocol. Unlike [25] which stopped at analyzing (labeled) CCA-security of the ensuing scheme for the RSA mode of TLS, we show how to derive SACCE security of TLS (for any mode) based on the IND-CCCA security of the labeled KEM. We then prove this IND-CCCA property to hold for the KEMs arising in various TLS modes, namely TLS-RSA, TLS-CCA, TLS-DH, and TLS-DHE.

LKEMs from TLS. Given a generic TLS protocol parameterized by $(\mathsf{KeyGen}, \mathsf{F}_C, \mathsf{F}_S)$ (see Fig. 2) along with cryptographic components Kdf and PRF, we consider the LKEM tlskem with algorithms $(\mathsf{tls.Gen}, \mathsf{tls.Enc}, \mathsf{tls.Dec})$, in which $\mathsf{tls.Gen}(1^\lambda)$ is the same as KeyGen and the algorithms $\mathsf{tls.Enc}, \mathsf{tls.Dec}$ are as below.

$\underline{\mathsf{tls.Enc}(\mathrm{PK}, \eta \| \mathrm{CERT}_S)}$:
$(\mathrm{CRES}, \mathrm{PMS}) \leftarrow \mathsf{F}_C(\mathrm{PK})$;
$\mathrm{MS} := \mathsf{Kdf}(\mathrm{PMS}, \eta)$;
$\mathrm{UCFIN} := \mathsf{PRF}(\mathrm{MS}, 1 \| \eta \| \mathrm{CERT}_S \| \mathrm{CRES})$;
$\mathrm{AKEY} := \mathsf{PRF}(\mathrm{MS}, 0 \| \eta)$;
$\mathrm{USFIN} :=$
$\mathsf{PRF}(\mathrm{MS}, 2 \| \eta \| \mathrm{CERT}_S \| \mathrm{CRES} \| \mathrm{UCFIN})$;
output $(\mathrm{CRES}, \ \mathrm{AKEY} \| \mathrm{USFIN} \| \mathrm{UCFIN})$.

$\underline{\mathsf{tls.Dec}(\mathrm{SK}, \eta \| \mathrm{CERT}_S, \mathrm{CRES})}$:
$\mathrm{PMS} \leftarrow \mathsf{F}_S(\mathrm{SK}, \mathrm{CRES})$;
if $\mathrm{PMS} = \perp$, set $\mathrm{PMS} \leftarrow_{\mathrm{R}} \{0,1\}^\lambda$;
$\mathrm{MS} := \mathsf{Kdf}(\mathrm{PMS}, \eta)$;
$\mathrm{UCFIN} := \mathsf{PRF}(\mathrm{MS}, 1 \| \eta \| \mathrm{CERT}_S \| \mathrm{CRES})$;
$\mathrm{AKEY} := \mathsf{PRF}(\mathrm{MS}, 0 \| \eta)$;
$\mathrm{USFIN} := \mathsf{PRF}(\mathrm{MS}, 2 \| \eta \| \mathrm{CERT}_S \| \mathrm{CRES} \| \mathrm{UCFIN})$;
output $\mathrm{AKEY} \| \mathrm{USFIN} \| \mathrm{UCFIN}$.

In order to consider CCCA security we augment tlskem with the following predicate.

$\underline{\mathsf{tls.Pred}(\mathrm{AKEY} \| \mathrm{USFIN} \| \mathrm{UCFIN}, \mathrm{CFIN})}$:
$(\mathrm{ST}_e^S, \mathrm{ST}_d^S) \leftarrow \mathsf{stE.Init}(1^\lambda)$;
check if $\mathrm{UCFIN} = \mathsf{stE.Dec}(\mathrm{CKEY}, \mathrm{H}_C, \mathrm{CFIN}, \mathrm{ST}_d^S)$.

5 TLS-RSA: Instantiations from OW-PCA

Here, we show that if the underlying KEM $(\mathsf{KeyGen}, \mathsf{F}_C, \mathsf{F}_S)$ is OW-PCA secure, then the tlskem scheme in Section 4.2 is IND-CCCA-secure in the random oracle model. Hence, by Theorem 1, the corresponding TLS scheme is SACCE-secure. In Section 5.2 we apply this result to show the security of TLS-RSA.

OW-PCA for KEM [34]. For a stateful adversary $\mathcal{A}$ and a KEM kem with algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, we define the advantage function

$$\mathbf{Adv}^{\mathrm{ow-pca}}_{\mathsf{kem}}(\mathcal{A}) := \Pr\left[K' = K^* : \begin{array}{l}(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{KeyGen}(1^\lambda);\\ (\psi^*, K^*) \leftarrow \mathsf{Enc}(\mathrm{PK});\\ K' \leftarrow \mathcal{A}^{\mathsf{PCA}(\mathrm{SK},\cdot,\cdot)}(\psi^*)\end{array}\right]$$

where $\mathsf{PCA}(\mathrm{SK}, \cdot, \cdot)$ is the oracle that takes as input (K, ψ) with $K \neq \perp$ and outputs 1 if $\mathsf{Dec}(\mathrm{SK}, \psi) = K$ and 0 otherwise. An encryption scheme is said to be *one-way against plaintext checking attacks* (OW-PCA) if for all PPT adversaries $\mathcal{A}$, the advantage $\mathbf{Adv}^{\mathrm{ow-pca}}_{\mathsf{kem}}(\mathcal{A})$ is a negligible function in λ.

5.1 IND-CCCA from OW-PCA

The following lemma is similar to that in [25, Theorem 3], with some significant differences: (i) the KEM key in [25, Theorem 3] is AKEY and the ciphertext is CRES$\|$UCFIN, whereas our KEM key is AKEY$\|$USFIN$\|$UCFIN and the ciphertext is simply CRES; (ii) [25] models PRF (referred to as h_s and h_z therein) also as a random oracle; (iii) [25] proves (labeled) IND-CCA security and does not consider encryption of UCFIN.

Lemma 1 (IND-CCCA from OW-PCA). *If* $(\mathsf{KeyGen}, \mathsf{F}_C, \mathsf{F}_S)$ *is OW-PCA secure,* PRF *is a pseudorandom function, and we model* Kdf() *as a random oracle, then the LKEM* tlskem *with predicate* tls.Pred *(in Section 4.2) is IND-CCCA secure in the random oracle model (c.f. Section 4.1). That is, for any adversary* $\mathcal{A}$ *that makes at most* Q *decryption queries, there exists adversaries* $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_4$ *such that*

$$\mathbf{Adv}^{\mathrm{ind-ccca}}_{\mathsf{lkem},\mathsf{pred}}(\mathcal{A}) \leq Q \cdot (\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{PRF}}(\mathcal{A}_1) + 2^{-\lambda}) + \mathbf{Adv}^{\mathrm{ow-pca}}_{\mathsf{kem}}(\mathcal{A}_2) + \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{PRF}}(\mathcal{A}_4).$$

Moreover, the running times of $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_4$ *are roughly that of* $\mathcal{A}$.

5.2 Implications for TLS-RSA

TLS-RSA KEM. The definition of the TLS-RSA KEM follows from the RSA PKCS #1v1.5 standard [26] adopted in TLS. Building on [25] we abstract the specification with parameters $\lambda_0 = \Theta(\lambda)$, $\lambda_1 = \Theta(\lambda)$ with $\lambda_0 \leq \lambda_1 - 88$ as follows:

- $\mathsf{KeyGen}(1^\lambda)$ is standard RSA key generation that outputs $(\mathrm{PK}, \mathrm{SK}) := ((N, e), d)$ where $de = 1 \pmod{\phi(N)}$ and N has λ_1 bits, where we assume that λ_1 is a multiple of 8.

- $\mathsf{F}_C : \{0,1\}^{\lambda_0} \to \mathbb{Z}_N^*$ takes as input λ_0-bit r, picks a padding $P \leftarrow_R \{0,1\}^{\lambda_1 - \lambda_0 - 24}$ at random (subject to none of the bytes of P being 00), sets $x := 00\|02\|P\|00\|r$ (where $00, 02$ are byte encodings), and outputs $y := x^e \pmod N$.
- $\mathsf{F}_S : \mathbb{Z}_N^* \to \{0,1\}^{\lambda_0}$ takes as input y, and attempts to parse $y^d \pmod N$ as a byte sequence of the form $00\|02\|P\|00\|r$ where P contains no zero bytes and r has exactly λ_0 bits. The procedure then outputs r if the parsing is successful (and $\perp$ if the parsing fails).

Here, the condition that $\lambda_0 \leq \lambda_1 - 88$ ensures that the random padding P has at least 8 bytes, as required by the standard [26]. We also assume in our description that KEM decapsulation involves performing a strict set of parsing checks.

The assumption that RSA PKCS #1v1.5 is OW-PCA is justified in [25, Theorem 1] via a reduction to an RSA-like assumption, known as "partial-domain RSA with decision oracle". The latter, given in [25, Section 2.3], asserts that the RSA permutation is one-way, even given an oracle that is parameterized by $\lambda_0 < \lambda_1$, takes as input (x_0, y), and reports whether the first λ_0 bits of $y^d \bmod N$ equals x_0 or not. A close examination of the proof of [25, Theorem 1] shows that the theorem holds no matter what set of parsing checks are carried out during decapsulation (so long as the decapsulation algorithm is correct). This is convenient because, as recent work [7] has shown, there is a good deal of variation in how the required parsing is done in different PKCS #1v1.5 implementations.[2]

In TLS-RSA, λ_0 is fixed to 384, reflecting the fixed size of PMS (at 48 bytes) in the TLS specification, while λ_1 (the bit-size of N) is typically 1024 or 2048 in TLS deployments. Jonsson and Kaliski in [25] discuss at some length why the above assumption is reasonable for typical parameters λ_0, λ_1 used in practice. While seemingly strong, it seems hard to avoid using an assumption of this type given the many known weaknesses in RSA-PKCS#1 v1.5. We are not aware of any further significant work studying this assumption. In particular, in spite of the importance of the widely-deployed PKCS #1v1.5 scheme and its use in TLS, to the best of our knowledge no weaknesses on the assumption have been reported since its introduction in [25] over 10 years ago.

The security of TLS-RSA follows from Theorem 1 and Lemma 1:

Theorem 2. *Under the following assumptions:*

- *RSA PKCS #1v1.5 is OW-PCA;*
- PRF *is a secure pseudorandom function;*
- stE *is a sLHAE encryption scheme,*

the TLS-RSA Protocol is a secure SACCE protocol in the random oracle model.

[2] This property of the proof would also allow us to incorporate into our analysis the additional check from the TLS specification that the leading 2 bytes of r should be an encoding of the TLS protocol version as sent by the client, at the cost of reducing λ_0, the bit-size of r, by 16. However, we omit this fine detail.

6 TLS-DH: Instantiation from PRF-ODH

We prove the security of TLS-DH following our methodology: We show that the KEM $(\mathsf{KeyGen}, \mathsf{F}_C, \mathsf{F}_S)$ that instantiates this mode in accordance with our generic representation of TLS (Figure 2) induces a labeled KEM, $\mathsf{dh.tlskem}$, that is IND-CCCA secure. Then, by Theorem 1 we conclude that TLS-DH is a secure SACCE protocol. Finally, we apply these results to TLS-DHE, namely, when both client and server provide ephemeral DH keys. Here, we merely provide a summary of our results; details can be found in the full version [29].

TLS-DH KEM. Let $G = \langle g \rangle$ be a cyclic group of prime order q generated by an element g. We define the TLS-DH KEM $(\mathsf{KeyGen}, \mathsf{F}_C, \mathsf{F}_S)$ via the following three algorithms.

- $\mathsf{KeyGen}(1^\lambda)$: Set $(\mathrm{PK}, \mathrm{SK}) := (g^v, v), v \leftarrow_{\mathrm{R}} \mathbb{Z}_q, |q| = \lambda$.
- $\mathsf{F}_C(\mathrm{PK} = g^v)$: Set $(\psi, K) := (g^u, g^{uv}), u \leftarrow_{\mathrm{R}} \mathbb{Z}_q$.
- $\mathsf{F}_S(\mathrm{SK} = v, h)$: Check that $h \in G$, if yes, output h^v, else output $\perp$ (reject).

We show the security of TLS-DH in the standard model based on the PRF-ODH assumption on the function Kdf. This assumption is an adaptation of the Oracle Diffie-Hellman (ODH) assumption [1] to the PRF setting and was introduced in [24] for their proof of TLS-DHE. For the proof of TLS-DH we need the multi-query version of the assumption while for TLS-DHE the single-query case is sufficient, as in [24]. In [29] we describe the assumption and also prove its necessity by constructing secure pseudorandom functions for which PRF-ODH does not hold and with which TLS-DH violates ACCE security.

Theorem 3. *Protocol TLS-DH obtained by instatiating the generic TLS protocol from Figure 2 with the above defined TLS-DH KEM is a secure SACCE protocol provided* Kdf *is PRF-ODH,* PRF *is a secure pseudorandom function, and* stE *is an sLHAE-secure encryption scheme.*

Extension to TLS-DHE with server-signed ephemeral DH (and no client authentication). In this variant of TLS-DH, the certified server's public key corresponds to a signature algorithm and the DH value, typically an ephemeral one, is signed by the server itself. All other details are exactly as in TLS-DH. The security of this protocol follows essentially from the analysis of TLS-DH by replacing $\mathrm{CERT}_S = \{\mathrm{ID}_S, \mathrm{PK} = g^v\}_{\mathrm{CA}}$ with $\mathrm{CERT}_S = (\mathrm{CERTSIG}_S, \mathsf{sig}_S(g^v, \ldots), g^v)$, where $\mathrm{CERTSIG}_S = \{\mathrm{ID}_S, \mathrm{PKSIG}_S\}_{\mathrm{CA}}$, PKSIG_S is a public key of S for a secure signature scheme, and sig_S is a signature produced by S under the corresponding signature key.

We note that [24] provided a specialized proof of TLS-DHE with client authentication. We obtain a proof for that particular case in Section 7 (Corollary 1). However, while [24] show forward security we do not include this property in our general treatment as it is not achieved by any of the other TLS modes.

7 The TLS Handshake Protocol with Mutual Authentication

In the full version [29] we augment server-only authentication, the SACCE model, with client authentication to obtain mutual authentication. The resultant model, ACCE, is mostly the same as in [24] except for the definition of matching sessions presented in Section 3. In this setting, the TLS client possesses a signature key (with a corresponding certificate CERT_C) which it uses to authenticate to the server (by computing a signature on the session transcript up to the CRES message). In this case both the definition of matching conversations and the TLS LKEM are augmented with CERT_C.

We first extend Theorem 1 to the case of mutual authentication, namely, showing that if tlskem is IND-CCCA secure and stE is sLHAE-secure then TLS with client and server authentication is ACCE-secure. Then we immediately obtain the following powerful corollary:

Corollary 1. *TLS-RSA, TLS-CCA, TLS-DH and TLS-DHE where the client is authenticated using a secure signature scheme are all ACCE secure under the same assumptions stated for the SACCE security of these modes.*

8 Conclusions and Discussion

Establishing the security of a central protocol like TLS is clearly a significant result. The fact that we achieve this in a systematic way that covers the different TLS modes (TLS-RSA, TLS-DH, and TLS-DHE, as well as the hypothetical TLS-CCA), has clear methodological advantages and may be seen as indicating that the core design of the protocol is sound. Yet, we find it important to stress the many shortcomings of the TLS protocol that would be best avoided in future secure channel protocol designs. We summarise these issues here, expanding on them in [29].

On the TLS Design. A main weakness in the overall TLS design is the unfortunate interaction of the TLS Handshake and the Record protocols with respect to the `Finished` messages. Instead, these two conceptually and functionally different parts of a secure channel protocol can and should be designed as separate components. This separation makes engineering sense for implementing and maintaining the protocol, especially in evolving application settings as is the case with TLS. It also makes formal security analysis of the protocol easier, which in turn increases the likelihood that this analysis will be correct and applicable to the actual protocol under study.

In addition to the above structural weakness of the protocol, TLS has suffered from the early implementation of its public key encryption mode, TLS-RSA, with RSA PKCS#1v1.5. Rather than moving to a CCA-secure implementation of the encryption function (e.g., via RSA-OAEP), the TLS community responded to Bleichenbacher's attack by keeping RSA PKCS#1v1.5 as the default implementation but disabling the error message in case of a decryption error. We stress that our proof of security for TLS-RSA relies crucially on there being no side channel that would reveal the existence of decryption failures to the attacker. While the TLS specification now takes care to avoid some explicit forms of leaking this information, implementations may still find ways

of leaking it. This makes the security of the protocol non-robust and shows the clear advantage of using a CCA-secure scheme in TLS that, as we show, would avoid these complications and potential weaknesses.

The fragility of the TLS-RSA design is further illustrated by the somewhat "accidental" nature of its security. The security of this mode relies crucially on the fact that the client's `Finished` message, CFIN, is sent immediately after the CRES message (containing the RSA ciphertext in TLS-RSA) and before the server responds with its own `Finished` message SFIN. Had CFIN been omitted or sent after SFIN, TLS-RSA would be completely insecure, e.g. subject to Bleichenbacher's attack.

On Our Attack Model. We caution that, while our work shows that accurate descriptions of already-deployed, complex protocols can be analysed using the provable security paradigm, we only analyse the "cryptographic core" of TLS. This means that our analysis rules out many (but not all) attacks. More specifically:

- Several recent attacks on the TLS Record Protocol are possible because TLS supports symmetric encryption algorithms that have turned out *not* to be sLHAE-secure: see, for example, the BEAST attack [18], the short MAC attack [35], Lucky 13 [3], and the RC4 attacks in [4]. Such attacks can be mounted by the adversary in our security model but are ruled out by our theorem statements, which assume the use of an sLHAE-secure encryption scheme.
- Our description of TLS-RSA includes the standard countermeasures to Bleichenbacher's attack, and our security proof then gives assurance that these countermeasures are effective in the context of the entire TLS protocol.

On the other hand, we do not treat ciphersuite (re)negotiation nor the TLS Record Protocol's compression or fragmentation features, meaning that, for example, none of the attacks from [19, 30, 38] are covered by our analysis.

A Final Thought. We believe that one cannot overstate the importance of adopting protocols in practice that have been first rigorously analyzed and proven in a plausible cryptographic model. Such proofs are necessarily limited by the expressiveness of the underlying model and do not guarantee security in every imaginable deployment setting, yet they can serve as a major source of confidence in the soundness of the design. This is particularly important given the practical difficulty in changing protocols when weaknesses are found – and TLS serves as a good example for the latter.

References

[1] Abdalla, M., Bellare, M., Rogaway, P.: The oracle diffie-hellman assumptions and an analysis of DHIES. In: CT-RSA 2001. LNCS, vol. 2020, pp. 143–158. Springer, Heidelberg (2001)

[2] AlFardan, N., Paterson, K.G.: Plaintext-recovery attacks against Datagram TLS. In: Network and Distributed System Security Symposium (NDSS 2012) (2012)

[3] AlFardan, N., Paterson, K.G.: Lucky thirteen: Breaking the TLS and DTLS record protocols. In: IEEE Symposium on Security and Privacy (2013), `www.isg.rhul.ac.uk/tls/Lucky13.html`

[4] AlFardan, N., Bernstein, D.J., Paterson, K.G., Poettering, B., Schuldt, J.C.: On the security of RC4 in TLS and WPA. In: USENIX Security Symposium (2013), www.isg.rhul.ac.uk/tls
[5] Bard, G.V.: The vulnerability of SSL to chosen plaintext attack. IACR Cryptology ePrint Archive, 2004:11 (2004)
[6] Bard, G.V.: A challenging but feasible blockwise-adaptive chosen-plaintext attack on SSL. In: SECRYPT, pp. 99–109 (2006)
[7] Bardou, R., Focardi, R., Kawamoto, Y., Simionato, L., Steel, G., Tsay, J.-K.: Efficient padding oracle attacks on cryptographic hardware. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 608–625. Springer, Heidelberg (2012)
[8] Bellare, M., Rogaway, P.: Entity authentication and key distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
[9] Bhargavan, K., Fournet, C., Corin, R., Zalinescu, E.: Verified cryptographic implementations for TLS. ACM Trans. Inf. Syst. Secur. 15(1), 3 (2012)
[10] Bhargavan, K., Fournet, C., Kohlweiss, M., Pironti, A., Strub, P.-Y.: Implementing TLS with verified cryptographic security. In: IEEE Symposium on Security and Privacy (2013), http://mitls.rocq.inria.fr/
[11] Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., Moeller, B.: Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security, TLS (May 2006), http://www.rfc-editor.org/rfc/rfc4492.txt
[12] Bleichenbacher, D.: Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 1–12. Springer, Heidelberg (1998)
[13] Brzuska, C., Fischlin, M., Smart, N., Warinschi, B., Williams, S.: Less is more: Relaxed yet composable security notions for key exchange. Cryptology ePrint Archive, Report 2012/242 (2012)
[14] Canetti, R., Krawczyk, H.: Analysis of key-exchange protocols and their use for building secure channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001), See also Cryptology ePrint Archive, Report 2001/040
[15] Canetti, R., Krawczyk, H.: Universally composable notions of key exchange and secure channels. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 337–351. Springer, Heidelberg (2002), See also Cryptology ePrint Archive, Report 2002/059
[16] Canvel, B., Hiltgen, A.P., Vaudenay, S., Vuagnoux, M.: Password Interception in a SSL/TLS Channel. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 583–599. Springer, Heidelberg (2003)
[17] Dierks, T., Rescorla, E.: The Transport Layer Security (TLS) Protocol Version 1.2 (August 2008), http://www.rfc-editor.org/rfc/rfc5246.txt
[18] Duong, T., Rizzo, J.: Here come the $\oplus$ Ninjas (2011) (unpublished manuscript)
[19] Duong, T., Rizzo, J.: The CRIME attack. Presentation at Ekoparty Security Conference (2012), http://www.ekoparty.org/eng/2012/juliano-rizzo.php
[20] Fahl, S., Harbach, M., Muders, T., Smith, M., Baumgärtner, L., Freisleben, B.: Why Eve and Mallory love Android: An analysis of Android SSL (in)security. In: ACM CCS, pp. 50–61 (2012)
[21] Georgiev, M., Iyengar, S., Jana, S., Anubhai, R., Boneh, D., Shmatikov, V.: The most dangerous code in the world: Validating SSL certificates in non-browser software. In: ACM CCS, pp. 38–49 (2012)
[22] He, C., Sundararajan, M., Datta, A., Derek, A., Mitchell, J.C.: A modular correctness proof of IEEE 802.11i and TLS. In: ACM CCS, pp. 2–15 (2005)

[23] Hofheinz, D., Kiltz, E.: Secure hybrid encryption from weakened key encapsulation. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 553–571. Springer, Heidelberg (2007)
[24] Jager, T., Kohlar, F., Schäge, S., Schwenk, J.: On the security of TLS-DHE in the standard model. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 273–293. Springer, Heidelberg (2012)
[25] Jonsson, J., Kaliski Jr., B.S.: On the security of RSA encryption in TLS. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 127–142. Springer, Heidelberg (2002)
[26] Kaliski, B.: PKCS#1: RSA Encryption Version 1.5 (March 1998), http://www.rfc-editor.org/rfc/rfc2313.txt
[27] Klíma, V., Pokorný, O., Rosa, T.: Attacking RSA-based sessions in SSL/TLS. In: Walter, C.D., Koç, Ç.K., Paar, C. (eds.) CHES 2003. LNCS, vol. 2779, pp. 426–440. Springer, Heidelberg (2003)
[28] Krawczyk, H.: The order of encryption and authentication for protecting communications (or: How secure is SSL?). In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 310–331. Springer, Heidelberg (2001)
[29] Krawczyk, H., Paterson, K.G., Wee, H.: On the security of the TLS protocol: A systematic analysis. In: Canetti, R., Garay, J. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 425–444. Springer, Heidelberg (2013); Cryptology ePrint Archive, Report 2013/339
[30] Mavrogiannopoulos, N., Vercauteren, F., Velichkov, V., Preneel, B.: A cross-protocol attack on the TLS protocol. In: ACM CCS, pp. 62–72 (2012)
[31] Modadugu, N., Rescorla, E.: The Design and Implementation of Datagram TLS. In: NDSS. The Internet Society (2004) ISBN 1-891562-18-5, 1-891562-17-7
[32] Moeller, B.: Security of CBC ciphersuites in SSL/TLS: Problems and countermeasures (May 2004) (unpublished manuscript), http://www.openssl.org/~bodo/tls-cbc.txt
[33] Morrissey, P., Smart, N.P., Warinschi, B.: A modular security analysis of the TLS handshake protocol. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 55–73. Springer, Heidelberg (2008)
[34] Okamoto, T., Pointcheval, D.: REACT: Rapid enhanced-security asymmetric cryptosystem transform. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 159–175. Springer, Heidelberg (2001)
[35] Paterson, K.G., Ristenpart, T., Shrimpton, T.: Tag size *does* matter: Attacks and proofs for the TLS record protocol. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 372–389. Springer, Heidelberg (2011)
[36] Paulson, L.C.: Inductive analysis of the internet protocol TLS. ACM Trans. Inf. Syst. Secur. 2(3), 332–351 (1999)
[37] Rescorla, E., Modadugu, N.: Datagram Transport Layer Security (April 2006), http://www.rfc-editor.org/rfc/rfc4347.txt
[38] Rescorla, E., Ray, M., Dispensa, S., Oskov, N.: Transport Layer Security (TLS) Renegotiation Indication Extension. In: RFC 5746 (Proposed Standard) (February 2010), http://www.ietf.org/rfc/rfc5746.txt
[39] Shoup, V.: On formal models for secure key exchange. Cryptology ePrint Archive. Report 1999/012 (1999), http://eprint.iacr.org/
[40] Vaudenay, S.: Security Flaws Induced by CBC Padding - Applications to SSL, IPSEC, WTLS... In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 534–546. Springer, Heidelberg (2002)
[41] Wagner, D., Schneier, B.: Analysis of the SSL 3.0 protocol. In: USENIX Workshop on Electronic Commerce, pp. 29–40 (1996)

New Techniques for SPHFs and Efficient One-Round PAKE Protocols

Fabrice Benhamouda[1], Olivier Blazy[2], Céline Chevalier[3], David Pointcheval[1], and Damien Vergnaud[1]

[1] ENS, Paris, France*
[2] Ruhr-Universität Bochum, Germany
[3] Université Panthéon–Assas, Paris, France

Abstract. *Password-authenticated key exchange* (PAKE) protocols allow two players to agree on a shared high entropy secret key, that depends on their own passwords only. Following the Gennaro and Lindell's approach, with a new kind of *smooth-projective hash functions* (SPHFs), Katz and Vaikuntanathan recently came up with the first concrete one-round PAKE protocols, where the two players just have to send simultaneous flows to each other. The first one is secure in the Bellare-Pointcheval-Rogaway (BPR) model and the second one in the Canetti's UC framework, but at the cost of *simulation-sound non-interactive zero-knowledge* (SS-NIZK) proofs (one for the BPR-secure protocol and two for the UC-secure one), which make the overall constructions not really efficient.

This paper follows their path with, first, a new efficient instantiation of SPHF on Cramer-Shoup ciphertexts, which allows to get rid of the SS-NIZK proof and leads to the design of the most efficient one-round PAKE known so far, in the BPR model, and in addition without pairings.

In the UC framework, the security proof required the simulator to be able to extract the hashing key of the SPHF, hence the additional SS-NIZK proof. We improve the way the latter extractability is obtained by introducing the notion of *trapdoor smooth projective hash functions* (TSPHFs). Our concrete instantiation leads to the most efficient one-round PAKE UC-secure against static corruptions to date.

We additionally show how these SPHFs and TSPHFs can be used for blind signatures and zero-knowledge proofs with straight-line extractability.

1 Introduction

Authenticated Key Exchange. Protocols are quite important primitives for practical applications, since they enable two parties to generate a shared high entropy secret key, to be later used with symmetric primitives in order to protect communications, while interacting over an insecure network under the control of an adversary. Various authentication means have been proposed, and the most

* CNRS – UMR 8548 and INRIA – EPI Cascade.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 449–475, 2013.

practical one is definitely a shared low entropy secret, or a password they can agree on over the phone, hence PAKE, for *Password-Authenticated Key Exchange*. The most famous instantiation has been proposed by Bellovin and Merritt [4], EKE for Encrypted Key Exchange, which simply consists of a Diffie-Hellman key exchange [17], where the flows are symmetrically encrypted under the shared password. Overall, the equivalent of 2 group elements have to be sent.

A first formal security model was proposed by Bellare, Pointcheval and Rogaway [3] (the BPR model), to deal with off-line dictionary attacks. It essentially says that the best attack should be the on-line exhaustive search, consisting in trying all the passwords by successive executions of the protocol with the server. Several variants of EKE with BPR-security proofs have been proposed in the ideal-cipher model or the random-oracle model [27]. Katz, Ostrovsky and Yung [23] proposed the first practical scheme (KOY), provably secure in the standard model under the DDH assumption. This is a 3-flow protocol, with the client sending 5 group elements plus a verification key and a signature, for a one-time signature scheme, and the server sending 5 group elements. It has been generalized by Gennaro and Lindell [20] (GL), making use of smooth projective hash functions.

Smooth Projective Hash Functions. (SPHFs) were introduced by Cramer and Shoup [16] in order to achieve IND-CCA security from IND-CPA encryption schemes, which led to the first efficient IND-CCA encryption scheme provably secure in the standard model under the DDH assumption [15]. They can be seen as a kind of implicit designated-verifier proofs of membership [1, 9]. Basically, SPHFs are families of pairs of functions $(\mathsf{Hash}, \mathsf{ProjHash})$ defined on a language L. These functions are indexed by a pair of associated keys $(\mathsf{hk}, \mathsf{hp})$, where hk, the hashing key, can be seen as the private key and hp, the projection key, as the public key. On a word $W \in \mathrm{L}$, both functions should lead to the same result: $\mathsf{Hash}(\mathsf{hk}, \mathrm{L}, W)$ with the hashing key and $\mathsf{ProjHash}(\mathsf{hp}, \mathrm{L}, W, w)$ with the projection key only but also a witness w that $W \in \mathrm{L}$. Of course, if $W \notin \mathrm{L}$, such a witness does not exist, and the smoothness property states that $\mathsf{Hash}(\mathsf{hk}, \mathrm{L}, W)$ is independent of hp. As a consequence, even knowing hp, one cannot guess $\mathsf{Hash}(\mathsf{hk}, \mathrm{L}, W)$.

One-Round PAKE in the BPR Model. Gennaro and Lindell [20] (GL) extended the initial definition of smooth projective hash functions for an application to PAKE. Their approach has thereafter been adapted to the *Universal Composability* (UC) framework by Canetti *et al.* [14], but for static corruptions only. It has been improved by Abdalla, Chevalier and Pointcheval [1] to resist to adaptive adversaries. But the 3-flow KOY protocol remains the most efficient protocol BPR-secure under the DDH assumption.

More recently, the ultimate step for PAKE has been achieved by Katz and Vaikuntanathan [24] (KV), who proposed a *practical* one-round PAKE, where the two players just have to send simultaneous flows to each other, that depend on their own passwords only. More precisely, each flow just consists of an IND-CCA ciphertext of the password and an SPHF projection key for the correctness of the

partner's ciphertext (the word is the ciphertext and the witness consists of the random coins of the encryption). The shared secret key is eventually the product of the two hash values, as in the KOY and GL protocols.

Katz and Vaikuntanathan Smooth Projective Hash Functions. Because of the simultaneous flows, one flow cannot explicitly depend on the partner's flow, which makes impossible the use of the Gennaro and Lindell SPHF (later named GL-SPHF), in which the projection key depends on the word (the ciphertext here). On the other hand, the adversary can wait for the player to send his flow first, and then adapt its message, which requires stronger security notions than the initial Cramer and Shoup SPHF (later named CS-SPHF), in which the smoothness does not hold anymore if the word is generated after having seen the projection key. This led Katz and Vaikuntanathan to provide a new definition for SPHF (later named KV-SPHF), where the projection key depends on the hashing key only, and the smoothness holds even if the word is chosen after having seen the projection key. Variations between CS-SPHF, GL-SPHF and KV-SPHF are in the way one computes the projection key hp from the hashing key hk and the word W, but also in the smoothness property, according to the freedom the adversary has to choose W, when trying to distinguish the hash value from a random value. As a side note, while CS-SPHF is close to the initial definition, useful for converting an IND-CPA encryption scheme to IND-CCA, GL-SPHFs and KV-SPHFs did prove quite useful too: we will use KV-SPHFs for our one-round PAKE protocols and a GL-SPHF for the blind signature scheme.

As just explained, the strongest definition of SPHF, which gives a lot of freedom to the adversary, is the recent KV-SPHF. However, previous SPHFs known on Cramer-Shoup ciphertexts were GL-SPHFs only. For their one-round PAKE, Katz and Vaikuntanathan did not manage to construct such a KV-SPHF for an efficient IND-CCA encryption scheme. They then suggested to use the Naor and Yung approach [26], with an ElGamal-like encryption scheme and a *simulation-sound non-interactive zero-knowledge* (SS-NIZK) proof [28]. Such an SS-NIZK proof is quite costly in general. They suggested to use Groth-Sahai [21] proofs in bilinear groups and the linear encryption [10] which leads to a PAKE secure under the DLin assumption with a ciphertext consisting of 66 group elements and a projection key consisting of 4 group elements. As a consequence, the two players have to send 70 group elements each, which is far more costly than the KOY protocol, but it is one-round only.

More recent results on SS-NIZK proofs or IND-CCA encryption schemes, in the discrete logarithm setting, improved on that: Libert and Yung [25] proposed a more efficient SS-NIZK proof of plaintext equality in the Naor-Yung-type cryptosystem with ElGamal-like encryption. The proof can be reduced from 60 to 22 group elements and the communication complexity of the resulting PAKE is decreased to 32 group elements per user. Jutla and Roy [22] proposed relatively-sound NIZK proofs as an efficient alternative to SS-NIZK proofs to build new publicly-verifiable IND-CCA encryption schemes. They can then decrease the PAKE communication complexity to 20 group elements per user. In any case, one can remark that all one-round PAKE schemes require pairing computations.

One-Round PAKE in the Universal Composability Framework. Katz and Vaikuntanathan [24] also proposed another construction of one-round PAKE, provably secure against static corruptions in the UC framework. To achieve such a level of security, the simulator has to be more powerful: it should be able to make a successful execution after a dummy simulation, with a wrong password. To this aim, Katz and Vaikuntanathan allowed the simulator to extract the hashing key of the SPHF, to allow it to compute afterwards the hash value on any word, even outside the language. More precisely, each player additionally encrypts his hashing key to allow the key recovery by the simulator, so that the latter can compute the hash value even when a dummy password has initially been committed, whereas a success is expected. While this is the first one-round PAKE provably secure in the UC framework, hashing key recovery requires an additional quite costly simulation-sound extractable NIZK proof. Although the latter can also be improved by the above more recent work [22], the UC-secure one-round PAKE is still much more costly than the BPR-secure protocol.

Achievements. Our first contribution is the description of an instantiation of KV-SPHF on Cramer-Shoup ciphertexts, and thus the first KV-SPHF on an efficient IND-CCA encryption scheme. We thereafter use it within the above KV framework for one-round PAKE [24], in the BPR security model. Our scheme just consists of 6 group elements in each direction under the DDH assumption (4 for the ciphertext, and 2 for the projection key). This has to be compared with the 20 group elements, or more, in the best constructions discussed above, which all need pairing-friendly groups and pairing computations, or with the KOY protocol that has a similar complexity but with three sequential flows.

We also present the first GL-SPHFs/KV-SPHFs able to handle multi-exponentiation equations without requiring pairings. Those SPHFs are thus quite efficient. They lead to two applications. First, our new KV-SPHFs enable several efficient instantiations of one-round *Language-Authenticated Key-Exchange* (LAKE) protocols [5]. Our above one-round PAKE scheme is actually a particular case of a more general one-round LAKE scheme, for which we provide a BPR-like security model and a security proof. Our general constructions also cover Credential-Authenticated Key Exchange [11]. Second, thanks to a new GL-SPHF, we improve on the *blind signature* scheme presented in [9], from $5\ell + 6$ group elements in $\mathbb{G}_1$ and 1 group element in $\mathbb{G}_2$ to $3\ell + 7$ group elements in $\mathbb{G}_1$ and 1 group element in $\mathbb{G}_2$, for an ℓ-bit message to be blindly signed with a Waters signature [29]. Our protocol is round-optimal, since it consists of two flows, and leads to a classical short Waters signature.

Our second contribution is the novel extension of SPHFs, called Trapdoor SPHFs, or TSPHFs. In addition to showing that an SPHF with an encryption of the hashing key and a simulation-sound extractable NIZK proof, as used in the UC-secure one-round PAKE of Katz and Vaikuntanathan, can be seen as an inefficient TSPHF, we provide efficient instantiations of TSPHFs. To do so, we first describe a new generic framework for SPHFs that allows an easy conversion to TSPHFs. We then apply it to our above KV-SPHF on a Cramer-Shoup ciphertext. Using our new TSPHF in the UC-secure one-round PAKE framework

from [24], we obtain a scheme which consists of 11 group elements in each direction (actually, 6 group elements in $\mathbb{G}_1$ and 5 group elements in $\mathbb{G}_2$ in an asymmetric bilinear setting, using the Cramer-Shoup encryption). It is secure in the UC framework against static corruptions under the SXDH assumption with a CRS, and just twice more costly than our above BPR-secure PAKE. This is the most efficient UC-secure one-round PAKE.

Finally, while SPHFs are often used as implicit designated-verifier proofs of membership, when one wants to make them explicit, by sending the hash value, one does not get the zero-knowledge property. We then show that TSPHFs actually lead to extractable zero-knowledge (E-ZK) arguments.

Outline. In Section 2, we first revisit the different definitions for SPHFs proposed in [16, 20, 24], respectively denoted CS-SPHFs, GL-SPHFs and KV-SPHFs, and give the first instantiation of KV-SPHF on Cramer-Shoup ciphertexts. This leads to our efficient one-round PAKE provably secure in the BPR model.

We then define our novel extension of SPHFs, called Trapdoor SPHFs, or TSPHFs, in Section 3. After the presentation of a new framework for SPHFs together with a generic way to convert SPHFs into TSPHFs, we provide efficient instantiations of TSPHFs in Section 4, and especially on Cramer-Shoup ciphertexts, which lead to our efficient UC-secure one-round PAKE scheme.

We conclude with more constructions in Section 5 and other applications of both SPHFs and TSPHFs: First, thanks to the various complex languages we can handle with SPHFs, in Section 6, we present our one-round LAKE and an improved *blind signature* scheme. Finally, we provide another application of our TSPHF constructions by presenting efficient E-ZK protocols in Section 7.

Full Versions. This paper was formed by merging two Crypto 2013 submissions, both extending SPHFs with applications to PAKE protocols. The first one provided more evolved SPHFs with the BPR-secure PAKE as an application, and the second introduced TSPHFs with application to UC-secure PAKE. Because of lack of space, many details are left to the full versions of both papers that are referred to along this paper as the SPHF full version [6] and the TSPHF full version [7] respectively.

2 New SPHF on Cramer-Shoup Ciphertexts

In this section, we first recall the definitions of SPHFs and present our classification based on the dependence between words and keys. According to this classification, there are three types of SPHFs: the (almost) initial Cramer and Shoup [16] type (CS-SPHF) introduced for enhancing an IND-CPA encryption scheme to IND-CCA, the Gennaro and Lindell [20] type (GL-SPHF) introduced for PAKE, and the Katz and Vaikuntanathan [24] type (KV-SPHF) introduced for one-round PAKE.

Then, after a quick review on the Cramer-Shoup encryption scheme, we introduce our new KV-SPHF on Cramer-Shoup ciphertexts which immediately leads to a quite efficient instantiation of the Katz and Vaikuntanathan one-round PAKE [24], secure in the BPR model.

2.1 General Definition of SPHFs

Let us consider a language $\mathrm{L} \subseteq \mathcal{S}et$, and some global parameters for the SPHF, assumed to be in the common random string (CRS). The SPHF system for the language L is defined by four algorithms:

- $\mathsf{HashKG}(\mathrm{L})$ generates a hashing key hk for the language L;
- $\mathsf{ProjKG}(\mathsf{hk}, \mathrm{L}, C)$ derives the projection key hp, possibly depending on the word C;
- $\mathsf{Hash}(\mathsf{hk}, \mathrm{L}, C)$ outputs the hash value of the word C from the hashing key;
- $\mathsf{ProjHash}(\mathsf{hp}, \mathrm{L}, C, w)$ outputs the hash value of the word C from the projection key hp, and the witness w that $C \in \mathrm{L}$.

The *correctness* of the SPHF assures that if $C \in \mathrm{L}$ with w a witness of this membership, then the two ways to compute the hash values give the same result: $\mathsf{Hash}(\mathsf{hk}, \mathrm{L}, C) = \mathsf{ProjHash}(\mathsf{hp}, \mathrm{L}, C, w)$. On the other hand, the security is defined through the *smoothness*, which guarantees that, if $C \notin \mathrm{L}$, the hash value is *statistically* indistinguishable from a random element, even knowing hp.

2.2 Smoothness Adaptivity and Key Word-Dependence

This paper will exploit the very strong notion KV-SPHF. Informally, while the GL-SPHF definition allows the projection key hp to depend on the word C, the KV-SPHF definition prevents the projection key hp from depending on C, as in the original CS-SPHF definition. In addition, the smoothness should hold even if C is chosen as an arbitrary function of hp. This models the fact the adversary can see hp before deciding which word C it is interested in. More formal definitions follow, where we denote Π the range of the hash function.

CS-SPHF. This is almost[1] the initial definition of SPHF, where the projection key hp does not depend on the word C (word-independent key), but the word C cannot be chosen after having seen hp for breaking the smoothness (non-adaptive smoothness). More formally, a CS-SPHF is ε-smooth if ProjKG does not use its input C and if, for any $C \in \mathcal{S}et \backslash \mathrm{L}$, the two following distributions are ε-close:

$$\{(\mathsf{hp}, H) \mid \mathsf{hk} \xleftarrow{\$} \mathsf{HashKG}(\mathrm{L});\ \mathsf{hp} \leftarrow \mathsf{ProjKG}(\mathsf{hk}, \mathrm{L}, \bot);\ H \leftarrow \mathsf{Hash}(\mathsf{hk}, \mathrm{L}, C)\}$$
$$\{(\mathsf{hp}, H) \mid \mathsf{hk} \xleftarrow{\$} \mathsf{HashKG}(\mathrm{L});\ \mathsf{hp} \leftarrow \mathsf{ProjKG}(\mathsf{hk}, \mathrm{L}, \bot);\ H \xleftarrow{\$} \Pi\}.$$

GL-SPHF. This is a relaxation, where the projection key hp can depend on the word C (word-dependent key). More formally, a GL-SPHF is ε-smooth if, for any $C \in \mathcal{S}et \backslash \mathrm{L}$, the two following distributions are ε-close:

$$\{(\mathsf{hp}, H) \mid \mathsf{hk} \xleftarrow{\$} \mathsf{HashKG}(\mathrm{L});\ \mathsf{hp} \leftarrow \mathsf{ProjKG}(\mathsf{hk}, \mathrm{L}, C);\ H \leftarrow \mathsf{Hash}(\mathsf{hk}, \mathrm{L}, C)\}$$
$$\{(\mathsf{hp}, H) \mid \mathsf{hk} \xleftarrow{\$} \mathsf{HashKG}(\mathrm{L});\ \mathsf{hp} \leftarrow \mathsf{ProjKG}(\mathsf{hk}, \mathrm{L}, C);\ H \xleftarrow{\$} \Pi\}.$$

[1] In the initial definition, the smoothness was defined for a word C randomly chosen from $\mathcal{S}et \backslash \mathrm{L}$, and not necessarily for any such word.

KV-SPHF. This is the strongest SPHF, in which the projection key hp does not depend on the word C (word-independent key) and the smoothness holds even if C depends on hp (adaptive smoothness). More formally, a KV-SPHF is ε-smooth if ProjKG does not use its input C and, for any function f onto $\mathcal{S}et\backslash\mathrm{L}$, the two following distributions are ε-close:

$$\{(\mathsf{hp}, H) \mid \mathsf{hk} \xleftarrow{\$} \mathsf{HashKG}(\mathrm{L});\ \mathsf{hp} \leftarrow \mathsf{ProjKG}(\mathsf{hk}, \mathrm{L}, \perp);\ H \leftarrow \mathsf{Hash}(\mathsf{hk}, \mathrm{L}, f(\mathsf{hp}))\}$$

$$\{(\mathsf{hp}, H) \mid \mathsf{hk} \xleftarrow{\$} \mathsf{HashKG}(\mathrm{L});\ \mathsf{hp} \leftarrow \mathsf{ProjKG}(\mathsf{hk}, \mathrm{L}, \perp);\ H \xleftarrow{\$} \Pi\}.$$

Remark 1. One can see that a perfectly smooth (*i.e.*, 0-smooth) CS-SPHF is also a perfectly smooth KV-SPHF, since each value H has exactly the same probability to appear, and so adaptively choosing C does not increase the above statistical distance. However, as soon as a weak word C can bias the distribution, f can exploit it.

2.3 SPHFs on Languages of Ciphertexts

We could cover languages as general as those proposed in [5], but for the sake of clarity, and since the main applications need some particular cases only, we focus on SPHFs for languages of ciphertexts, whose corresponding plaintexts verify some relations. We denote these languages $\textsc{Lofc}_{\mathsf{full\text{-}aux}}$.

The parameter full-aux will parse in two parts (crs, aux): the public part crs, known in advance, and the private part aux, possibly chosen later. More concretely, crs represents the public values: it will define the encryption scheme (and will thus contain the global parameters and the public key of the encryption scheme) with the global format of both the tuple to be encrypted and the relations it should satisfy, and possibly additional public coefficients; while aux represents the private values (indeed, unless specified differently, as in Section 7, aux is assumed private): it will specify the relations, with more coefficients or constants that will remain private, and thus implicitly known by the sender and the receiver (as the expected password, for example, in PAKE protocols).

To keep aux secret, hp should not leak any information about it. We will thus restrict HashKG and ProjKG not to use the parameter aux, but just crs. This is a stronger restriction than required for our purpose, since one can use aux without leaking any information about it. But we already have quite efficient instantiations, and it makes everything much simpler to present.

2.4 SPHFs on Cramer-Shoup Ciphertexts

Labeled Cramer-Shoup Encryption Scheme (CS). We briefly review the CS labeled encryption scheme, where we combine all the public information in the encryption key. We thus have a group $\mathbb{G}$ of prime order p, with two independent generators $(g_1, g_2) \xleftarrow{\$} \mathbb{G}^2$, a hash function $\mathfrak{H}_K \xleftarrow{\$} \mathcal{H}$ from a collision-resistant hash function family onto $\mathbb{Z}_p^*$, and a reversible mapping $\mathcal{G}$ from $\{0,1\}^n$ to $\mathbb{G}$. From 5 scalars $(x_1, x_2, y_1, y_2, z) \xleftarrow{\$} {\mathbb{Z}_p}^5$, one also sets $c = g_1^{x_1} g_2^{x_2}$, $d = g_1^{y_1} g_2^{y_2}$, and $h = g_1^z$. The encryption key is $\mathsf{ek} = (\mathbb{G}, g_1, g_2, c, d, h, \mathfrak{H}_K)$, while the decryption

key is $\mathsf{dk} = (x_1, x_2, y_1, y_2, z)$. For a message $m \in \{0,1\}^n$, with $M = \mathcal{G}(m) \in \mathbb{G}$, the labeled Cramer-Shoup ciphertext is:

$$C \stackrel{\text{def}}{=} \mathsf{CS}(\ell, \mathsf{ek}, M; r) \stackrel{\text{def}}{=} (\mathbf{u} = (g_1^r, g_2^r), e = M \cdot h^r, v = (cd^\xi)^r),$$

with $\xi = \mathfrak{H}_K(\ell, \mathbf{u}, e) \in \mathbb{Z}_p^*$. If one wants to encrypt a vector of group elements $(M_1, \ldots, M_n)$, all at once in a non-malleable way, one computes all the individual ciphertexts with a common $\xi = \mathfrak{H}_K(\ell, \mathbf{u}_1, \ldots, \mathbf{u}_n, e_1, \ldots, e_n)$ for $v_1, \ldots, v_n$. Hence, everything done on tuples of ciphertexts will work on ciphertexts of vectors. In addition, the Cramer-Shoup labeled encryption scheme on vectors is IND-CCA under the DDH assumption.

The (known) GL-SPHF for CS. Gennaro and Lindell [20] proposed an SPHF on labeled Cramer-Shoup ciphertexts: the hashing key just consists of a random tuple $\mathsf{hk} = (\eta, \theta, \mu, \nu) \xleftarrow{\$} \mathbb{Z}_p^4$. The associated projection key, on a ciphertext $C = (\mathbf{u} = (u_1, u_2) = (g_1^r, g_2^r), e = \mathcal{G}(m) \cdot h^r, v = (cd^\xi)^r)$, is $\mathsf{hp} = g_1^\eta g_2^\theta h^\mu (cd^\xi)^\nu \in \mathbb{G}$. Then, one can compute the hash value in two different ways, for the language $\text{LoFC}_{\mathsf{ek},m}$ of the valid ciphertexts of $M = \mathcal{G}(m)$, where $\mathsf{crs} = \mathsf{ek}$ is public but $\mathsf{aux} = m$ is kept secret:

$$\begin{aligned} H \stackrel{\text{def}}{=} \mathsf{Hash}(\mathsf{hk}, (\mathsf{ek}, m), C) &\stackrel{\text{def}}{=} u_1^\eta u_2^\theta (e/\mathcal{G}(m))^\mu v^\nu \\ &= \mathsf{hp}^r \stackrel{\text{def}}{=} \mathsf{ProjHash}(\mathsf{hp}, (\mathsf{ek}, m), C, r) \stackrel{\text{def}}{=} H'. \end{aligned}$$

A (new) KV-SPHF for CS. We give here the description of the first known KV-SPHF on labeled Cramer-Shoup ciphertexts: the hashing key just consists of a random tuple $\mathsf{hk} = (\eta_1, \eta_2, \theta, \mu, \nu) \xleftarrow{\$} \mathbb{Z}_p^5$; the associated projection key is the pair $\mathsf{hp} = (\mathsf{hp}_1 = g_1^{\eta_1} g_2^\theta h^\mu c^\nu, \mathsf{hp}_2 = g_1^{\eta_2} d^\nu) \in \mathbb{G}^2$. Then one can compute the hash value in two different ways, for the language $\text{LoFC}_{\mathsf{ek},m}$ of the valid ciphertexts of $M = \mathcal{G}(m)$ under ek:

$$\begin{aligned} H = \mathsf{Hash}(\mathsf{hk}, (\mathsf{ek}, m), C) &\stackrel{\text{def}}{=} u_1^{(\eta_1 + \xi\eta_2)} u_2^\theta (e/\mathcal{G}(m))^\mu v^\nu \\ &= (\mathsf{hp}_1 \mathsf{hp}_2^\xi)^r \stackrel{\text{def}}{=} \mathsf{ProjHash}(\mathsf{hp}, (\mathsf{ek}, m), C, r) = H'. \end{aligned}$$

Theorem 2. *The above* SPHF *is a perfectly smooth (*i.e.*, 0-smooth)* KV-SPHF.

The proof can be found in Section 4.1 as an illustration of our new framework.

2.5 An Efficient One-Round PAKE

Review of the Katz and Vaikuntanathan PAKE. As explained earlier, Katz and Vaikuntanathan [24] recently proposed a one-round PAKE scheme. Their general framework follows Gennaro and Lindell [20] approach, which needs an SPHF on a labeled IND-CCA encryption scheme. To allow a SPHF-based PAKE scheme to be one-round, the ciphertext and the SPHF projection key for verifying the correctness of the partner's ciphertext should be sent together, before having seen the partner's ciphertext: the projection key should be independent of the ciphertext. In addition, the adversary can wait until it receives the partner's

- Players U and U' both use $\mathsf{ek} = (\mathbb{G}, g_1, g_2, c, d, h, \mathfrak{H}_K)$;
- U, with password pw, chooses $\mathsf{hk} = (\eta_1, \eta_2, \theta, \mu, \nu) \xleftarrow{\$} \mathbb{Z}_p^5$, computes $\mathsf{hp} = (\mathsf{hp}_1 = g_1^{\eta_1} g_2^{\theta} h^{\mu} c^{\nu}, \mathsf{hp}_2 = g_1^{\eta_2} d^{\nu})$, sets $\ell = (U, U', \mathsf{hp})$, and generates $C = (\mathbf{u} = (g_1^r, g_2^r), e = \mathcal{G}(\mathsf{pw}) \cdot h^r, v = (cd^{\xi})^r)$ with r a random scalar in $\mathbb{Z}_p$ and $\xi = \mathfrak{H}_K(\ell, \mathbf{u}, e)$. U sends $\mathsf{hp} \in \mathbb{G}^2$ and $C \in \mathbb{G}^4$ to U';
- Upon receiving $\mathsf{hp}' = (\mathsf{hp}'_1, \mathsf{hp}'_2) \in \mathbb{G}^2$ and $C' = (\mathbf{u}' = (u'_1, u'_2), e', v') \in \mathbb{G}^4$ from U', U sets $\ell' = (U', U, \mathsf{hp}')$ and $\xi' = \mathfrak{H}_K(\ell', \mathbf{u}', e')$ and computes
$$\mathsf{sk}_U = {u'_1}^{(\eta_1 + \xi' \eta_2)} {u'_2}^{\theta} (e'/\mathcal{G}(\mathsf{pw}))^{\mu} {v'}^{\nu} \cdot (\mathsf{hp}'_1 {\mathsf{hp}'_2}^{\xi})^r.$$

Fig. 1. One-Round PAKE based on DDH

projection key before generating the ciphertext, and thus a stronger smoothness is required. This is exactly why we need a KV-SPHF in this one-round PAKE framework.

Our Construction. Our KV-SPHF on Cramer-Shoup ciphertexts can be used in the Katz and Vaikuntanathan framework for PAKE [24]. It leads to the most efficient PAKE known so far, and it is *one-round*. Each user indeed only sends 6 elements of $\mathbb{G}$ (see Figure 1), instead of 70 elements of $\mathbb{G}$ for the Katz and Vaikuntanathan's instantiation using a Groth-Sahai SS-NIZK [21], or 20 group elements for the Jutla and Roy's [22] improvement using a relatively-sound NIZK.

The formal security result follows from the Theorem 4 in Section 6. We want to insist that our construction does not need pairing-friendly groups, and the plain DDH assumption is enough, whereas the recent constructions made heavy use of pairing-based proofs *à la* Groth-Sahai.

3 Computational Smoothness and Definition of TSPHF

In order to build a one-round PAKE provably secure in the UC framework, one needs the simulator to be able to compute the hash value even when a dummy password has initially been committed, whereas a success is expected. Katz and Vaikuntanathan [24] thus asked the players to add an encryption of the hashing key together with the projection key, and a proof a correctness (a simulation-sound extractable NIZK proof). We now improve on this technique.

More precisely, in this section, we introduce TSPHFs, which are SPHFs with a trapdoor enabling a simulator to compute the hash value on any word C without knowing hk nor any witness, but only knowing hp. TSPHFs also provide a way to ensure that hp is valid. It can be seen that, intuitively, in most cases, a TSPHF cannot be statistically smooth, and so, before introducing TSPHFs, we need to introduce a new notion of smoothness: computational smoothness.

3.1 Computationally-Smooth SPHF

Let us first suppose there exists an algorithm Setup which takes as input the security parameter $\mathfrak{K}$ and outputs a CRS crs together with a trapdoor τ, which

is not the trapdoor of the TSPHF, but just a trapdoor of crs. The trapdoor τ can be $\perp$, but in our article, the trapdoor will contain the decryption key of the encryption scheme, and possibly other data such that, for any $C \in \mathcal{S}et$, it is possible to check whether $C \in \text{LOFC}_{\mathsf{full\text{-}aux}}$ or not, in polynomial time.

Let us then consider the two games $\mathsf{Exp}_{\mathcal{A}}^{\text{smooth}-b}(\mathcal{A})$ (with $b = 0$ or 1) depicted in Figure 2, where Π denotes the set of hash values. There are two variants of the games: whether the SPHF is adaptively-smooth (KV-SPHF) or not (CS-SPHF and GL-SPHF).

Let us first explain the games for a non-adaptively-smooth SPHF. The procedure **Initialize** generates and outputs the CRS crs and its trapdoor τ. It is important to notice that computational smoothness has to hold even when the adversary knows the trapdoor, and so may depend on what is in the trapdoor τ.

During the execution of the game, the adversary is allowed to make one query $\mathbf{ProjKG}(\mathsf{aux}, C)$ to get a projection key hp associated with aux and C, and then one query $\mathbf{Hash}(\perp)$ to get the hash value of C. If $C \in \text{LOFC}_{\mathsf{full\text{-}aux}}$, smoothness does not apply, thus $\mathbf{Hash}(C)$ really returns the hash value H of C: $H = \mathsf{Hash}(\mathsf{hk}, \mathsf{full\text{-}aux}, C)$, for a hashing key hk associated with hp. Otherwise, the smoothness should apply with a real-or-random indistinguishability game, and thus, if $b = 0$ the real hash value is returned too, whereas a random value in Π is returned when $b = 1$. Eventually, the adversary ends the game by querying the **Finalize** procedure with its guess b' for b. We remark that the procedure **Hash** may or may not be polynomial time, depending on τ, since it is not necessarily possible to efficiently check whether $C \in \text{LOFC}_{\mathsf{full\text{-}aux}}$.

For the adaptively-smooth variant, the adversary does not need to provide the word C when it makes a query to **ProjKG**. It gives $\perp$ instead and can choose C adaptively after having seen hp, as input to the **Hash** query.

Formally, an SPHF is (t, ε)-smooth if for all adversary $\mathcal{A}$ running in time at most t:

$$\left|\Pr\left[\mathsf{Exp}_{\mathcal{A}}^{\text{smooth}-1}(\mathcal{A}) = 1\right] - \Pr\left[\mathsf{Exp}_{\mathcal{A}}^{\text{smooth}-0}(\mathcal{A}) = 1\right]\right| \leq \varepsilon.$$

The classical statistical-smoothness implies the (t, ε)-smoothness for any t, and any non-negligible ε (and whatever is the trapdoor τ).

3.2 Trapdoor SPHF

A TSPHF is an extension of a classical SPHF with an additional algorithm TSetup, which takes as input the CRS crs and outputs an additional CRS crs′ and a trapdoor τ' specific to crs′, which can be used to compute the hash value of words C knowing only hp. For TSPHF, we assume $\mathsf{full\text{-}aux} = (\mathsf{crs}, \mathsf{crs}', \mathsf{aux})$, although the language $\text{LOFC}_{\mathsf{full\text{-}aux}}$ still does not depend on crs′. Formally, a TSPHF is defined by seven algorithms:

- TSetup(crs) takes as input the CRS crs (generated by Setup) and generates the second CRS crs′, together with a trapdoor τ';
- HashKG, ProjKG, Hash, and ProjHash behave as for a classical SPHF;

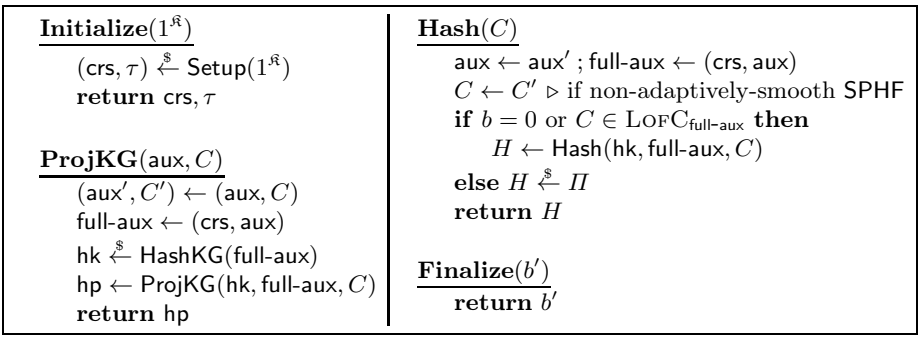


Fig. 2. Games $\mathsf{Exp}^{\text{smooth}-b}_{\mathfrak{K}}(\mathcal{A})$ ($b = 0$ or 1) for computational smoothness

- VerHP(hp, full-aux, C) outputs 1 if hp is a valid projection key, and 0 otherwise. When hp does not depend on C (word-independent key), the input C can be replaced by $\perp$;
- THash(hp, full-aux, C, τ') outputs the hash value of C from the projection key hp and the trapdoor τ'.

It must verify the following properties:

- *Correctness* is defined by two properties: *hash correctness*, which corresponds to correctness for classical SPHFs, and an additional property called *trapdoor correctness*, which states that, for any $C \in \mathcal{S}et$, if hk and hp are honestly generated, we have: VerHP(hp, full-aux, C) = 1 and Hash(hk, full-aux, C) = THash(hp, full-aux, C, τ'), with overwhelming probability;
- *Smoothness* is exactly the same as for SPHFs, except that in the **Initialize** procedure, TSetup is also called, but while τ' is dropped, crs′ is forwarded to the adversary (together with crs and τ);
- The (t, ε)-*soundness* property says that, given crs, τ and crs′, no adversary running in time at most t can produce a projection key hp, a value aux, a word C and valid witness w such that VerHP(hp, full-aux, C) = 1 but THash(hp, full-aux, C, τ') $\neq$ ProjHash(hp, full-aux, C, w), with probability at least ε. The perfect soundness states that the property holds for any t and any $\varepsilon > 0$.

It is important to notice that τ is not an input of THash and it is possible to use THash, while generating crs with an algorithm which cannot output τ (as soon as the distribution of crs output by this algorithm is indistinguishable from the one output by Setup, obviously). For example, if τ contains a decryption key, it is still possible to use the IND-CPA game for the encryption scheme, while making calls to THash.

3.3 A Naive Construction of TSPHFs Using NIZK

A naive solution to transform any SPHF into a TSPHF consists in replacing the projection key hp by a pair (hp, π), where π is an extractable NIZK (ENIZK)

proof of the knowledge of a hashing key hk such that hp is the projection key of hk. This is essentially the approach of [24]. In the TSPHF full version [7], we show that this provides a correct, smooth and sound TSPHF. Intuitively the hash correctness directly comes from the correctness of the original SPHF, the trapdoor correctness and the soundness come from the extractability of the ENIZK proof (and may not be perfect) and the smoothness comes from the zero-knowledge property of the ENIZK proof. We also show some improvements for this naive construction to make quite efficient TSPHFs, and in particular to avoid having to do bit-by-bit Groth-Sahai ENIZK proofs. These improvements can be seen as a generalization of the method proposed by Jutla and Roy in [22, Section 8]. But even with these improvements, this naive construction is still less efficient than the constructions described in the sequel.

4 Construction of DDH-Based TSPHFs

In the SPHF full version [6], we propose a formal framework for SPHFs using a new notion of graded rings, derived from [19]. It enables to deal with cyclic groups, bilinear groups (with symmetric or asymmetric pairings), or even groups with multi-linear maps. In particular, it helps to construct concrete SPHFs for quadratic pairing equations over ciphertexts, which enable to construct efficient LAKE [5] for any language handled by the Groth-Sahai NIZK proofs, and so for any NP-language (see Section 6.1).

However, we focus here on cyclic groups, with the basic intuition only, and provide some illustrations. While we keep the usual multiplicative notation for the cyclic group $\mathbb{G}$, we use an extended notation: $r \odot u = u \odot r = u^r$, for $r \in \mathbb{Z}_p$ and $u \in \mathbb{G}$, and $u \oplus v = u \cdot v$, for $u, v \in \mathbb{G}$. Basically, $\oplus$ and $\odot$ correspond to the addition and the multiplication in the exponents, that are thus both commutative. We then extend this notation in a natural way when working on vectors and matrices.

4.1 Generic Framework for GL-SPHF/KV-SPHF

Our goal is to deal with languages of ciphertexts $\textsc{Lofc}_{\mathsf{full\text{-}aux}}$: we assume that crs is fixed and we write $\mathrm{L}_{\mathsf{aux}} = \textsc{Lofc}_{\mathsf{full\text{-}aux}} \subseteq \mathcal{S}et$ where $\mathsf{full\text{-}aux} = (\mathsf{crs}, \mathsf{aux})$.

Language Representation. For a language $\mathrm{L}_{\mathsf{aux}}$, we assume there exist two positive integers k and n, a function $\Gamma : \mathcal{S}et \mapsto \mathbb{G}^{k \times n}$, and a family of functions $\Theta_{\mathsf{aux}} : \mathcal{S}et \mapsto \mathbb{G}^{1 \times n}$, such that for any word $C \in \mathcal{S}et$, $(C \in \mathrm{L}_{\mathsf{aux}}) \iff (\exists \boldsymbol{\lambda} \in \mathbb{Z}_p^{1 \times k}$ such that $\Theta_{\mathsf{aux}}(C) = \boldsymbol{\lambda} \odot \Gamma(C))$. In other words, we assume that $C \in \mathrm{L}_{\mathsf{aux}}$, if and only if, $\Theta_{\mathsf{aux}}(C)$ is a linear combination of (the exponents in) the rows of some matrix $\Gamma(C)$. For a KV-SPHF, Γ is supposed to be a constant function (independent of the word C). Otherwise, one gets a GL-SPHF.

We furthermore require that a user, who knows a witness w of the membership $C \in \mathrm{L}_{\mathsf{aux}}$, can efficiently compute the above *linear* combination $\boldsymbol{\lambda}$. This may seem a quite strong requirement but this is actually verified by very expressive languages over ciphertexts such as ElGamal, Cramer-Shoup and variants.

We briefly illustrate it on our KV-SPHF on CS: $C = (u_1 = g_1^r, u_2 = g_2^r, e = M \cdot h^r, v = (cd^\xi)^r)$, with $k = 2$, $\mathsf{aux} = M$ and $n = 5$:

$$\Gamma = \begin{pmatrix} g_1 & 1 & g_2 & h & c \\ 1 & g_1 & 1 & 1 & d \end{pmatrix} \qquad \boldsymbol{\lambda} = (r, r\xi) \qquad \begin{array}{c} \boldsymbol{\lambda} \odot \Gamma = (g_1^r, g_1^{r\xi}, g_2^r, h^r, (cd^\xi)^r) \\ \Theta_M(C) = (u_1, u_1^\xi, u_2, e/M, v). \end{array}$$

Essentially, one tries to make the first columns of $\Gamma(C)$ and the first components of $\Theta_{\mathsf{aux}}(C)$ to completely determine $\boldsymbol{\lambda}$. In our illustration, the first two columns with $u_1 = g_1^r$ and $u_1^\xi = g_1^{r\xi}$ really imply $\boldsymbol{\lambda} = (r, r\xi)$, and the three last columns help to check the language membership: we want $u_2 = g_2^r$, $e/M = h^r$, and $v = (cd^\xi)^r$, with the same r as for u_1.

Smooth Projective Hash Function. With the above notations, the hashing key is a vector $\mathsf{hk} = \boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)^\intercal \xleftarrow{\$} \mathbb{Z}_p^n$, while the projection key is, for a word C, $\mathsf{hp} = \boldsymbol{\gamma}(C) = \Gamma(C) \odot \boldsymbol{\alpha} \in \mathbb{G}^k$ (if Γ depends on C, this leads to a GL-SPHF, otherwise, one gets a KV-SPHF). Then, the hash value is:

$$\mathsf{Hash}(\mathsf{hk}, \mathsf{full\text{-}aux}, C) \stackrel{\text{def}}{=} \Theta_{\mathsf{aux}}(C) \odot \boldsymbol{\alpha} = \boldsymbol{\lambda} \odot \boldsymbol{\gamma}(C) \stackrel{\text{def}}{=} \mathsf{ProjHash}(\mathsf{hp}, \mathsf{full\text{-}aux}, C, w).$$

Our above Γ, $\boldsymbol{\lambda}$, and Θ_M immediately lead to our KV-SPHF on CS from the Section 2.4: with $\mathsf{hk} = (\eta_1, \eta_2, \theta, \mu, \nu) \xleftarrow{\$} \mathbb{Z}_p^5$, the product with Γ leads to: $\mathsf{hp} = (\mathsf{hp}_1 = g_1^{\eta_1} g_2^\theta h^\mu c^\nu, \mathsf{hp}_2 = g_1^{\eta_2} d^\nu) \in \mathbb{G}^2$, and

$$\begin{aligned} H = \mathsf{Hash}(\mathsf{hk}, (\mathsf{ek}, m), C) &\stackrel{\text{def}}{=} u_1^{(\eta_1 + \xi\eta_2)} u_2^\theta (e/\mathcal{G}(m))^\mu v^\nu \\ &= (\mathsf{hp}_1 \mathsf{hp}_2^\xi)^r \stackrel{\text{def}}{=} \mathsf{ProjHash}(\mathsf{hp}, (\mathsf{ek}, m), C, r) = H'. \end{aligned}$$

The generic framework detailed in the SPHF full version [6] also contains a security analysis that proves the above generic SPHF is perfectly smooth: Intuitively, for a word $C \notin \mathrm{L}_{\mathsf{aux}}$ and a projection key $\mathsf{hp} = \boldsymbol{\gamma}(C) = \Gamma(C) \odot \boldsymbol{\alpha}$, the vector $\Theta_{\mathsf{aux}}(C)$ is not in the linear span of $\Gamma(C)$, and thus $H = \Theta_{\mathsf{aux}}(C) \odot \boldsymbol{\alpha}$ is independent from $\Gamma(C) \odot \boldsymbol{\alpha} = \mathsf{hp}$. This also proves the Theorem 2 as a particular case.

4.2 Efficient Construction of TSPHFs under DDH

We now explain how to construct a TSPHF in a bilinear group $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$, from any SPHF constructed via the above framework, provided the SPHF does not require pairings (as all the SPHFs described in this paper), and under an additional assumption detailed later (for the smoothness to hold). To this aim, we extend our notations with $g_1 \odot g_2 = g_2 \odot g_1 = e(g_1, g_2)$, and scalars can operate on any group element as before. Intuitively, our TSPHF construction is such that all the "SPHF" part of the TSPHF is in $\mathbb{G}_1$, whereas the trapdoor part is in $\mathbb{G}_2$. And the trapdoor part simply contains some representation of $\boldsymbol{\alpha}$, representation which cannot be used without knowing the trapdoor τ'.

The second CRS is a random element $\mathsf{crs}' = \zeta \xleftarrow{\$} \mathbb{G}_2$, and its trapdoor is its discrete logarithm τ', such that $\zeta = g_2^{\tau'} = \tau' \odot g_2$. The hashing key $\mathsf{hk} = \boldsymbol{\alpha}$ is

the same as before. The projection key is the ordered pair $\mathsf{hp} = (\boldsymbol{\gamma}, \boldsymbol{\chi})$, where $\boldsymbol{\gamma}$ is the same as before, and $\boldsymbol{\chi} = \zeta \odot \boldsymbol{\alpha}$. The projection key is valid (*i.e.*, $\mathsf{VerHP}(\mathsf{hp}, \mathsf{full\text{-}aux}, C) = 1$) if and only if

$$\boldsymbol{\chi} \in \mathbb{G}_2^n \qquad \text{and} \qquad \zeta \odot \boldsymbol{\gamma} = \Gamma \odot \boldsymbol{\chi}, \tag{1}$$

Then, for any word $C \in \mathrm{L}_{\mathsf{full\text{-}aux}}$ with witness w corresponding to the vector $\boldsymbol{\lambda}$, the hash value is

$$\mathsf{Hash}(\mathsf{hk}, \mathsf{full\text{-}aux}, C) \stackrel{\text{def}}{=} \Theta(C) \odot \boldsymbol{\alpha} \odot g_2 = \boldsymbol{\lambda} \odot \boldsymbol{\gamma} \odot g_2 \stackrel{\text{def}}{=} \mathsf{ProjHash}(\mathsf{hp}, \mathsf{full\text{-}aux}, C, w).$$

Equation (1) means that $\boldsymbol{\chi}$ can be written $\boldsymbol{\chi} = \tau' \odot \boldsymbol{\alpha'}$, with $\boldsymbol{\alpha'} \in \mathbb{Z}_p^n$ verifying $\boldsymbol{\gamma} = \Gamma \odot \boldsymbol{\alpha'}$, *i.e.*, $\mathsf{hk}' = \boldsymbol{\alpha'}$ is a valid hashing key for $\boldsymbol{\gamma}$. We do not have necessarily $\boldsymbol{\alpha} = \boldsymbol{\alpha'}$, however, for any word $C \in \mathrm{L}_{\mathsf{full\text{-}aux}}$, we have and we set

$$\boldsymbol{\lambda} \odot \boldsymbol{\gamma} \odot g_2 = \Theta(C) \odot \boldsymbol{\alpha'} \odot g_2 = {\tau'}^{-1} \odot \Theta(C) \odot \boldsymbol{\chi} \stackrel{\text{def}}{=} \mathsf{THash}(\mathsf{hp}, \mathsf{full\text{-}aux}, C, \tau').$$

In the TSPHF full version [7], we prove the resulting TSPHF is computationally smooth under the DDH assumption in $\mathbb{G}_2$, if the discrete logarithms of $\Gamma_{\mathsf{aux}}(C)$ can be computed from τ. This latter assumption on $\Gamma_{\mathsf{aux}}(C)$ and τ is required for technical reasons in the proof of smoothness. The correctness and the perfect soundness are easy to prove from the construction, and so the resulting TSPHF is correct, smooth and sound.

4.3 TSPHF on Cramer-Shoup Ciphertexts

We apply this technique to extend the SPHF on Cramer-Shoup ciphertexts from Section 2 into a TSPHF. Let $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ be a bilinear group. We consider the same language and use the same notations as in Section 2 except we replace $\mathbb{G}$ by $\mathbb{G}_1$, g_1 and g_2 by $g_{1,1}$ and $g_{1,2}$ resp., and h by h_1, while g_2 is a generator of $\mathbb{G}_2$.

To get a TSPHF, we choose a random scalar τ' in $\mathbb{Z}_p$ and set $\mathsf{crs}' = \zeta = g_2^{\tau'}$. Then the hashing key, the projection key and the hash value of the TSPHF are defined as follows:

$$\mathsf{hk} = (\eta_1, \eta_2, \theta, \mu, \nu) \xleftarrow{\$} \mathbb{Z}_p^5$$

$$\mathsf{hp} = (\mathsf{hp}_1 = g_{1,1}^{\eta_1} g_{1,2}^{\theta} h_1^{\mu} c^{\nu}, \mathsf{hp}_2 = g_{1,1}^{\eta_2} d^{\nu}, \mathsf{hp}_3) \in \mathbb{G}_1^2 \times \mathbb{G}_2^5$$

$$\text{where } \mathsf{hp}_3 = (\chi_{1,1} = \zeta^{\eta_1}, \chi_{1,2} = \zeta^{\eta_2}, \chi_2 = \zeta^{\theta}, \chi_3 = \zeta^{\mu}, \chi_4 = \zeta^{\nu}) \in \mathbb{G}_2^5$$

$$\mathsf{Hash}(\mathsf{hk}, (\mathsf{ek}, m), C) = e({u_1}^{(\eta_1 + \xi\eta_2)} {u_2}^{\theta} (e/\mathcal{G}(m))^{\mu} v^{\nu}, g_2)$$

$$\mathsf{ProjHash}(\mathsf{hp}, (\mathsf{ek}, m), C, r) = e((\mathsf{hp}_1 {\mathsf{hp}_2}^{\xi})^r, g_2)$$

The projection key is valid if and only if: $e(\mathsf{hp}_1, \zeta) = e(g_{1,1}, \chi_{1,1}) \cdot e(g_{1,2}, \chi_2) \cdot e(h_1, \chi_3) \cdot e(c, \chi_4)$ and $e(\mathsf{hp}_2, \zeta) = e(g_{1,1}, \chi_{1,2}) \cdot e(d, \chi_4)$. For any $C \in \mathrm{Lof C}_{(\mathsf{crs},\mathsf{aux})}$, the hash value can be computed from C and τ' as $\mathsf{THash}(\mathsf{hp}, (\mathsf{ek}, m), C, \tau')$:

$$\Big(e(u_1, \chi_{1,1} \cdot \chi_{1,2}^{\xi}) \cdot e(u_2, \chi_2) \cdot e(e/\mathcal{G}(m), \chi_3) \cdot e(v, \chi_4)\Big)^{1/\tau'}.$$

The resulting TSPHF is smooth under the DDH in $\mathbb{G}_2$, hence the global SXDH assumption. More complex and concrete examples of TSPHFs can be found in the TSPHF full version [7].

CRS: ek and $\zeta \in \mathbb{G}_2$ Common password: pw
Both users do the same:

- U, with expected partner U', generates $\mathsf{hk} \xleftarrow{\$} \mathbb{Z}_p^5$, $\mathsf{hp} \in \mathbb{G}_1^2 \times \mathbb{G}_2^5$ and $C = \mathsf{CS}(\ell, \mathsf{ek}, \mathsf{pw}; r) \in \mathbb{G}_1^4$, where $\ell = (U, U', \mathsf{hp})$. U sends $\mathsf{hp} \in \mathbb{G}_1^2 \times \mathbb{G}_2^5$ and $C \in \mathbb{G}_1^4$
- Upon receiving $\mathsf{hp}' \in \mathbb{G}_1^2 \times \mathbb{G}_2^5$ and $C' \in \mathbb{G}_1^4$, U checks the validity of hp'. If the validity check fails, U aborts, otherwise U sets $\ell' = (U', U, \mathsf{hp}')$. U computes

$$\begin{aligned}\mathsf{sk}_U &= \mathsf{Hash}(\mathsf{hk}, (\mathsf{ek}, \mathsf{pw}), C') \cdot \mathsf{ProjHash}(\mathsf{hp}', (\mathsf{ek}, m), C, r)\\ &= e(u_1'^{(\eta_1 + \xi' \alpha_{\eta_2})} u_2'^{\theta} (e'/\mathcal{G}(\mathsf{pw}))^{\mu} v'^{\nu}, g_2) \cdot e((\mathsf{hp}_1' \mathsf{hp}_2'^{\xi})^r, g_2)\end{aligned}$$

Fig. 3. UC-Secure One-Round PAKE based on DDH

4.4 One-Round UC-Secure PAKE from TSPHF

We now show how our TSPHF can lead to a very efficient one-round PAKE, secure in the UC framework with static corruptions. This is a slight variant of the one-round PAKE from [24], where the SPHF and the SS-NIZK proofs are replaced by a TSPHF, which can be much more efficient; and where, as in our previous PAKE, we use the Cramer-Shoup encryption scheme as commitment scheme, instead of the original inefficient IND-CCA encryption scheme based on the Naor-Yung principle [26]. The concrete scheme is depicted in Figure 3. The communication complexity is of 6 elements in $\mathbb{G}_1$ and 5 elements in $\mathbb{G}_2$ only in each direction.

We show in the TSPHF full version [7] that the protocol actually works with any TSPHF on an IND-CCA encryption scheme, and provide a full generic proof. It is in the same vein as the KV's proof but a bit more intricate for two reasons: we do not assume a prior agreement of the session ID which makes our scheme a truly one-round protocol; our TSPHF does not guarantee the smoothness (even computationally) when the trapdoor τ' is known, and then, we have to modify the order of the games to use this trapdoor at the very end only.

One can remark that the original scheme in [24] can be seen as an instantiation of our scheme with the naive TSPHF based on NIZK (Section 3.3). Therefore, the security of the original KV's PAKE protocol is actually implied by our proof. And our proof also shows that their construction can be simplified by removing the commitment of hk and replacing the SS-NIZK by an ENIZK proof of knowledge of hk.

5 More Constructions of SPHFs

In this section, we first illustrate more our generic framework, by constructing more evolved SPHFs, and then we show some interesting applications. One can note that all these constructions are without pairings, the generic framework can thus be used to extend them with trapdoors, with some more applications presented in Section 7.

5.1 KV-SPHF for Linear Multi-exponentiation Equations

We present several instantiations of KV-SPHFs, in order to illustrate our framework, but also to show that our one-round PAKE protocol from Section 2.5 can be extended to one-round LAKE [5]. In PAKE/LAKE, we use SPHFs to prove that the plaintexts associated with some ElGamal-like ciphertexts verify some relations. The communication complexity of these protocols depends on the ciphertexts size and of the projection keys size. We first focus on ElGamal ciphertexts, and then explain how to handle Cramer-Shoup ciphertexts. More constructions are detailed in the SPHF full version [6].

Notations. We work in a group $\mathbb{G}$ of prime order p, generated by g, in which we assume the DDH assumption to hold. We define ElGamal encryption scheme with encryption key $\mathsf{ek} = (g, h = g^x)$. Let n, m and t be three positive integers. In the following i, j and k always range from 1 to n, from 1 to m and from 1 to t respectively in all the products $\prod_i$, $\prod_j$, $\prod_k$ and tuples $(\cdot)_i$, $(\cdot)_j$, $(\cdot)_k$. We are interested in languages of the ciphertexts $C_i = (u_i = g^{r_i}, e_i = h^{r_i} \cdot X_i)$, for which $X_1, \ldots, X_n \in \mathbb{G}$ satisfy

$$\begin{array}{r} \exists y_1, \ldots, y_m \in \mathbb{Z}_p,\ \forall k \in \{1, \ldots, t\},\ \prod_{i=1}^{n} X_i^{a_{k,i}} \cdot \prod_{j=1}^{m} A_{k,j}^{y_j} = B_k,\ \text{with} \\ \end{array} \begin{array}{l} \mathsf{crs} = (p, \mathbb{G}, \mathsf{ek}, (A_{k,j})_{k,j}) \\ \mathsf{aux} = ((a_{k,i})_{k,i}, (B_k)_k), \end{array} \quad (2)$$

where $(A_{k,j})_{k,j} \in \mathbb{G}^{t \times m}$ are public and known in advance (*i.e.*, are in crs), while $((a_{k,i})_{k,i}, (B_k)_k) \in \mathbb{Z}_p^{t \times n} \times \mathbb{G}^t$ can be kept secret (*i.e.*, can be in aux). This can be seen as a system of t linear multi-exponentiation equations.

The Groth-Sahai Approach. Naive use of the Groth Sahai framework invites us to also commit to scalars as $Y_j = g^{y_j}$ and to show that the plaintexts $(X_i)_i$ and $(Y_j)_j$ satisfy:

$$\begin{array}{c} \exists y_1, \ldots, y_m \in \mathbb{Z}_p,\ \forall k \in \{1, \ldots, t\},\ \prod_i X_i^{a_{k,i}} \cdot \prod_j A_{k,j}^{y_j} = B_k, \\ \forall j \in \{1, \ldots, m\},\ Y_j = g^{y_j}. \end{array}$$

Since there is no efficient way to extract y_j from Y_j, committing to y_j is often not useful.

A First SPHF. We thus consider the language of the ciphertexts $C_i = (u_i = g^{r_i}, e_i = h^{r_i} \cdot X_i)$, for $X_1, \ldots, X_n \in \mathbb{G}$ satisfying (2). The witnesses are $(X_i)_i$, $(r_i)_i$ and $(y_j)_j$, or just $(r_i)_i$ and $(y_j)_j$. The matrix Γ is the following one:

$$\Gamma = \left(\begin{array}{cc|cc} g & & 1 & \\ & \ddots & & \ddots \\ 1 & & g & \\ \hline & 1 & A_{1,1}^{-1} & \cdots \\ & & \vdots & \\ & & A_{1,m}^{-1} & \cdots \end{array} \right.\quad \left.\begin{array}{cc} h & & 1 & \\ & \ddots & & \ddots \\ 1 & & h & \\ \hline A_{t,1}^{-1} & & & \\ \vdots & & & \\ A_{t,m}^{-1} & & & \end{array}\right)$$

$$\begin{aligned} \Theta_{\mathsf{aux}}(\boldsymbol{C}) &= \left(\left(\textstyle\prod_i u_i^{a_{k,i}}\right)_k, \left(\textstyle\prod_i e_i^{a_{k,i}} / B_k\right)_k\right) \\ \boldsymbol{\lambda} &= \left(\left(\textstyle\sum_i a_{k,i} r_i\right)_k, (y_j)_j\right) \\ \boldsymbol{\lambda} \odot \Gamma &= \Big(\left(\textstyle\prod_i g^{a_{k,i} r_i}\right)_k, \\ &\qquad \left(\textstyle\prod_i h^{a_{k,i} r_i} / \prod_j A_{k,j}^{y_j}\right)_k\Big) \end{aligned}$$

The upper-left diagonal block imposes the first t values on $\boldsymbol{\lambda}$, while the last t columns define the t relations: The last t components of $\Theta_{\mathsf{aux}}(\boldsymbol{C})$, namely $\prod_i e_i^{a_{k,i}}/B_k = \prod_i h^{a_{k,i}r_i} \cdot \prod_i X_i^{a_{k,i}}/B_k$ (for $k = 1, \ldots, t$), are equal to the last t components of $\boldsymbol{\lambda} \odot \Gamma$, namely $\prod_i h^{a_{k,i}r_i} / \prod_j A_{k,j}^{y_j}$ ($(y_j)_j$ are just the last t components of $\boldsymbol{\lambda}$), if and only if the relations in (2) are all satisfied. It thus leads to the following KV-SPHF, with $(\mathsf{hp}_{1,k} = g^{\eta_k} h^{\mu_k})_k$ and $(\mathsf{hp}_{2,j} = \prod_k A_{k,j}^{-\mu_k})_j$, for $\mathsf{hk} = ((\eta_k)_k, (\mu_k)_k)$:

$$H = \prod_k \left(\prod_i (u_i^{\eta_k} e_i^{\mu_k})/B_k^{\mu_k} \right) = \prod_k \mathsf{hp}_{1,k}^{\sum_i a_{k,i} r_i} \cdot \prod_j \mathsf{hp}_{2,j}^{y_j} = H'.$$

As a consequence, the ciphertexts and the projection keys globally consist of $2n+t+m$ elements from $\mathbb{G}$ only. This is much more compact than the $2n+4m+t$ elements one would get by additionally committing the $(Y_j = g^{y_j})_j$.

Ciphertexts with Randomness Reuse. In some cases, even the constants $(a_{k,i})_{k,i}$ can be public and known in advance, and thus moved from aux to crs. In this case, one can furthermore shorten ElGamal ciphertexts by using multiple independent encryption keys for encrypting the X_i's: $\mathsf{ek}_i = (g, h_i = g^{x_i})$, for $i = 1, \ldots, m$. This allows to reuse the same random coins [2]. More precisely, we are now interested in the language of the ciphertexts $C = (u = g^r, (e_i = h_i^r \cdot X_i)_i)$, for $X_1, \ldots, X_n \in \mathbb{G}$ still satisfying (2). This improves on the length of the ciphertexts, from $2n$ group elements in $\mathbb{G}$ to $n+1$, and the t first rows of the matrix can be combined into the unique row $(g, \prod_i h_i^{a_{1,i}}, \ldots, \prod_i h_i^{a_{t,i}})$. It thus leads to the following KV-SPHF, with $\mathsf{hp}_1 = g^\eta \cdot \prod_k (\prod_i h_i^{a_{k,i}})^{\mu_k}$, $(\mathsf{hp}_{2,j} = \prod_k A_{k,j}^{-\mu_k})_j$, for $\mathsf{hk} = (\eta, (\mu_k)_k)$:

$$H = u^\eta \cdot \prod_k \left(\prod_i e_i^{a_{k,i}}/B_k \right)^{\mu_k} = \mathsf{hp}_1^r \cdot \prod_j \mathsf{hp}_{2,j}^{y_j} = H'.$$

Globally, the ciphertexts and the projection keys consist of $n+m+2$ elements from $\mathbb{G}$: this is independent of the number of equations. This randomness-reuse technique will be exploited in Section 6.2 for improving blind signature schemes.

From ElGamal to Cramer-Shoup Encryption. In order to move from ElGamal ciphertexts to Cramer-Shoup ciphertexts (when non-malleability is required), if one already has Γ, Θ_{aux}, and $\boldsymbol{\Lambda}$ to guarantee that the ElGamal plaintexts satisfy a relation, one simply has to make a bigger matrix, diagonal per blocks, with the block Γ and the smaller blocks $(\Gamma_k)_k$ for every ciphertext C_k, where each block Γ_k is the Cramer-Shoup matrix from Section 4.1 without the fourth column (the column with h). The initial matrix Γ guarantees the relations on the ElGamal sub-ciphertexts, and the matrices Γ_k guarantee the validity of the Cramer-Shoup ciphertexts. Since some witnesses are the same, some rows/columns can be packed together. More complex languages on Cramer-Shoup ciphertexts will be exploited in Section 6.1, we thus illustrate how the above combination can be optimized in the case of multi-exponentiation equations.

KV-SPHF on Cramer-Shoup Ciphertexts for Linear Multi-Exponentiation Equations. Let us convert the above KV-SPHF to Cramer-Shoup ciphertexts, with some optimizations: we write $C = (u_1 = g_1^r, u_2 = {g_2}^r, e_1 = h_1^r \cdot X_1, \ldots, e_n = h_n^r \cdot X_n, v = (cd^\xi)^r)$, where $\xi = \mathfrak{H}_K(u_1, u_2, e_1, \ldots, e_n) \in \mathbb{Z}_p^*$, for $X_1, \ldots, X_n \in \mathbb{G}$ satisfying (2). We make the matrix more compact as follows, with $\boldsymbol{\lambda} = (r, r\xi, (y_j)_j)$:

$$\Gamma = \left(\begin{array}{c|c|c|c|c} g_1 & 1 & g_2 & \prod_i h_i^{a_{1,i}} \ldots \prod_i h_i^{a_{t,i}} & c \\ \hline 1 & g_1 & 1 & 1 \ \ldots \ 1 & d \\ \hline 1 & 1 & 1 & A_{1,1}^{-1} \cdots A_{t,1}^{-1} & 1 \\ \vdots & \vdots & \vdots & \vdots \quad \vdots & \vdots \\ 1 & 1 & 1 & A_{1,m}^{-1} \cdots A_{t,m}^{-1} & 1 \end{array}\right).$$

This leads to the following KV-SPHF, with $\mathsf{hp}_1 = g_1^{\eta_1} {g_2}^\theta \cdot \prod_k (\prod_i h_i^{a_{k,i}})^{\mu_k} \cdot c^\nu$, $\mathsf{hp}_2 = g_1^{\eta_2} d^\nu$, $(\mathsf{hp}_{3,j} = \prod_k A_{k,j}^{-\mu_k})_j$, for $\mathsf{hk} = (\eta_1, \eta_2, \theta, (\mu_k)_k, \nu)$:

$$H = u_1^{\eta_1 + \xi\eta_2} \cdot u_2^\theta \cdot \prod_k \big(\prod_i e_i^{a_{k,i}} / B_k\big)^{\mu_k} \cdot v^\nu = (\mathsf{hp}_1 \mathsf{hp}_2^\xi)^r \cdot \prod_j \mathsf{hp}_{3,j}^{y_j} = H'.$$

5.2 GL-SPHF on Bit Encryption

Our general framework allows to construct KV-SPHFs for any language handled by the Groth-Sahai NIZK proofs (see the SPHF full version [6]). While these KV-SPHFs encompass the language of ciphertexts encrypting a bit, they require pairing evaluations. We show here a more efficient GL-SPHF for bit encryption, which does not need pairings.

Let us consider an ElGamal ciphertext $C = (u = g^r, e = h^r g^y)$, in which one wants to prove that $y \in \{0, 1\}$. We can define the following matrix that depends on C, hence a GL-SPHF:

$$\Gamma(C) = \begin{pmatrix} g & h & 1 & 1 \\ 1 & g & u & e/g \\ 1 & 1 & g & h \end{pmatrix} \qquad \begin{array}{c} \Theta_{\mathsf{aux}}(C) = (u, e, 1, 1) \qquad \boldsymbol{\lambda} = (r, y, -ry) \\ \boldsymbol{\lambda} \odot \Gamma(C) = (g^r, h^r g^y, (u/g^r)^y, (e/gh^r)^y) \end{array}$$

Because of the triangular block in $\Gamma(C)$, one sees that $\Theta_{\mathsf{aux}}(C) = \boldsymbol{\lambda} \odot \Gamma(C)$ if and only if $g^{y(y-1)} = 1$, and thus that $y \in \{0, 1\}$. With $\mathsf{hp}_1 = g^\nu h^\theta$, $\mathsf{hp}_2 = g^\theta u^\eta (e/g)^\lambda$, and $\mathsf{hp}_3 = g^\eta h^\lambda$, for $\mathsf{hk} = (\nu, \theta, \eta, \lambda)$: $H = u^\nu e^\theta = \mathsf{hp}_1^r \cdot \mathsf{hp}_2^y / \mathsf{hp}_3^{ry} = H'$.

6 More Applications of SPHFs

6.1 One-Round LAKE

Since we have shown that our framework allows to design KV-SPHFs for complex languages, we extend our PAKE protocol to LAKE [5]. To this aim, we provide a new security model, inspired from BPR [3] and a complete security proof, which implies the security of our PAKE protocol from Section 2.5.

Review of Language-Authenticated Key Exchange. LAKE is a general framework [5] that generalizes AKE primitives: each player U owns a word W in a certain language $\mathcal{L}$ and expects the other player to own a word W' in a language $\mathcal{L}'$. If everything is compatible (*i.e.*, the languages are the expected languages and the words are indeed in the appropriate languages), the players compute a common high-entropy secret key, otherwise they learn nothing about the partner's values. In any case, external eavesdroppers do not learn anything, even not the outcome of the protocol: did it succeed or not?

More precisely, we assume the two players have initially agreed on a common public part pub for the languages, but then they secretly parametrize the languages with the private parts priv: $\mathcal{L}_{\mathsf{pub},\mathsf{priv}}$ is the language they want to use, and $\mathcal{L}_{\mathsf{pub},priv'}$ is the language they assume the other player will use. In addition, each player owns a word W in his language. We will thus have to use SPHFs on ciphertexts on W, priv and $priv'$, with a common $\mathsf{crs} = (\mathsf{ek}, \mathsf{pub})$ and aux with the private parameters. For simple languages, this encompasses PAKE and Verifier-based PAKE. We refer to [5] for more applications of LAKE.

A New Security Model for LAKE. The first security model for LAKE [5] has been given in the UC framework [13], as an extension of the UC security for PAKE [14]. In this paper, we propose an extension of the PAKE security model presented by Bellare, Pointcheval, and Rogaway [3] model for LAKE: the adversary $\mathcal{A}$ plays a find-then-guess game against n players $(P_i)_{i=1,\ldots,n}$. It has access to several instances Π_U^s for each player $U \in \{P_i\}$ and can activate them (in order to model concurrent executions) via several queries: Execute-queries model passive eavesdroppings; Send-queries model active attacks; Reveal-queries model a possible bad later use of the session key; the Test-query models the secrecy of the session key. The latter query has to be asked to a *fresh* instance (which basically means that the session key is not trivially known to the adversary) and models the fact that the session key should look random for an outsider adversary.

Our extension actually differs from the original PAKE security model [3] when defining the quality of an adversary. The goal of an adversary is to distinguish the answer of the Test-query on a fresh instance: a trivial attack is the so-called on-line dictionary attack which consists in trying all the possibilities when interacting with a target player. For PAKE schemes, the advantage of such an attack is q_s/N, where q_s is the number of Send-queries and N the number of possible passwords. A secure PAKE scheme should guarantee this is the best attack, or equivalently that the advantage of any adversary is bounded by $q_s \times 2^{-m}$, where m is the min-entropy of the password distribution. In our extension, for LAKE, the trivial attack consists in trying all the possibilities for priv, $priv'$ with a word W in $\mathcal{L}_{\mathsf{pub},\mathsf{priv}}$.

Definition 3 (Security for LAKE). *A* LAKE *protocol is claimed* (t, ε)*-secure if the advantage of any adversary running in time* t *is bounded by* $q_s \times 2^{-m} \times \mathsf{Succ}^{\mathcal{L}}(t) + \varepsilon$*, where* m *is the min-entropy of the pair* $(\mathsf{priv}, \mathsf{priv}')$*, and* $\mathsf{Succ}^{\mathcal{L}}(t)$ *is the maximal success an adversary can get in finding a word in any* $\mathcal{L}_{\mathsf{pub},\mathsf{priv}}$ *within time* t*.*

- Players U and U' both use ek and agreed on pub.
- U, with $(\mathsf{priv}, priv', W)$, generates $\mathsf{hk} = \mathsf{HashKG}(\mathsf{ek}, \mathsf{pub})$ and $\mathsf{hp} = \mathsf{ProjKG}(\mathsf{hk}, (\mathsf{ek}, \mathsf{pub}), \bot)$.
  U computes $\ell = (U, U', \mathsf{hp})$ and $C = \mathsf{Encrypt}(\ell, \mathsf{ek}, (\mathsf{priv}, priv', W); r)$, with r a random scalar in $\mathbb{Z}_p$, and sends hp, C to U'.
- Upon receiving hp', C' from U', it sets $\ell' = (U', U, \mathsf{hp}')$,
  U computes $H = \mathsf{Hash}(\mathsf{hk}, ((\mathsf{ek}, \mathsf{pub}), (priv', \mathsf{priv})), (\ell', C'))$, $H' = \mathsf{ProjHash}(\mathsf{hp}', ((\mathsf{ek}, \mathsf{pub}), (\mathsf{priv}, priv')), (\ell, C), r)$, and $\mathsf{sk} = H \cdot H'$.

For $\mathsf{crs} = (\mathsf{ek}, \mathsf{pub})$ and $\mathsf{aux} = (\mathsf{priv}, priv')$,

$$\textsc{Lof}\mathrm{C}_{\mathsf{crs},\mathsf{aux}} = \left\{ (\ell, C) \;\middle|\; \begin{array}{c} \exists r, \exists W,\ C = \mathsf{Encrypt}(\ell, \mathsf{ek}, (\mathsf{priv}, priv', W); r) \\ \text{and } W \in \mathcal{L}_{\mathsf{pub},\mathsf{priv}} \end{array} \right\}.$$

Fig. 4. One-Round LAKE

Note that the min-entropy of the pair $(\mathsf{priv}, \mathsf{priv}')$ might be conditioned to the public information from the context.

Our Instantiation. Using the same approach as Katz and Vaikuntanathan for their one-round PAKE [24], one can design the scheme proposed on Figure 4, in which both users U and U' use the encryption key ek and the public part pub. This defines $\mathsf{crs} = (\mathsf{ek}, \mathsf{pub})$. When running the protocol, U owns a word W for a private part priv, and thinks about a private part $priv'$ for U', while U' owns a word W' for a private part priv', and thinks about a private $priv$ for U.

This gives a concrete instantiation of one-round LAKE as soon as one can design a KV-SPHF on the language $\textsc{Lof}\mathrm{C}_{(\mathsf{ek},\mathsf{pub}),(\mathsf{priv},priv')} = \{(\ell, C) \mid \exists r, \exists W,\ C = \mathsf{Encrypt}(\ell, \mathsf{ek}, (\mathsf{priv}, priv', W); r)$ and $W \in \mathcal{L}_{\mathsf{pub},\mathsf{priv}}\}$. More precisely, each player encrypts $(\mathsf{priv}, priv', W)$ as a vector, which thus leads to $C = (C_1, C_2, C_3)$. We then use the combination of three SPHFs: two on equality-test for the plaintexts priv (for C_1) and $priv'$ (for C_2), and one on $\textsc{Lof}\mathrm{C}_{(\mathsf{ek},\mathsf{pub}),\mathsf{priv}}$ for the ciphertext C_3 of $W \in \mathcal{L}_{\mathsf{pub},\mathsf{priv}}$.

We stress that hk and hp can depend on crs but not on aux, hence the notations used in the Figure 4. Using a similar proof as in [24], one can state the following theorem (more details on the security model and the full proof can be found in the SPHF full version [6]):

Theorem 4. *If the encryption scheme is* IND-CCA*, and* $\textsc{Lof}\mathrm{C}_{(\mathsf{ek},\mathsf{pub}),(\mathsf{priv},priv')}$ *languages admit* KV-SPHF*s, then our* LAKE *protocol is secure.*

From LAKE to PAKE. One can remark that this theorem immediately proves the security of our PAKE from Figure 1: one uses $\mathsf{priv} = \mathsf{priv}' = \mathsf{pw}$ and $\mathsf{pub} = \emptyset$, for the language of the ciphertexts of pw.

6.2 Two-Flow Waters Blind Signature

In [9], the authors presented a technique to do efficient blind signatures using an SPHF: it is still the most efficient Waters blind signature known so far. In addition, the resulting signature is a classical Waters signature.

The construction basically consists in encrypting the message bit-by-bit under distinct bases, that will allow the generation of a masked Waters hash of the message. Thereafter, the signer will easily derive a masked signature the user will eventually unmask. However, in order to generate the masked signature, the signer wants some guarantees on the ciphertexts, namely that some ciphertexts contain a bit (in order to allow extractability) and that another ciphertext contains a Diffie-Hellman value. Using our new techniques, we essentially improve on the proof of bit encryption by using the above randomness-reuse technique.

Construction. We refer the reader to [9] for the notations and to the SPHF full version [6] for details on the proof, and also for the complete construction of the GL-SPHF. Here, we give a sketch of the protocol (in which i always ranges from 1 to ℓ, except if stated otherwise) and its communication cost:

- $\mathsf{Setup}(1^{\mathfrak{K}})$, where $\mathfrak{K}$ is the security parameter, generates a pairing-friendly system $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e; g_1, g_2)$, with g_1 and g_2 generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively, a random generator $h_s \in \mathbb{G}_1$ as well as independent generators $\boldsymbol{u} = (u_i)_{i\in\{0,\ldots,\ell\}} \in \mathbb{G}_1^{\ell+1}$ for the Waters hash function $\mathcal{F}(M) = u_0 \prod_i u_i^{M_i}$, for $M = (M_i)_i \in \{0,1\}^\ell$, and finally random scalars $(x_i)_i \in \mathbb{Z}_p^\ell$. It also sets $\mathsf{ek} = (h_i)_i = (g_1^{x_i})_i$ and $g_s = \prod_i h_i$. It outputs the global parameters $\mathsf{param} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2, \mathsf{ek}, g_s, h_s, \boldsymbol{u})$. Essentially, g_1 and ek compose the encryption key for an ElGamal ciphertext on a vector, applying the randomness-reuse technique, while g_s, g_2 and h_s are the bases used for the Waters signature;
- $\mathsf{KeyGen}(\mathsf{param})$ picks at random $x \in \mathbb{Z}_p$, sets the signing key $\mathsf{sk} = h_s^x$ and the verification key $\mathsf{vk} = (g_s^x, g_2^x)$;
- $\mathsf{BSProtocol}\langle\mathcal{S}(\mathsf{sk}), \mathcal{U}(\mathsf{vk}, M)\rangle$ runs as follows, where $\mathcal{U}$ wants to get a signature on $M = (M_i)_i \in \{0,1\}^\ell$:
  - Message Encryption: $\mathcal{U}$ chooses a random $r \in \mathbb{Z}_p$ and encrypts $u_i^{M_i}$ for all the i's with the same random r: $c_0 = g_1^r$ and $(c_i = h_i^r u_i^{M_i})_i$. $\mathcal{U}$ also encrypts vk_1^r, into $d_0 = g_1^s$, $d_1 = h_1^s \mathsf{vk}_1^r$, with a different random s: It eventually sends $(c_0, (c_i)_i, (d_0, d_1)) \in \mathbb{G}_1^{\ell+3}$;
  - Signature Generation: $\mathcal{S}$ first computes the masked Waters hash of the message $c = u_0 \prod_i c_i = (\prod_i h_i)^r \mathcal{F}(M) = g_s^r \mathcal{F}(M)$, and generates the masked signature $(\sigma_1' = h_s^x c^t = h_s^x g_s^{rt} \mathcal{F}(M)^t, \sigma_2 = (g_s^t, g_2^t))$ for a random $t \xleftarrow{\$} \mathbb{Z}_p$;
  - SPHF: $\mathcal{S}$ needs the guarantee that each ElGamal ciphertext (c_0, c_i) encrypts either 1 or u_i under the key (g_1, h_i), and (d_0, d_1) encrypts the Diffie-Hellman value of $(g_1, c_0, \mathsf{vk}_1)$ under the key (g_1, h_1). The signer chooses a random $\mathsf{hk} = (\eta, (\theta_i)_i, (\nu_i)_i, \gamma, (\mu_i)_i, \lambda)$ and sets $\mathsf{hp}_1 = g_1^\eta \cdot \prod_i h_i^{\theta_i} \cdot \mathsf{vk}_1^\lambda$, $(\mathsf{hp}_{2,i} = u_i^{\theta_i} c_0^{\nu_i} (c_i/u_i)^{\mu_i})_i$, $(\mathsf{hp}_{3,i} = g_1^{\theta_i} h_i^{\mu_i})_i$, and $\mathsf{hp}_4 = g_1^\gamma h_1^\lambda$, then $H = c_0^\eta \cdot \prod_i c_i^{\theta_i} \cdot d_0^\gamma \cdot d_1^\lambda = \mathsf{hp}_1^r \cdot \prod_i \mathsf{hp}_{2,i}^{M_i} \cdot \mathsf{hp}_{3,i}^{-rM_i} \cdot \mathsf{hp}_4^s = H' \in \mathbb{G}_1$. This SPHF is easily obtained from the above GL-SPHF on bit encryption, as shown in the SPHF full version [6];
  - Masked Signature: $\mathcal{S}$ sends $(\mathsf{hp}, \Sigma = \sigma_1' \cdot H, \sigma_2) \in \mathbb{G}_1^{2\ell+3} \times \mathbb{G}_2$;

  - Signature Recovery: Upon receiving $(\mathsf{hp}, \Sigma, \sigma_2)$, using his witnesses and $\mathsf{hp}, \mathcal{U}$ computes H' and unmasks σ'_1. Thanks to the knowledge of r, it can compute $\sigma_1 = \sigma'_1 \cdot (\sigma_{2,1})^{-r}$. Note that if $H' = H$, then $\sigma_1 = h_s^x \mathcal{F}(M)^t$, which together with $\sigma_2 = (g_s^t, g_2^t)$ is a valid Waters signature on M;
- $\mathsf{Verif}(\mathsf{vk}, M, (\sigma_1, (\sigma_{2,1}, \sigma_{2,2}))$, checks whether both $e(\sigma_{2,1}, g_2) = e(g_s, \sigma_{2,2})$ and $e(\sigma_1, g_2) = e(h, \mathsf{vk}_2) \cdot e(\mathcal{F}(M), \sigma_{2,2})$ are satisfied or not.

Complexity. The whole process requires only $3\ell + 7$ elements in $\mathbb{G}_1$ ($\ell + 3$ for the ciphertexts, $2\ell + 4$ for the projection key, Σ and $\sigma_{2,1}$) and 1 in $\mathbb{G}_2$ ($\sigma_{2,2}$). This is more efficient than the instantiation from [9] ($5\ell + 6$ elements in $\mathbb{G}_1$ and 1 in $\mathbb{G}_2$) already using an SPHF, and much more efficient than the instantiation from [8] ($6\ell + 7$ elements in $\mathbb{G}_1$ and $6\ell + 5$ in $\mathbb{G}_2$) using a Groth-Sahai [21] NIZK proof.

7 Application of TSPHFs to Zero-Knowledge Arguments

In this section, we are interested in the application of SPHFs and TSPHFs to zero-knowledge arguments. Zero-knowledge arguments are used to convince a verifier that some statement or word x is in a given NP-language $\mathcal{L}$, defined by a polynomial time relation $\mathcal{R}$: $\mathcal{L} = \{x \mid \exists (w, y), \mathcal{R}(x, (w, y)) = 1\}$. This means that a word x is valid if there exists a witness (w, y) such that $\mathcal{R}(x, (w, y)) = 1$. The witness is divided in two parts (w, y): we want to prove that we know some w for which there exists y such that $\mathcal{R}(x, (w, y)) = 1$. We use the notation of [12] and write this as:

$$\text{Ж}w,\ \exists y,\ \mathcal{R}(x, (w, y)) = 1.$$

This formalism generalizes both extractable arguments of knowledge (when $y = \perp$) and non-extractable zero-knowledge arguments (when $w = \perp$). More precisely, we are interested in (partially) extractable zero-knowledge arguments (E-ZK) and extractable honest-verifier zero-knowledge arguments (HVE-ZK). E-ZK have to be *complete*, *sound*, *extractable* and *zero-knowledge*. Completeness states that an honest verifier always accepts a proof made by an honest prover for a valid statement and using a valid witness. Soundness states that no adversary can make an honest verifier accept a proof of a false statement x. Extractability states that there exists an extractor able to simulate a verifier and to output a valid partial witness w from any successful interaction with an adversary playing the role of a prover. The zero-knowledge property ensures that it is possible to simulate a prover for any true statement x even without access to a witness (w, y) for this statement x. HVE-ZK are similar to E-ZK with the difference, that for HVE-ZK, the zero-knowledge property holds only when verifiers are honest. Formal definitions can be found in the TSPHF full version [7].

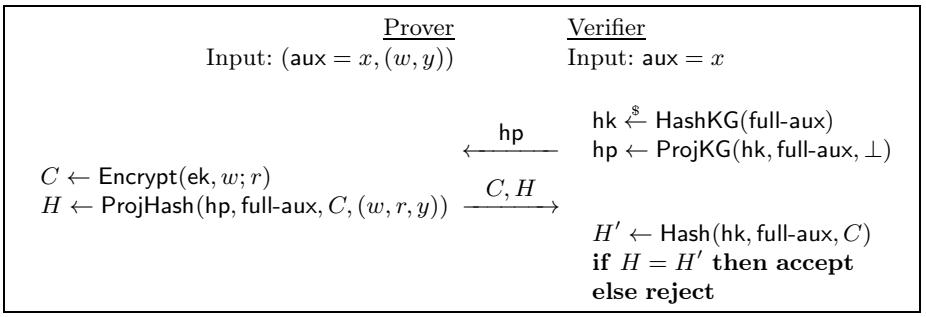


Fig. 5. Extractable Honest-Verifier Zero-Knowledge Argument from KV-SPHFs

We first show that SPHFs enable efficient constructions of HVE-ZK arguments. But these constructions are often not witness-indistinguishable, *i.e.*, a malicious prover may be able to distinguish which witness has been used by an honest prover[2], in general. Then we show that TSPHFs can overcome this limitation and we provide efficient constructions of E-ZK from TSPHFs.

7.1 Honest-Verifier Zero-Knowledge Arguments from SPHFs

The idea of the construction is that a prover, who knows some valid statement x together with a valid witness (w, y), encrypts w, using an IND-CPA encryption scheme, in some ciphertext C, under some encryption key ek contained in crs. Then, using an SPHF, he shows that the ciphertext C is an encryption of a valid partial witness w for the word x: the verifier chooses some hashing key hk and sends the corresponding projection key hp to the prover; the prover sends back the hash value H of the ciphertext C computed from hp, w, y and the random coins used in C, using ProjHash; and the verifier checks he gets the same hash value from hk, using Hash. If the SPHF is a KV-SPHF, the prover can send the ciphertext C together with H after receiving hp from the verifier. This yields a two-flow protocol. More precisely, we use a KV-SPHF for the following language:

$$\text{Lof}\mathsf{C}_{\mathsf{full\text{-}aux}} = \{C \mid \exists w,\ \exists r,\ \exists y,\ C = \mathsf{Encrypt}(\mathsf{ek}, w; r) \text{ and } \mathcal{R}(x, (w, y))\},$$

where aux is the statement x, and crs contains the encryption key ek and possibly some global parameters related to the language $\mathcal{L}$ associated with the relation $\mathcal{R}$. The complete protocol is depicted in Figure 5. In all this section, aux is public, and so it is no more required that ProjHash does not use its input aux.

It is possible to use a GL-SPHF instead of a KV-SPHF for the above language, if the ciphertext C is sent before hp. The protocol becomes three-flow but can require fewer bits to be transmitted, because GL-SPHFs are often more efficient than KV-SPHFs. Details can be found in the TSPHF full version [7].

[2] The formal definition of witness-indistinguishability can be found in the TSPHF full version [7].

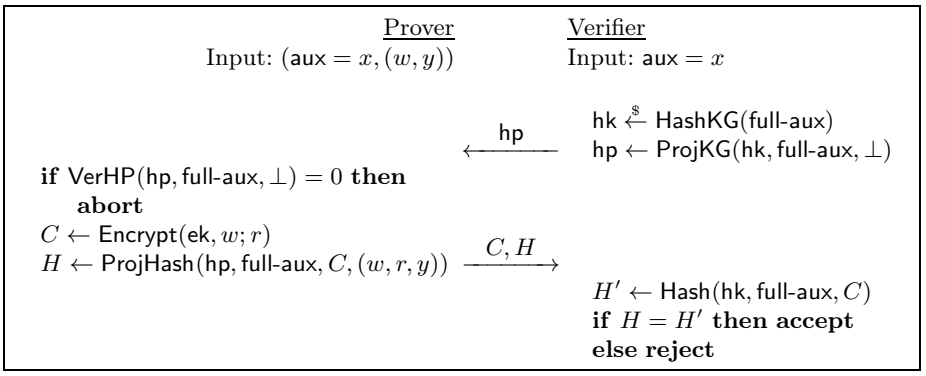


Fig. 6. Extractable Zero-Knowledge Argument from KV-TSPHFs

Completeness comes from the correctness of the SPHF and soundness comes from the statistical smoothness of the SPHF. The extractor just acts as an honest verifier and decrypts the ciphertext C of the adversarial prover at the end. The simulator for the honest-verifier zero-knowledge property just encrypts an arbitrary value in C and computes H using hk: $H = \mathsf{Hash}(\mathsf{hk}, \text{full-aux}, C)$. The IND-CPA property of the encryption scheme used for C ensures the simulator transcripts are computationally indistinguishable from real transcripts, and so the proposed construction is honest-verifier zero-knowledge.

7.2 Limitation of SPHFs

Unfortunately, without any extra property on the SPHF, the above construction is not witness indistinguishable, and so not zero-knowledge, in general. The main problem is that, for some SPHFs, it may be possible to generate hp in such a way that the hash value H computed by the prover (through ProjHash) depends on the witness used. This happens, in particular, when the language $\textsc{Lof}\mathrm{C}_{\text{full-aux}}$ of the SPHF (and also the language $\mathcal{L}$ of the HVE-ZK) is a disjunction of two languages and when the generic construction of [1] for disjunctions is used to construct the SPHF.

The previous problem does not happen for SPHFs where it is easy to distinguish valid hp from invalid ones, such as all SPHFs of this article. However, even in this case, we do not see how to prove that the resulting generic construction yields a zero-knowledge argument, because, if the simulator does not have access to hk, but only to hp, there is no trivial way to compute H.

7.3 Zero-Knowledge Arguments from TSPHFs

Let us now introduce our generic two-flow construction of E-ZK arguments from TSPHFs. The scheme is depicted in Figure 6. It is similar to the above generic construction of HVE-ZK arguments from SPHFs, except the KV-SPHF is replaced by a KV-TSPHF and the verifier aborts if the received hp is not valid.

It is also possible to use a GL-TSPHF (instead of a KV-TSPHF), at the expense of requiring three flows instead of two. In addition, if the IND-CPA encryption scheme is replaced by a labeled IND-CCA encryption scheme and the TSPHF is adapted, the resulting zero-knowledge argument becomes true-simulation extractable, which roughly means that it is possible to extract the witness of any successful proof from an adversarial prover, even if this adversarial prover has access to simulated transcripts of true statements. Details can be found in the TSPHF full version [7].

Completeness and correctness can be proven as above. For the zero-knowledge property, the simulator consists in encrypting an arbitrary value in C and computing H using hp and the trapdoor τ': $\mathsf{THash}(\mathsf{hp}, \mathsf{full\text{-}aux}, C, \tau')$. This works thanks to the IND-CPA property of the encryption scheme.

Soundness and extractability are slightly more complex to prove and require that, for any w and x, knowing τ provides a way to test whether x is valid and w is a partial witness of x, with overwhelming probability. This property is actually always verified by TSPHFs constructed as in Section 4.2. Proofs are given in the TSPHF full version [7].

7.4 Instantiations and Comparison with Ω-Protocols

Let us consider the KV-SPHFs on ElGamal ciphertexts of Section 5.1, in which possibly some of the $a_{k,i}$'s, $A_{k,j}$'s and B_k's are moved from crs to aux, which is possible here, since ProjHash is now allowed to depend on aux. If we apply the generic constructions of HVE-ZK and E-ZK to these KV-SPHFs (after transforming them to KV-TSPHFs, using the generic transformation of Section 4.2, for the E-ZK construction), we get HVE-ZK and E-ZK for languages of systems of linear multi-exponentation relations in cyclic groups:

$$\aleph(X_i)_i \in \mathbb{G}^n,\ \exists (y_j)_j \in \mathbb{Z}_p^m,\ \forall k \in \{1,\ldots,t\},\ \prod_{i=1}^{n} X_i^{a_{k,i}} \cdot \prod_{j=1}^{m} A_{k,j}^{y_j} = B_k,$$

where a statement x is a tuple containing all the constants $a_{k,i}$, $A_{k,j}$ and B_k, or some of them (in this case, the other constants are in crs).

Compared to Ω-protocols [18], which are the classical way to do a HVE-ZK, our HVE-ZK protocol from KV-SPHFs is two-flow instead of three-flow and has a lower communication complexity. Whereas our E-ZK protocol from TSPHFs is still two-flow instead of three-flow, it verifies a stronger notion of security (zero-knowledge versus honest-verifier zero-knowledge) and has just a slightly greater communication complexity.

Acknowledgments. We would like to thank Jonathan Katz for his helpful comments and anonymous referees of Crypto 2013 for their valuable inputs.

This work was supported in part by the French ANR-12-INSE-0014 SIMPATIC Project and in part by the European Commission through the FP7-ICT-2011-EU-Brazil Program under Contract 288349 SecFuNet. The second author was funded by a Sofja Kovalevskaja Award of the Alexander von Humboldt Foundation and the German Federal Ministry for Education and Research.

References

1. Abdalla, M., Chevalier, C., Pointcheval, D.: Smooth projective hashing for conditionally extractable commitments. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 671–689. Springer, Heidelberg (2009)
2. Bellare, M., Boldyreva, A., Staddon, J.: Randomness re-use in multi-recipient encryption schemeas. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 85–99. Springer, Heidelberg (2002)
3. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated key exchange secure against dictionary attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000)
4. Bellovin, S.M., Merritt, M.: Encrypted key exchange: Password-based protocols secure against dictionary attacks. In: 1992 IEEE Symposium on Security and Privacy, pp. 72–84. IEEE Computer Society Press (May 1992)
5. Ben Hamouda, F., Blazy, O., Chevalier, C., Pointcheval, D., Vergnaud, D.: Efficient UC-secure authenticated key-exchange for algebraic languages. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 272–291. Springer, Heidelberg (2013)
6. Benhamouda, F., Blazy, O., Chevalier, C., Pointcheval, D., Vergnaud, D.: New smooth projective hash functions and one-round authenticated key exchange. Cryptology ePrint Archive, Report 2013/034 (2013), `http://eprint.iacr.org/`
7. Benhamouda, F., Pointcheval, D.: Trapdoor smooth projective hash functions. Cryptology ePrint Archive, Report 2013/341 (2013), `http://eprint.iacr.org/`
8. Blazy, O., Fuchsbauer, G., Pointcheval, D., Vergnaud, D.: Signatures on randomizable ciphertexts. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 403–422. Springer, Heidelberg (2011)
9. Blazy, O., Pointcheval, D., Vergnaud, D.: Round-optimal privacy-preserving protocols with smooth projective hash functions. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 94–111. Springer, Heidelberg (2012)
10. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
11. Camenisch, J., Casati, N., Gross, T., Shoup, V.: Credential authenticated identification and key exchange. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 255–276. Springer, Heidelberg (2010)
12. Camenisch, J., Krenn, S., Shoup, V.: A framework for practical universally composable zero-knowledge protocols. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 449–467. Springer, Heidelberg (2011)
13. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: 42nd Annual Symposium on Foundations of Computer Science, pp. 136–145. IEEE Computer Society Press (October 2001)
14. Canetti, R., Halevi, S., Katz, J., Lindell, Y., MacKenzie, P.: Universally composable password-based key exchange. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 404–421. Springer, Heidelberg (2005)
15. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998)
16. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002)

17. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory 22(6), 644–654 (1976)
18. Garay, J.A., MacKenzie, P.D., Yang, K.: Strengthening zero-knowledge protocols using signatures. Journal of Cryptology 19(2), 169–209 (2006)
19. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices and applications. Cryptology ePrint Archive, Report 2012/610 (2012), http://eprint.iacr.org/
20. Gennaro, R., Lindell, Y.: A framework for password-based authenticated key exchange. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 524–543. Springer, Heidelberg (2003), http://eprint.iacr.org/2003/032.ps.gz
21. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
22. Jutla, C., Roy, A.: Relatively-sound nIZKs and password-based key-exchange. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 485–503. Springer, Heidelberg (2012)
23. Katz, J., Ostrovsky, R., Yung, M.: Efficient password-authenticated key exchange using human-memorable passwords. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 475–494. Springer, Heidelberg (2001)
24. Katz, J., Vaikuntanathan, V.: Round-optimal password-based authenticated key exchange. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 293–310. Springer, Heidelberg (2011)
25. Libert, B., Yung, M.: Non-interactive CCA-secure threshold cryptosystems with adaptive security: New framework and constructions. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 75–93. Springer, Heidelberg (2012)
26. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd Annual ACM Symposium on Theory of Computing. ACM Press (May 1990)
27. Pointcheval, D.: Password-based authenticated key exchange (invited talk). In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 390–397. Springer, Heidelberg (2012)
28. Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: 40th Annual Symposium on Foundations of Computer Science, pp. 543–553. IEEE Computer Society Press (October 1999)
29. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)

Practical Multilinear Maps over the Integers

Jean-Sébastien Coron[1], Tancrède Lepoint[2,3], and Mehdi Tibouchi[4]

[1] University of Luxembourg
jean-sebastien.coron@uni.lu
[2] CryptoExperts, France
[3] École Normale Supérieure, France
tancrede.lepoint@cryptoexperts.com
[4] NTT Secure Platform Laboratories, Japan
tibouchi.mehdi@lab.ntt.co.jp

Abstract. Extending bilinear elliptic curve pairings to multilinear maps is a long-standing open problem. The first plausible construction of such multilinear maps has recently been described by Garg, Gentry and Halevi, based on ideal lattices. In this paper we describe a different construction that works over the integers instead of ideal lattices, similar to the DGHV fully homomorphic encryption scheme. We also describe a different technique for proving the full randomization of encodings: instead of Gaussian linear sums, we apply the classical leftover hash lemma over a quotient lattice. We show that our construction is relatively practical: for reasonable security parameters a one-round 7-party Diffie-Hellman key exchange requires less than 40 seconds per party. Moreover, in contrast with previous work, multilinear analogues of useful, base group assumptions like DLIN appear to hold in our setting.

1 Introduction

Multilinear Maps. Extending bilinear elliptic curve pairings to multilinear maps is a long-standing open problem. In 2003 Boneh and Silverberg showed two interesting applications of multilinear maps [BS03], namely multipartite Diffie-Hellman and very efficient broadcast encryption; however they were pessimistic about the existence of such maps from the realm of algebraic geometry.

The first plausible construction of multilinear maps has recently been described by Garg, Gentry and Halevi, based on ideal lattices [GGH13]. The main difference with bilinear pairings is that the encoding $a \cdot g$ of an element a is randomized (with some noise) instead of deterministic; only the computed multilinear map $e(a_1 \cdot g, \ldots, a_\kappa \cdot g)$ is a deterministic function of the a_i's only. The construction has bounded degree with a maximum degree κ at most polynomial in the security parameter. Indeed, the encoding noise grows linearly with the degree, and when the noise reaches a certain threshold the encoding can become incorrect, as for ciphertexts in a somewhat homomorphic encryption scheme. The security of the construction relies on new hardness assumptions which are natural extensions of the Decisional Diffie-Hellman (DDH) assumption. To gain more confidence in their scheme the authors provide an extensive cryptanalytic

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 476–493, 2013.

survey. The authors focus on one application: the multipartite Diffie-Hellman key exchange.

The construction from [GGH13] works in the polynomial ring $R = \mathbb{Z}[X]/(X^n+1)$, where n is large enough to ensure security. One generates a secret short ring element $\boldsymbol{g} \in R$, generating a principal ideal $\mathcal{I} = \langle \boldsymbol{g} \rangle \subset R$. One also generates an integer parameter q and another random secret $\boldsymbol{z} \in R/qR$. One encodes elements of the quotient ring $R/\mathcal{I}$, namely elements of the form $\boldsymbol{e}+\mathcal{I}$ for some $\boldsymbol{e}$, as follows: a level-i encoding of the coset $\boldsymbol{e}+\mathcal{I}$ is an element of the form $u_k = [\boldsymbol{c}/\boldsymbol{z}^i]_q$, where $\boldsymbol{c} \in \boldsymbol{e}+\mathcal{I}$ is short. Such encodings can be both added and multiplied, as long as the norm of the numerators remain shorter than q; in particular the product of κ encodings at level 1 gives an encoding at level κ. For such level-κ encodings one can then define a zero-testing parameter $\boldsymbol{p}_{zt} = [\boldsymbol{h}\boldsymbol{z}^\kappa/\boldsymbol{g}]_q$, for some small $\boldsymbol{h} \in R$. Then given a level-κ encoding $\boldsymbol{u} = [\boldsymbol{c}/\boldsymbol{z}^\kappa]$ one can compute $[\boldsymbol{p}_{zt} \cdot \boldsymbol{u}]_q = [\boldsymbol{h}\boldsymbol{c}/\boldsymbol{g}]_q$. When $\boldsymbol{c}$ is an encoding of zero we have $\boldsymbol{c}/\boldsymbol{g} \in R$, which implies that $\boldsymbol{h}\boldsymbol{c}/\boldsymbol{g}$ is small in R, and therefore $[\boldsymbol{h}\boldsymbol{c}/\boldsymbol{g}]_q$ is small; this provides a way to test whether a level-κ encoding $\boldsymbol{c}$ is an encoding of 0. For the same reason the high-order bits of $[\boldsymbol{p}_{zt} \cdot \boldsymbol{u}]_q = [\boldsymbol{h}\boldsymbol{c}/\boldsymbol{g}]_q$ only depend on the coset $\boldsymbol{e}+\mathcal{I}$ and not on the particular $\boldsymbol{c} \in \boldsymbol{e}+\mathcal{I}$; this makes it possible to extract a representation of cosets encoded at level κ, and eventually defines a degree-κ multilinear map for level-1 encodings.

Our Contributions. Our main contribution is to describe a different construction that works over the integers instead of ideal lattices, similar to the DGHV fully homomorphic encryption scheme [DGHV10] and its batch variant [CCK+13]. Our construction offers the same flexibility as the original from [GGH13]; in particular it can be modified to support the analogue of asymmetric maps and composite-order maps. Moreover, it does not seem vulnerable to the "zeroizing" attack that breaks base group hardness assumptions like the analogues of DLIN and subgroup membership for the multilinear maps of [GGH13]. Since those assumptions are believed necessary to adapt constructions of primitives like adaptively secure functional encryption and NIZK, our construction seems even more promising for applications than [GGH13].

As in [GGH13], the security of our construction relies on new assumptions; it cannot be derived from "classical" assumptions such as the Approximate-GCD assumption used in [DGHV10]. We describe various possible attacks against our scheme; this enables us to derive parameters for which our scheme remains secure against these attacks.

Our new construction works as follows: one first generates n secret primes p_i and publishes $x_0 = \prod_{i=1}^n p_i$ (where n is large enough to ensure correctness and security); one also generates n small secret primes g_i, and a random secret integer z modulo x_0. A level-k encoding of a vector $\boldsymbol{m} = (m_i) \in \mathbb{Z}^n$ is then an integer c such that for all $1 \leqslant i \leqslant n$:

$$c \equiv \frac{r_i \cdot g_i + m_i}{z^k} \pmod{p_i} \tag{1}$$

for some small random integers r_i; the integer c is therefore defined modulo x_0 by CRT. It is clear that such encodings can be both added and multiplied

modulo x_0, as long as the numerators remain smaller than the p_i's. In particular the product of κ encodings c_j at level 1 gives an encoding at level κ where the corresponding vectors $\boldsymbol{m}_j$ are multiplied componentwise. For such level-κ encodings one defines a zero-testing parameter p_{zt} with:

$$p_{zt} = \sum_{i=1}^{n} h_i \cdot \left(z^\kappa \cdot g_i^{-1} \bmod p_i\right) \cdot \prod_{i' \neq i} p_{i'} \bmod x_0$$

for some small integers h_i. Given a level-κ encoding c as in (1), one can compute $\omega = p_{zt} \cdot c \bmod x_0$, which gives:

$$\omega = \sum_{i=1}^{n} h_i \cdot \left(r_i + m_i \cdot (g_i^{-1} \bmod p_i)\right) \cdot \prod_{i' \neq i} p_{i'} \bmod x_0 \ .$$

Then if $m_i = 0$ for all i, since the r_i's and h_i's are small, we obtain that ω is small compared to x_0; this enables to test whether c is an encoding of $\mathbf{0}$ or not. Moreover for non-zero encodings the leading bits of ω only depend on the m_i's and not on the noise r_i; for level-κ encodings this enables to extract a function of the m_i's only, which eventually defines as in [GGH13] a degree-κ multilinear map.[1]

Our second contribution is to describe a different technique for proving the full randomization of encodings. As in [GGH13] the randomization of encodings is obtained by adding a random subset-sum of encodings of $\mathbf{0}$ from the public parameters. However as in [GGH13] the Leftover Hash Lemma (LHL) cannot be directly applied since the encodings live in some infinite ring instead of a finite group. The solution in [GGH13] consists in using linear sums with Gaussian coefficients; it is shown in [AGHS12] that the resulting sum has a Gaussian distribution (over some lattice). As noted by the authors, this can be seen as a "leftover hash lemma over lattices". In this paper we describe a different technique that does not use Gaussian coefficients; instead it consists in working modulo some lattice $L \subset \mathbb{Z}^n$ and applying the leftover hash lemma over the quotient $\mathbb{Z}^n/L$, which is still a finite group. Such technique was already used to prove the security of the batch variant of the DGHV fully homomorphic encryption scheme [CCK+13, CLT13a]. Here we provide a more formal description: we clearly state our "LHL over lattices" so that it can later be applied as a black-box (as the corresponding Theorem 3 in [AGHS12]). Our quotient lattice technique can independently be applied to the original encoding scheme from [GGH13], while the Gaussian sum technique from [AGHS12] is also applicable to ours.

Our third contribution is to describe the first implementation of multilinear maps. It appears that the basic versions of both [GGH13] and our scheme are rather unpractical, because of the huge public parameter size required to randomize the encodings. Therefore we use a simple optimization that consists in storing only a small subset of the public elements and combining them pairwise to generate the full public-key. Such optimization was originally described in [GH11] for

[1] Technically for p_{zt} we use a *vector* of integers instead of a single integer (see Sec. 3).

reducing the size of the encryption of the secret-key bits in the implementation of Gentry's FHE scheme [Gen09]. It was also used in [CMNT11] to reduce the public-key size of the DGHV scheme; however, as opposed to the latter work our randomization of encodings is heuristic only, whereas in [CMNT11] the semantic security was still guaranteed. Thanks to this optimization our construction becomes relatively practical: for reasonable security parameters a multipartite Diffie-Hellman computation with 7 users requires less than 40 seconds, with a public parameter size of roughly 2.6 GBytes; a proof-of-concept implementation is available at [Lep].

2 Definition of Randomized Encodings and Multilinear Maps

In this section we recall the setting introduced in [GGH13] for the notion of randomized encodings and multilinear maps, which they call *graded encoding schemes*. There are essentially two main differences with classical bilinear pairings (and their generalization to cryptographic multilinear maps as considered in [BS03]):

1. In bilinear pairings (and more generally cryptographic multilinear maps) we have a map $e\colon G^\kappa \to G_T$ that is linear with respect to all its κ inputs:

$$e(a_1 \cdot g, \ldots, a_\kappa \cdot g) = \left(\prod_{i=1}^{\kappa} a_i\right) \cdot e(g, \ldots, g)\ . \tag{2}$$

   One can view $a \cdot g$ as an "encoding" of the integer $a \in \mathbb{Z}_p$ over the group G of order p generated by g. The main difference in our setting is that encodings are now randomized. This means that an element $a \in R$ (where R is a ring that plays the role of the exponent space $\mathbb{Z}_p$) has many possible encodings; only the final multilinear map $e(a_1 \cdot g, \ldots, a_\kappa \cdot g)$ is a deterministic function of the a_i's only, and not on the randomness used to encode a_i into $a_i \cdot g$.
2. The second main difference is that to every encoding is now associated a *level*. At level 0 we have the "plaintext" ring elements $a \in R$, at level 1 we have the encoding $a \cdot g$, and by combining k such encodings $a_i \cdot g$ at level 1 one obtains a level-k encoding where the underlying elements a_i are homomorphically multiplied in R. The difference with "classical" cryptographic multilinear maps is that we can now multiply any (bounded) subset of encodings, instead of strictly κ at a time as with (2). For encodings at level κ we have a special zero-testing parameter $\boldsymbol{p}_{zt}$ that can extract a deterministic function of the underlying ring elements. This enables to define a degree-κ multilinear map for encodings at level 1.

2.1 Graded Encoding System

We recall the formal definition of a κ-Graded Encoding System from [GGH13]. For simplicity we only consider the symmetric case throughout the paper; we

refer to [GGH13] for a more general framework that can handle the asymmetric case.

Definition 1 (κ-Graded Encoding System [GGH13]). *A κ-Graded Encoding System for a ring R is a system of sets $\mathcal{S} = \{S_v^{(\alpha)} \in \{0,1\}^* : v \in \mathbb{N}, \alpha \in R\}$, with the following properties:*

1. *For every $v \in \mathbb{N}$, the sets $\{S_v^{(\alpha)} : \alpha \in R\}$ are disjoint.*
2. *There are binary operations $+$ and $-$ (on $\{0,1\}^*$) such that for every $\alpha_1, \alpha_2 \in R$, every $v \in \mathbb{N}$, and every $u_1 \in S_v^{(a_1)}$ and $u_2 \in S_v^{(a_2)}$, it holds that $u_1 + u_2 \in S_v^{(\alpha_1+\alpha_2)}$ and $u_1 - u_2 \in S_v^{(\alpha_1-\alpha_2)}$ where $\alpha_1 + \alpha_2$ and $\alpha_1 - \alpha_2$ are addition and subtraction in R.*
3. *There is an associative binary operation $\times$ (on $\{0,1\}^*$) such that for every $\alpha_1, \alpha_2 \in R$, every v_1, v_2 with $0 \leqslant v_1 + v_2 \leqslant \kappa$, and every $u_1 \in S_{v_1}^{(\alpha_1)}$ and $u_2 \in S_{v_2}^{(\alpha_2)}$, it holds that $u_1 \times u_2 \in S_{v_1+v_2}^{(\alpha_1 \cdot \alpha_2)}$ where $\alpha_1 \cdot \alpha_2$ is multiplication in R.*

2.2 Multilinear Map Procedures

We also recall the definition of the procedures for manipulating encodings. As previously we consider only the symmetric case; we refer to [GGH13] for the general case.

Instance Generation. The randomized $\mathsf{InstGen}(1^\lambda, 1^\kappa)$ takes as inputs the parameters λ and κ, and outputs $(\mathsf{params}, \boldsymbol{p}_{zt})$, where params is a description of a κ-Graded Encoding System as above, and $\boldsymbol{p}_{zt}$ is a zero-test parameter.

Ring Sampler. The randomized $\mathsf{samp}(\mathsf{params})$ outputs a "level-zero encoding" $a \in S_0^{(\alpha)}$ for a nearly uniform element $\alpha \in_R R$. Note that the encoding a does not need to be uniform in $S_0^{(\alpha)}$.

Encoding. The (possibly randomized) $\mathsf{enc}(\mathsf{params}, a)$ takes as input a level-zero encoding $a \in S_0^{(\alpha)}$ for some $\alpha \in R$, and outputs the level-one encoding $u \in S_1^{(\alpha)}$ for the same α.

Re-randomization. The randomized $\mathsf{reRand}(\mathsf{params}, i, u)$ re-randomizes encodings relative to the same level i. Specifically, given an encoding $u \in S_v^{(\alpha)}$, it outputs another encoding $u' \in S_v^{(\alpha)}$. Moreover for any two $u_1,\ u_2 \in S_v^{(\alpha)}$, the output distributions of $\mathsf{reRand}(\mathsf{params}, i, u_1)$ and $\mathsf{reRand}(\mathsf{params}, i, u_2)$ are nearly the same.

Addition and Negation. Given params and two encodings relative to the same level, $u_1 \in S_i^{(\alpha_1)}$ and $u_2 \in S_i^{(\alpha_2)}$, we have $\mathsf{add}(\mathsf{params}, u_1, u_2) \in S_i^{(\alpha_1+\alpha_2)}$ and $\mathsf{neg}(\mathsf{params}, u_1) \in S_i^{(-\alpha_1)}$. Below we write $u_1 + u_2$ and $-u_1$ as a shorthand for applying these procedures.

Multiplication. For $u_1 \in S_i^{(\alpha_1)}$ and $u_2 \in S_j^{(\alpha_2)}$, we have $\mathsf{mul}(\mathsf{params}, u_1, u_2) = u_1 \times u_2 \in S_{i+j}^{(\alpha_1 \cdot \alpha_2)}$.

Zero-Test. The procedure $\mathsf{isZero}(\mathsf{params}, \boldsymbol{p}_{zt}, u)$ outputs 1 if $u \in S_\kappa^{(0)}$ and 0 otherwise.

Extraction. The procedure extracts a random function of ring elements from their level-κ encoding. Namely $\mathsf{ext}(\mathsf{params}, \boldsymbol{p}_{zt}, u)$ outputs $s \in \{0,1\}^\lambda$, such that:

1. For any $\alpha \in R$ and $u_1,\ u_2 \in S_\kappa^{(\alpha)}$, $\mathsf{ext}(\mathsf{params}, \boldsymbol{p}_{zt}, u_1) = \mathsf{ext}(\mathsf{params}, \boldsymbol{p}_{zt}, u_2)$.
2. The distribution $\{\mathsf{ext}(\mathsf{params}, \boldsymbol{p}_{zt}, u) : \alpha \in_R R, u \in S_\kappa^{(\alpha)}\}$ is nearly uniform over $\{0,1\}^\lambda$.

This concludes the definition of the procedures. In [GGH13] the authors consider a slightly relaxed definition of isZero and ext, where isZero can still output 1 even for some non-zero encoding u with negligible probability, and ext can extract different outputs when applied to encodings of the same elements, also with negligible probability; see [GGH13] for the corresponding definitions.

2.3 Hardness Assumptions

Finally we recall the hardness assumptions for multilinear maps from [GGH13]; as previously we consider only the symmetric case and refer to [GGH13] for the general case. In this symmetric case given a set of $\kappa + 1$ level-one encodings of random elements, it should be unfeasible to distinguish a level-κ encoding of their product from random.

Graded DDH (GDDH). Let $\mathcal{GE}$ be a graded encoding scheme consisting of all the routines above. For an adversary $\mathcal{A}$ and parameters λ, κ we consider the following process:

1. $(\mathsf{params}, \boldsymbol{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^\kappa)$
2. Choose $a_j \leftarrow \mathsf{samp}(\mathsf{params})$ for all $1 \leqslant j \leqslant \kappa + 1$
3. Set $u_j \leftarrow \mathsf{reRand}(\mathsf{params}, 1, \mathsf{enc}(\mathsf{params}, 1, a_j))$ for all $1 \leqslant j \leqslant \kappa + 1$ // encodings at level 1
4. Choose $b \leftarrow \mathsf{samp}(\mathsf{params})$ // encoding at level 0
5. Set $\tilde{u} = a_{\kappa+1} \times \prod_{i=1}^{\kappa} u_i$ // encoding of the right product at level κ
6. Set $\hat{u} = b \times \prod_{i=1}^{\kappa} u_i$ // encoding of a random product at level κ

The GDDH distinguisher is given as input the $\kappa + 1$ level-one encodings u_j and either $\tilde{u}$ (encoding the right product) or $\hat{u}$ (encoding a random product), and must decide which is the case. The GDDH assumption states that the advantage of any efficient adversary is negligible in the security parameter.

3 Our New Encoding Scheme

System Parameters. The main parameters are the security parameter λ and the required multilinearity level $\kappa \leqslant \mathsf{poly}(\lambda)$. Based on λ and κ, we choose the vector dimension n, the bit-size η of the primes p_i, the bit-size α of the primes g_i,

the maximum bit-size ρ of the randomness used in encodings, and various other parameters that will be specified later; the constraints that these parameters must satisfy are described in the next section. For integers z, p we denote the reduction of z modulo p by $(z \bmod p)$ or $[z]_p$ with $-p/2 < [z]_p \leqslant p/2$.

In our scheme a level-k encoding of a vector $\boldsymbol{m} = (m_i) \in \mathbb{Z}^n$ is an integer c such that for all $1 \leqslant i \leqslant n$:

$$c \equiv \frac{r_i \cdot g_i + m_i}{z^k} \pmod{p_i} \tag{3}$$

where the r_i's are ρ-bit random integers (specific to the encoding c), with the following secret parameters: the p_i's are η-bit prime integers, the g_i's are α-bit primes, and the denominator z is a random (invertible) integer modulo $x_0 = \prod_{i=1}^{n} p_i$. The integer c is therefore defined by CRT modulo x_0, where x_0 is made public. Since the p_i's must remain secret, the user cannot encode the vectors $\boldsymbol{m} \in \mathbb{Z}^n$ by CRT directly from (3); instead one includes in the public parameters a set of ℓ level-0 encodings x'_j of random vectors $\boldsymbol{a}_j \in \mathbb{Z}^n$, and the user can generate a random level-0 encoding by computing a random subset-sum of those x'_j's.

Remark 1. From (3) each integer m_i is actually defined modulo g_i. Therefore our scheme encodes vectors $\boldsymbol{m}$ from the ring $R = \mathbb{Z}_{g_1} \times \cdots \times \mathbb{Z}_{g_n}$.

Instance Generation: $(\mathsf{params}, \boldsymbol{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^\kappa)$. We generate n secret random η-bit primes p_i and publish $x_0 = \prod_{i=1}^{n} p_i$. We generate a random invertible integer z modulo x_0. We generate n random α-bit prime integers g_i and a secret matrix $\boldsymbol{A} = (a_{ij}) \in \mathbb{Z}^{n \times \ell}$, where each component a_{ij} is randomly generated in $[0, g_i) \cap \mathbb{Z}$. We generate an integer y, three sets of integers $\{x_j\}_{j=1}^{\tau}$, $\{x'_j\}_{j=1}^{\ell}$ and $\{\Pi_j\}_{j=1}^{n}$, a zero-testing vector $\boldsymbol{p}_{zt}$, and a seed s for a strong randomness extractor, as described later. We publish the parameters $\mathsf{params} = \big(n, \eta, \alpha, \rho, \beta, \tau, \ell, y, \{x_j\}_{j=1}^{\tau}, \{x'_j\}_{j=1}^{\ell}, \{\Pi_j\}_{j=1}^{n}, s\big)$ and $\boldsymbol{p}_{zt}$.

Sampling Level-Zero Encodings: $c \leftarrow \mathsf{samp}(\mathsf{params})$. We publish as part as our instance generation a set of ℓ integers x'_j, where each x'_j encodes at level-0 the column vector $\boldsymbol{a}_j \in \mathbb{Z}^n$ of the secret matrix $\boldsymbol{A} = (a_{ij}) \in \mathbb{Z}^{n \times \ell}$. More precisely, using the CRT modulo x_0 we generate integers x'_j such that:

$$1 \leqslant j \leqslant \ell, \qquad x'_j \equiv r'_{ij} \cdot g_i + a_{ij} \pmod{p_i} \tag{4}$$

where the r'_{ij}'s are randomly generated in $(-2^\rho, 2^\rho) \cap \mathbb{Z}$.

Our randomized sampling algorithm $\mathsf{samp}(\mathsf{params})$ works as follows: we generate a random binary vector $\boldsymbol{b} = (b_j) \in \{0,1\}^\ell$ and output the level-0 encoding

$$c = \sum_{j=1}^{\ell} b_j \cdot x'_j \bmod x_0 \ .$$

From Equation (4), this gives $c \equiv \left(\sum_{j=1}^{\ell} r'_{ij} b_j\right) \cdot g_i + \sum_{j=1}^{\ell} a_{ij} b_j \pmod{p_i}$; as required the output c is a level-0 encoding:

$$c \equiv r_i \cdot g_i + m_i \pmod{p_i} \tag{5}$$

of some vector $\boldsymbol{m} = \boldsymbol{A} \cdot \boldsymbol{b} \in \mathbb{Z}^n$ which is a random subset-sum of the column vectors $\boldsymbol{a}_j$. We note that for such level-0 encodings we get $|r_i \cdot g_i + m_i| \leqslant \ell \cdot 2^{\rho+\alpha}$ for all i.

The following Lemma states that, as required, the distribution of $\boldsymbol{m}$ can be made statistically close to uniform over $R = \mathbb{Z}_{g_1} \times \cdots \times \mathbb{Z}_{g_n}$; the proof is based on applying the LHL over the set R (see the full version of this paper [CLT13b]).

Lemma 1. *Let* $c \leftarrow \mathsf{samp}(\mathsf{params})$ *and write* $c \equiv r_i \cdot g_i + m_i \pmod{p_i}$. *Assume* $\ell \geqslant n \cdot \alpha + 2\lambda$. *The distribution of* $(\mathsf{params}, \boldsymbol{m})$ *is statistically close to the distribution of* $(\mathsf{params}, \boldsymbol{m}')$ *where* $\boldsymbol{m}' \leftarrow R$.

Encodings at Higher Levels: $c_k \leftarrow \mathsf{enc}(\mathsf{params}, k, c)$. To allow encoding at higher levels, we publish as part of our instance-generation a level-one random encoding of $\mathbf{1}$, namely an integer y such that:

$$y \equiv \frac{r_i \cdot g_i + 1}{z} \pmod{p_i}$$

for random integers $r_i \in (-2^{\rho}, 2^{\rho}) \cap \mathbb{Z}$; as previously the integer y is computed by CRT modulo x_0.

Given a level-0 encoding c of $\boldsymbol{m} \in \mathbb{Z}^n$ as given by (5), we can then compute a level-1 encoding of the same $\boldsymbol{m}$ by computing $c_1 = c \cdot y \bmod x_0$. Namely we obtain as required:

$$c_1 \equiv \frac{r_i^{(1)} \cdot g_i + m_i}{z} \pmod{p_i} \tag{6}$$

for some integers $r_i^{(1)}$, and we get $|r_i^{(1)} \cdot g_i + m_i| \leqslant \ell \cdot 2^{2(\rho+\alpha)}$ for all i. More generally to generate a level-k encoding we compute $c_k = c_0 \cdot y^k \bmod x_0$.

In multipartite Diffie-Hellman key-exchange every user keeps a private level-0 encoding c and publishes a level-1 encoding of the same underlying (unknown) $\boldsymbol{m}$; however one cannot publish $c_1 = c \cdot y \bmod x_0$ directly since the private level-0 encoding c could be recovered immediately from $c = c_1/y \bmod x_0$. Instead the level-1 encoding c_1 must first be re-randomized into a new level-1 encoding c'_1 whose distribution does not depend on the original c as long as it encodes the same $\boldsymbol{m}$.

Re-randomization: $c' \leftarrow \mathsf{reRand}(\mathsf{params}, i, c)$. To allow re-randomization of encodings at level one, we publish as part of our instance-generation a set of n integers Π_j which are all level-1 random encodings of zero:

$$1 \leqslant j \leqslant n, \qquad \Pi_j \equiv \frac{\varpi_{ij} \cdot g_i}{z} \pmod{p_i}\ .$$

The matrix $\boldsymbol{\Pi} = (\varpi_{ij}) \in \mathbb{Z}^{n\times n}$ is a diagonally dominant matrix generated as follows: the non-diagonal entries are randomly and independently generated in $(-2^\rho, 2^\rho) \cap \mathbb{Z}$, while the diagonal entries are randomly generated in $(n2^\rho, n2^\rho + 2^\rho) \cap \mathbb{Z}$.

We also publish as part of our instance-generation a set of τ integers x_j, where each x_j is a level-1 random encoding of zero:

$$1 \leqslant j \leqslant \tau, \qquad x_j \equiv \frac{r_{ij} \cdot g_i}{z} \pmod{p_i}$$

and where the column vectors of the matrix $(r_{ij}) \in \mathbb{Z}^{n\times\tau}$ are randomly and independently generated in the half-open parallelepiped spanned by the columns of the previous matrix $\boldsymbol{\Pi}$; see the full version of this paper [CLT13b] for a concrete algorithm.

Given as input a level-1 encoding c_1 as given by (6), we randomize c_1 with a random subset-sum of the x_j's and a linear combination of the Π_j's:

$$c_1' = c_1 + \sum_{j=1}^{\tau} b_j \cdot x_j + \sum_{j=1}^{n} b_j' \cdot \Pi_j \bmod x_0 \tag{7}$$

where $b_j \leftarrow \{0,1\}$, and $b_j' \leftarrow [0, 2^\mu) \cap \mathbb{Z}$. The following Lemma shows that as required the distribution of c_1' is nearly independent of the input (as long as it encodes the same $\boldsymbol{m}$). This follows essentially from our "leftover hash lemma over lattices"; see Section 4.

Lemma 2. *Let the encodings* $c \leftarrow \mathsf{samp}(\mathsf{params})$, $c_1 \leftarrow \mathsf{enc}(\mathsf{params}, 1, c)$, *and* $c_1' \leftarrow \mathsf{reRand}(\mathsf{params}, 1, c_1)$. *Write* $c_1' \equiv (r_i \cdot g_i + m_i)/z \pmod{p_i}$ *for all* $1 \leqslant i \leqslant n$, *and* $\boldsymbol{r} = (r_1, \ldots, r_n)^T$. *Let* $\tau \geqslant n \cdot (\rho + \log_2(2n)) + 2\lambda$ *and* $\mu \geqslant \rho + \alpha + \lambda$. *The distribution of* $(\mathsf{params}, \boldsymbol{r})$ *is statistically close to that of* $(\mathsf{params}, \boldsymbol{r'})$, *where* $\boldsymbol{r'} \in \mathbb{Z}^n$ *is randomly generated in the half-open parallelepiped spanned by the column vectors of* $2^\mu \boldsymbol{\Pi}$.

Writing $c_1' \equiv (r_i' \cdot g_i + m_i)/z \pmod{p_i}$, and using $|r_{ij} \cdot g_i| \leqslant 2n2^{\rho+\alpha}$ for all i, j, we obtain $|r_i' \cdot g_i + m_i| \leqslant \ell 2^{2(\rho+\alpha)} + \tau \cdot 2n2^{\rho+\alpha} + n \cdot 2n2^{\mu+\rho+\alpha}$. Using $\mu \geqslant \rho+\alpha+\lambda$ this gives $|r_i' \cdot g_i + m_i| \leqslant 4n^2 2^{\mu+\rho+\alpha}$.

Adding and Multiplying Encodings. It is clear that one can homomorphically add encodings. Moreover the product of κ level-1 encodings u_i can be interpreted as an encoding of the product. Namely, given level-one encodings u_j of vectors $\boldsymbol{m}_j \in \mathbb{Z}^n$ for $1 \leqslant j \leqslant \kappa$, with $u_j \equiv (r_{ij} \cdot g_i + m_{ij})/z \pmod{p_i}$, we simply let:

$$u = \prod_{j=1}^{\kappa} u_j \bmod x_0 \ .$$

This gives:

$$u \equiv \frac{\prod_{j=1}^{\kappa} (r_{ij} \cdot g_i + m_{ij})}{z^\kappa} \equiv \frac{r_i \cdot g_i + \left(\prod_{j=1}^{\kappa} m_{ij}\right) \bmod g_i}{z^\kappa} \pmod{p_i}$$

for some $r_i \in \mathbb{Z}$. This is a level-κ encoding of the vector $\boldsymbol{m}$ obtained by componentwise product of the vectors $\boldsymbol{m}_j$, as long as $\prod_{j=1}^{\kappa}(r_{ij} \cdot g_i + m_{ij}) < p_i$ for all i. When computing the product of κ level-1 encodings from reRand and one level-0 encoding from samp (as in multipartite Diffie-Hellman key exchange), we obtain from previous bounds $|r_i| \leqslant (4n^2 2^{\mu+\rho+\alpha})^{\kappa} \cdot \ell \cdot 2^{\rho+1}$ for all i.

Zero Testing. $\mathsf{isZero}(\mathsf{params}, \boldsymbol{p}_{zt}, u_\kappa) \stackrel{?}{=} 0/1$. We can test equality between encodings by subtracting them and testing for zero. To enable zero testing we randomly generate an integer matrix $\boldsymbol{H} = (h_{ij}) \in \mathbb{Z}^{n \times n}$ such that $\boldsymbol{H}$ is invertible in $\mathbb{Z}$ and both $\|\boldsymbol{H}^T\|_\infty \leqslant 2^\beta$ and $\|(\boldsymbol{H}^{-1})^T\|_\infty \leqslant 2^\beta$, for some parameter β specified later; here $\|\cdot\|_\infty$ is the operator norm on $n \times n$ matrices with respect to the ℓ^∞ norm on $\mathbb{R}^n$. A technique for generating such $\boldsymbol{H}$ is discussed in the full version of this paper [CLT13b]. We then publish as part of our instance generation the following zero-testing vector $\boldsymbol{p}_{zt} \in \mathbb{Z}^n$:

$$(\boldsymbol{p}_{zt})_j = \sum_{i=1}^{n} h_{ij} \cdot \left(z^\kappa \cdot g_i^{-1} \bmod p_i\right) \cdot \prod_{i' \neq i} p_{i'} \bmod x_0 \,. \tag{8}$$

To determine whether a level-κ encoding c is an encoding of zero or not, we compute the vector $\boldsymbol{\omega} = c \cdot \boldsymbol{p}_{zt} \bmod x_0$ and test whether $\|\boldsymbol{\omega}\|_\infty$ is small:

$$\mathsf{isZero}(\mathsf{params}, \boldsymbol{p}_{zt}, c) = \begin{cases} 1 & \text{if } \|c \cdot \boldsymbol{p}_{zt} \bmod x_0\|_\infty < x_0 \cdot 2^{-\nu} \\ 0 & \text{otherwise} \end{cases}$$

for some parameter ν specified later.

Namely for a level-κ ciphertext c we have $c \equiv (r_i \cdot g_i + m_i)/z^\kappa \pmod{p_i}$ for some $r_i \in \mathbb{Z}$; therefore for all $1 \leqslant i \leqslant n$ we can write:

$$c = q_i \cdot p_i + (r_i \cdot g_i + m_i) \cdot \left(z^{-\kappa} \bmod p_i\right) \tag{9}$$

for some $q_i \in \mathbb{Z}$. Therefore combining (8) and (9), we get:

$$(\boldsymbol{\omega})_j = (c \cdot \boldsymbol{p}_{zt} \bmod x_0)_j = \sum_{i=1}^{n} h_{ij} \cdot \left(r_i + m_i \cdot (g_i^{-1} \bmod p_i)\right) \cdot \prod_{i' \neq i} p_{i'} \bmod x_0 \,. \tag{10}$$

Therefore if $m_i = 0$ for all $1 \leqslant i \leqslant n$, then $\|\boldsymbol{\omega}\|_\infty = \|c \cdot \boldsymbol{p}_{zt} \bmod x_0\|_\infty$ is small compared to x_0 when the r_i's are small enough, *i.e.* a limited number of additions/multiplications on encodings has been performed. Conversely if $m_i \neq 0$ for some i we show that $\|\boldsymbol{\omega}\|_\infty$ must be large. More precisely we prove the following lemma in the full version of this paper [CLT13b].

Lemma 3. *Let n, η, α and β be as in our parameter setting. Let ρ_f be such that $\rho_f + \lambda + \alpha + 2\beta \leqslant \eta - 8$, and let $\nu = \eta - \beta - \rho_f - \lambda - 3 \geqslant \alpha + \beta + 5$. Let c be such that $c \equiv (r_i \cdot g_i + m_i)/z^\kappa \pmod{p_i}$ for all $1 \leqslant i \leqslant n$, where $0 \leqslant m_i < g_i$ for all i. Let $\boldsymbol{r} = (r_i)_{1 \leqslant i \leqslant n}$ and assume that $\|\boldsymbol{r}\|_\infty < 2^{\rho_f}$. If $\boldsymbol{m} = 0$ then $\|\boldsymbol{\omega}\|_\infty < x_0 \cdot 2^{-\nu-\lambda-2}$. Conversely if $\boldsymbol{m} \neq 0$ then $\|\boldsymbol{\omega}\|_\infty \geqslant x_0 \cdot 2^{-\nu+2}$.*

Extraction. $sk \leftarrow \mathsf{ext}(\mathsf{params}, \boldsymbol{p}_{zt}, u_\kappa)$. This part is essentially the same as in [GGH13]. To extract a random function of the vector $\boldsymbol{m}$ encoded in a level-κ encoding c, we multiply c by the zero-testing parameter $\boldsymbol{p}_{zt}$ modulo x_0, collect the ν most significant bits of each of the n components of the resulting vector, and apply a strong randomness extractor (using the seed s from params):

$$\mathsf{ext}(\mathsf{params}, \boldsymbol{p}_{zt}, c) = \mathsf{Extract}_s\big(\mathsf{msbs}_\nu(c \cdot \boldsymbol{p}_{zt} \bmod x_0)\big)$$

where msbs_ν extracts the ν most significant bits of the result. Namely if two encodings c and c' encode the same $\boldsymbol{m} \in \mathbb{Z}^n$ then from Lemma 3 we have $\|(c - c') \cdot \boldsymbol{p}_{zt} \bmod x_0\|_\infty < x_0 \cdot 2^{-\nu-\lambda-2}$, and therefore we expect that $\boldsymbol{\omega} = c \cdot \boldsymbol{p}_{zt} \bmod x_0$ and $\boldsymbol{\omega'} = c' \cdot \boldsymbol{p}_{zt} \bmod x_0$ agree on their ν most significant bits, and therefore extract to the same value.[2]

Conversely if c and c' encode different vectors then by Lemma 3 we must have $\|(c - c') \cdot \boldsymbol{p}_{zt} \bmod x_0\|_\infty > x_0 \cdot 2^{-\nu+2}$, and therefore the ν most significant bits of the corresponding $\boldsymbol{\omega}$ and $\boldsymbol{\omega'}$ must be different. This implies that for random $\boldsymbol{m} \in R = \mathbb{Z}_{g_1} \times \cdots \times \mathbb{Z}_{g_n}$ the min-entropy of $\mathsf{msbs}_\nu(c \cdot \boldsymbol{p}_{zt} \bmod x_0)$ when c encodes $\boldsymbol{m}$ is at least $\log_2 |R| \geqslant n(\alpha - 1)$. Therefore we can use a strong randomness extractor to extract a nearly uniform bit-string of length $\lfloor \log_2 |R| \rfloor - \lambda$.

3.1 Setting the Parameters

The system parameters must satisfy the following constraints:

- The bit-size ρ of the randomness used for encodings must satisfy $\rho = \Omega(\lambda)$ to avoid brute force attack on the noise, including the improved attack from [CN12] with complexity $\tilde{\mathcal{O}}(2^{\rho/2})$.
- The bit-size α of the primes g_i must be large enough so that the order of the group $R = \mathbb{Z}_{g_1} \times \cdots \times \mathbb{Z}_{g_n}$ does not contain small prime factors; this is required to prove the security of the multipartite Diffie-Hellman Key Exchange (see the full version of this paper [CLT13b]). One can take $\alpha = \lambda$.
- The parameter n must be large enough to thwart lattice-based attacks on the encodings, namely $n = \omega(\eta \log \lambda)$; see Section 5.1.
- The number ℓ of level-0 encodings x'_j for samp must satisfy $\ell \geqslant n \cdot \alpha + 2\lambda$ in order to apply the leftover hash lemma; see Lemma 1.
- The number τ of level-1 encodings x_j must satisfy $\tau \geqslant n \cdot (\rho + \log_2(2n)) + 2\lambda$ in order to apply our "leftover hash lemma over lattices". For the same reason the bit-size μ of the linear sum coefficients must satisfy $\mu \geqslant \alpha + \rho + \lambda$; see Lemma 2.

[2] Two coefficients ω and ω' from $\boldsymbol{\omega}$ and $\boldsymbol{\omega'}$ could still be on the opposite sides of a boundary, with $\lfloor \omega / 2^k \rfloor = v$ and $\lfloor \omega' / 2^k \rfloor = v + 1$, so that ω and ω' would extract to different MSBs v and $v+1$. Heuristically this happens with probability $\mathcal{O}(2^{-\lambda})$. The argument can be made rigorous by generating a public random integer W modulo x_0 in the parameters, and extracting the MSBs of $\omega + W \bmod x_0$ instead of $\omega \bmod x_0$ for all coefficients ω of the vector $\boldsymbol{\omega}$.

- The bitsize β of the matrix $\boldsymbol{H}$ entries must satisfy $\beta = \Omega(\lambda)$ in order to avoid the GCD attack from Section 5.2. One can take $\beta = \lambda$.
- The bit-size η of the primes p_i must satisfy $\eta \geqslant \rho_f + \alpha + 2\beta + \lambda + 8$, where ρ_f is the maximum bit size of the randoms r_i a level-κ encoding (see Lemma 3). When computing the product of κ level-1 encodings and an additional level-0 encoding (as in a multipartite Diffie-Hellman key exchange with $N = \kappa + 1$ users), one obtains $\rho_f = \kappa \cdot (\mu + \rho + \alpha + 2\log_2 n + 2) + \rho + \log_2 \ell + 1$ (see previous Section).
- The number ν of most significant bits to extract can then be set to $\nu = \eta - \beta - \rho_f - \lambda - 3$ (see Lemma 3).

3.2 Security of Our Construction

As in [GGH13] the security of our construction relies on new assumptions that do not seem to be reducible to more classical assumptions. Namely, as in [GGH13] one can make the assumption that solving the Graded DDH problem (GDDH) recalled in Section 2.3 is hard in our scheme. This enables to prove the security of the one-round N-way Diffie-Hellman key exchange protocol [GGH13]. Ideally one would like to reduce such assumption to a more classical assumption, such as the Approximate-GCD assumption, but that does not seem feasible. Therefore to gain more confidence in our scheme we describe various attacks in Section 5.

4 Another Leftover Hash Lemma over Lattices

As mentioned in the introduction, to prove the full randomization of encodings (Lemma 2) one cannot apply the classical Leftover Hash Lemma (LHL) because the noise in the encodings live in some infinite ring instead of a finite group. In [GGH13] the issue was solved by using linear sums with Gaussian coefficients. Namely the analysis in [AGHS12] shows that the resulting sum has a Gaussian distribution (over some lattice). As noted by the authors this technique can be seen as a "leftover hash lemma over lattices". Such a technique would be applicable to our scheme as well.

In this section we describe an alternative technique (without Gaussian coefficients) that can also be seen as a "leftover hash lemma over lattices". It consists in working modulo a lattice $L \subset \mathbb{Z}^n$ and applying the classical leftover hash lemma over the finite group $\mathbb{Z}^n/L$. This technique was already used in [CCK+13, CLT13a] to prove the security of a batch extension of the DGHV scheme. In this paper we provide a more formal description; namely we clearly state our "LHL over lattices" so that it can later be applied as a black-box (as the corresponding Theorem 3 in [AGHS12]). Namely our quotient lattice technique can independently be applied to the original encoding scheme from [GGH13].

4.1 Classical Leftover Hash Lemma

We first recall the classical Leftover Hash Lemma. We say that the distributions D_1, D_2 over a finite domain X are ε-statistically close if the statistical distance

$\Delta(D_1, D_2) = \frac{1}{2}\sum_{x\in X} |D_1(x) - D_2(x)|$ is smaller than ε. A distribution D is ε-uniform if its statistical distance from the uniform distribution is at most ε. A family $\mathcal{H}$ of hash functions from X to Y, both finite sets, is said to be pairwise-independent if for all distinct $x, x' \in X$, $\Pr_{h\leftarrow H}[h(x) = h(x')] = 1/|Y|$.

Lemma 4 (Leftover Hash Lemma [HILL99]). *Let $\mathcal{H}$ be a family of pairwise hash functions from X to Y. Suppose that $h \leftarrow \mathcal{H}$ and $x \leftarrow X$ are chosen uniformly and independently. Then, $(h, h(x))$ is $\frac{1}{2}\sqrt{|Y|/|X|}$-uniform over $\mathcal{H}\times Y$.*

One can then deduce the following Lemma for random subset sums over a finite abelian group.

Lemma 5. *Let G be a finite abelian group. Set $x_1, \ldots, x_m \leftarrow G$ uniformly and independently, set $s_1, \ldots, s_m \leftarrow \{0, 1\}$, and set $y = \sum_{i=1}^m s_i x_i \in G$. Then $(x_1, \ldots, x_m, y)$ is $1/2\sqrt{|G|/2^m}$-uniform over G^{m+1}.*

Proof. We consider the following hash function family $\mathcal{H}$ from $\{0, 1\}^m$ to G. Each member $h \in \mathcal{H}$ is parameterized by the elements $(x_1, \ldots, x_m) \in G^m$. Given $\boldsymbol{s} \in \{0, 1\}^m$, we define $h(\boldsymbol{s}) = \sum_{i=1}^m s_i \cdot x_i \in G$. The hash function family is clearly pairwise independent. Therefore by Lemma 4, $(h, h(x))$ is $\frac{1}{2}\sqrt{|G|/2^m}$-uniform over G^{m+1}. □

4.2 Leftover Hash Lemma over Lattices

Let $L \subset \mathbb{Z}^n$ be a lattice of rank n of basis $\boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$. Then every $\boldsymbol{x} \subset \mathbb{Z}^n$ can be uniquely written as:

$$\boldsymbol{x} = \xi_1\boldsymbol{b}_1 + \ldots + \xi_n\boldsymbol{b}_n$$

for some real numbers ξ_i. Moreover, for every vector $\boldsymbol{x} \in \mathbb{Z}^n$ there is a unique $\boldsymbol{a} \in L$ such that:

$$\boldsymbol{y} = \boldsymbol{x} - \boldsymbol{a} = \xi'_1\boldsymbol{b}_1 + \ldots + \xi'_n\boldsymbol{b}_n$$

where $0 \leqslant \xi'_i < 1$; we write $\boldsymbol{y} = \boldsymbol{x} \bmod \boldsymbol{B}$. Therefore each vector of $\mathbb{Z}^n/L$ has a unique representative in the half-open parallelepiped defined by the previous equation.

We denote by $\mathcal{D}_{\boldsymbol{B}}$ the distribution obtained by generating a random element in the quotient $\mathbb{Z}^n/L$ and taking its unique representative in the half-open parallelepiped generated by the basis $\boldsymbol{B}$. Given a basis $\boldsymbol{B} = (\boldsymbol{b_1}, \ldots, \boldsymbol{b}_n)$ and $\mu \in \mathbb{Z}^*$ we denote by $\mu\boldsymbol{B}$ the basis $(\mu\boldsymbol{b}_1, \ldots, \mu\boldsymbol{b}_n)$. We are now ready to state our "Leftover Hash Lemma over Lattices".

Lemma 6. *Let $L \subset \mathbb{Z}^n$ be a lattice of rank n of basis $\boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$. Let $\boldsymbol{x}_i$ for $1 \leqslant i \leqslant m$ be generated independently according to the distribution $\mathcal{D}_{\boldsymbol{B}}$. Set $s_1, \ldots, s_m \leftarrow \{0, 1\}$ and $t_1, \ldots, t_n \leftarrow \mathbb{Z} \cap [0, 2^\mu)$. Let $\boldsymbol{y} = \sum_{i=1}^m s_i\boldsymbol{x}_i + \sum_{i=1}^n t_i\boldsymbol{b_i}$ and $\boldsymbol{y'} \leftarrow \mathcal{D}_{2^\mu\boldsymbol{B}}$. Then the distributions $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y})$ and $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y'})$ are ε-statistically close, with $\varepsilon = mn \cdot 2^{-\mu} + 1/2\sqrt{|\det L|/2^m}$.*

Proof. We consider the intermediate variable:

$$\boldsymbol{y''} = \left(\sum_{i=1}^{m} s_i \boldsymbol{x}_i \bmod \boldsymbol{B} \right) + \sum_{i=1}^{n} t_i \boldsymbol{b_i} \ . \tag{11}$$

Firstly by applying the leftover hash lemma over the finite abelian group $G = \mathbb{Z}^n/L$, we obtain that the distributions $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \sum_{i=1}^{m} s_i \boldsymbol{x}_i \bmod \boldsymbol{B})$ and $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{\psi})$ are ε_1-statistically close, where $\boldsymbol{\psi} \leftarrow \mathcal{D}_{\boldsymbol{B}}$ and

$$\varepsilon_1 = 1/2\sqrt{|G|/2^m} = 1/2\sqrt{|\det(L)|/2^m} \ .$$

This implies that the distributions $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y''})$ and $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y'})$ are also ε_1-statistically close.

Secondly we write:

$$\sum_{i=1}^{m} s_i \boldsymbol{x}_i \bmod \boldsymbol{B} = \sum_{i=1}^{m} s_i \boldsymbol{x}_i - \sum_{j=1}^{n} \chi_j \boldsymbol{b}_j \tag{12}$$

where $\chi_j \in \mathbb{Z}$ for all j. We also write $\boldsymbol{x_i} = \sum_j \xi_{ij} \boldsymbol{b}_j$ where by definition $0 \leqslant \xi_{ij} < 1$ for all i, j. This gives:

$$\sum_{i=1}^{m} s_i \boldsymbol{x}_i \bmod \boldsymbol{B} = \sum_{i=1}^{m} s_i \sum_{j=1}^{n} \xi_{ij} \boldsymbol{b}_j - \sum_{j=1}^{n} \chi_j \boldsymbol{b}_j = \sum_{j=1}^{n} \left(\sum_{i=1}^{m} s_i \xi_{ij} - \chi_j \right) \boldsymbol{b}_j \ ,$$

which implies $0 \leqslant \sum_{i=1}^{m} s_i \xi_{ij} - \chi_j < 1$ for all j, and therefore $0 \leqslant \chi_j \leqslant m$ for all j. Combining Equations (11) and (12) we have:

$$\boldsymbol{y''} = \sum_{i=1}^{m} s_i \boldsymbol{x}_i + \sum_{i=1}^{n} (t_i - \chi_i) \boldsymbol{b_i} \ ,$$

where as shown above $0 \leqslant \chi_i \leqslant m$ for all i. This implies that the distributions $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y})$ and $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y''})$ are ε_2-statistically close, with $\varepsilon_2 = mn2^{-\mu}$. Therefore the distributions $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y})$ and $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y'})$ are $(\varepsilon_1 + \varepsilon_2)$-statistically close, which proves the Lemma. □

We also show that the previous distribution $\mathcal{D}_{2^\mu \boldsymbol{B}}$ is not significantly modified when a small vector $\boldsymbol{z} \in \mathbb{Z}^n$ is added and the operator norm of the corresponding matrix B^{-1} is upper-bounded; see the full version of this paper [CLT13b] for the proof.

Lemma 7. *Let $L \subset \mathbb{Z}^n$ be a full-rank lattice of basis $\boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$, and let $B \in \mathbb{Z}^{n \times n}$ be the matrix whose column vectors are the $\boldsymbol{b}_i$'s. For any $\boldsymbol{z} \in \mathbb{Z}^n$, the distribution of $\boldsymbol{z} + \mathcal{D}_{2^\mu \boldsymbol{B}}$ is ε-statistically close to that of $\mathcal{D}_{2^\mu \boldsymbol{B}}$, where $\varepsilon = 2^{-\mu} \cdot (\|\boldsymbol{z}\|_\infty \cdot \|B^{-1}\|_\infty + 1)$.*

4.3 Re-randomization of Encodings: Proof of Lemma 2

We are now ready to apply our "LHL over lattices" to prove the full randomization of encodings as stated in Lemma 2. Namely the re-randomization equation (7) can be rewritten in vector form as:

$$\boldsymbol{r}' = \boldsymbol{r} + \boldsymbol{X} \cdot \boldsymbol{b} + \boldsymbol{\Pi} \cdot \boldsymbol{b}'$$

where $\boldsymbol{b} \leftarrow \{0,1\}^\tau$ and $\boldsymbol{b'} \leftarrow ([0, 2^\mu) \cap \mathbb{Z})^n$, and the columns of the matrix $\boldsymbol{X} \in \mathbb{Z}^{n \times \tau}$ are uniformly and independently generated in the parallelepiped spanned by the columns of the matrix $\boldsymbol{\Pi} \in \mathbb{Z}^{n \times n}$. To conclude, it therefore suffices to apply Lemma 6 and Lemma 7, using additionally an upper bound on $\|\boldsymbol{\Pi}^{-1}\|_\infty$. For this we use the fact that $\boldsymbol{\Pi}$ has been generated as a diagonally dominant matrix. We refer to the full version of this paper [CLT13b] for the full proof of Lemma 2.

5 Attacks against Our Multilinear Scheme

5.1 Lattice Attack on the Encodings

We first describe a lattice attack against level-0 encodings. We consider an element $x_0 = \prod_{i=1}^n p_i$ and a set of τ integers $x_j \in \mathbb{Z}_{x_0}$ such that:

$$x_j \bmod p_i = r_{ij}$$

where $r_{ij} \in (-2^\rho, 2^\rho) \cap \mathbb{Z}$. We want to estimate the complexity of the classical orthogonal lattice attack for recovering (some of) the noise values r_{ij}.

This attack works by considering the integer vector formed by a subset of the x_j's, say $\boldsymbol{x} = (x_j)_{1 \leqslant j \leqslant t}$ for some $n < t \leqslant \tau$, and relating the lattice of vectors orthogonal to $\boldsymbol{x} \bmod x_0$ to the lattice of vectors orthogonal to each of the corresponding noise value vectors $\boldsymbol{r}_i = (r_{ij})_{1 \leqslant j \leqslant t}$.

More precisely, let $L \subset \mathbb{Z}^t$ the lattice of vectors $\boldsymbol{u}$ such that:

$$\boldsymbol{u} \cdot \boldsymbol{x} \equiv 0 \pmod{x_0}.$$

Clearly, L contains $x_0\mathbb{Z}^t$ so it is of full rank t. Moreover, we have

$$\mathrm{vol}(L) = [\mathbb{Z}^t : L] = x_0 / \gcd(x_0, x_1, \ldots, x_t) = x_0 \ .$$

As a result, we heuristically expect the successive minima of L to be around $\mathrm{vol}(L)^{1/t} \approx 2^{n \cdot \eta / t}$, and hence applying lattice reduction should yield a reduced basis $(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_t)$ with vectors of length $\|\boldsymbol{u}_k\| \approx 2^{n \cdot \eta / t + \alpha t}$ for some constant $\alpha > 0$ depending on the lattice reduction algorithm ($2^{\alpha t}$ is the Hermite factor).

Now, a vector $\boldsymbol{u} \in L$ satisfies $\boldsymbol{u} \cdot \boldsymbol{x} \equiv 0 \pmod{x_0}$, so for each $i \in \{1, \ldots, n\}$, reducing modulo p_i gives:

$$\boldsymbol{u} \cdot \boldsymbol{r}_i \equiv 0 \pmod{p_i}.$$

In particular, if $\boldsymbol{u}$ is short enough to satisfy $\|\boldsymbol{u}\| \cdot \|\boldsymbol{r}_i\| < p_i$, this implies $\boldsymbol{u} \cdot \boldsymbol{r}_i = 0$ in $\mathbb{Z}$. As a result, if we have:

$$2^{n \cdot \eta / t + \alpha t} \cdot 2^{\rho} < 2^{\eta}, \tag{13}$$

we expect the vectors $(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{t-n})$ from the previous lattice reduction step to be orthogonal to the $\boldsymbol{r}_i$'s, and hence computing the rank n orthogonal lattice to the lattice spanned by those vectors should reveal the $\boldsymbol{r}_i$'s.

Since t must be greater than n for the attack to apply, condition (13) implies in particular that:

$$\alpha < \frac{\eta - \rho}{n}.$$

Since a Hermite factor of $2^{\alpha t}$ is achieved in time $2^{\Omega(1/\alpha)}$ (usually by carrying out BKZ reduction with block size $\beta = \omega(1/\alpha)$, in which each block is BKZ-reduced in time exponential in β, see e.g. [HPS11]), we obtain that this orthogonal lattice attack has a complexity exponential in n. In fact, with $\gamma = \eta \cdot n$, we get a time complexity of $2^{\Omega(\gamma/\eta^2)}$, similar to [DGHV10, §5.2] (see also [CH12]).

5.2 GCD Attack on the Zero-Testing Parameter

We consider the ratio modulo x_0 of two coefficients from the zero-testing vector $\boldsymbol{p}_{zt}$, namely $u := (\boldsymbol{p}_{zt})_1 / (\boldsymbol{p}_{zt})_2 \bmod x_0$. From (8) we obtain for all $1 \leqslant i \leqslant \ell$:

$$u \equiv h_{i1}/h_{i2} \pmod{p_i}$$

We can therefore recover p_i by computing $\gcd(h_{i2} \cdot u - h_{i1}, x_0)$ for all possible h_{i1}, h_{i2}. Since the h_{ij}'s are upper bounded by 2^{β} in absolute value, the attack has complexity $\mathcal{O}(2^{2\beta})$. By using a technique similar to [CN12], the attack complexity can be reduced to $\tilde{\mathcal{O}}(2^{\beta})$.

We describe more attacks in the full version of this paper [CLT13b].

6 Optimizations and Implementation

In this section we describe an implementation of our scheme in the one-round N-way Diffie-Hellman key exchange protocol; we recall the protocol in the full version of this paper [CLT13b], as described in [BS03, GGH13].

We note that without optimizations the size of the public parameters makes our scheme completely unpractical; this is also the case in [GGH13]. Namely, for sampling we need to store at least $n \cdot \alpha$ encodings (resp. $n \cdot \rho$ encodings for re-randomization), each of size $n \cdot \eta$ bits; the public-key size is then at least $n^2 \cdot \eta \cdot \alpha$ bits. With $n \simeq 10^4$, $\eta \simeq 10^3$ and $\alpha \simeq 80$, the public-key size would be at least 1 TB. Therefore we use three heuristic optimizations to reduce the memory requirement; we refer to the full version of this paper [CLT13b] for a detailed description.

1. Non-uniform sampling: for the sampling algorithm we use a small number of encodings ℓ only; this implies that the sampling cannot be proved uniform anymore.
2. Quadratic re-randomization: we only store a small subset of encodings which are later combined pairwise to generate the full set of encodings. This implies that the randomization of encodings becomes heuristic only.
3. Integer p_{zt}: we use a single integer p_{zt} instead of a vector $\boldsymbol{p}_{zt}$ with n components. An encoding c of zero still gives a small integer $\omega = p_{zt} \cdot c \bmod x_0$, but the converse does not necessarily hold anymore.

We have implemented a one-round N-way Diffie-Hellman key exchange protocol with $N = 7$ users, in C++ using the Gnu MP library [Gt13] to perform operations on large integers. We refer to the full version of this paper [CLT13b] for a description of the protocol. We provide our concrete parameters and the resulting timings in Table 1, for security parameters ranging from 52 to 80 bits.

Table 1. Parameters and timings to instantiate a *one-round 7-way Diffie-Hellman key exchange protocol* with $\ell = 160$, $\beta = 80$, $\alpha = 80$, $N = 7$ (*i.e.* $\kappa = 6$) and $\nu = 160$ on a 16-core computer (Intel(R) Xeon(R) CPU E7-8837 at 2.67GHz) using GMP 5.1.1. Note that the Setup was parallelized on the 16 cores to speed-up the process while the other steps ran on a single core.

Instantiation	λ	n	η	Δ	ρ	pk size
Small	52	540	1838	23	41	24 MB
Medium	62	2085	2043	45	56	129 MB
Large	72	8250	2261	90	72	709 MB
Extra	80	26115	2438	161	85	2.6 GB

Instantiation	Setup (once)	Publish (per party)	KeyGen (per party)
Small	6 s	0.23 s	0.20 s
Medium	38 s	1.0 s	1.2 s
Large	1700 s	5.1 s	5.9 s
Extra	29000 s	18 s	20 s

The timings above show that our scheme is relatively practical, as the KeyGen phase of the multipartite Diffie-Hellman protocol requires only a few seconds per user; however the parameter size is still very large even with our optimizations.

Acknowledgments. We would like to thank the Crypto referees for their helpful comments.

References

[AGHS12] Agrawal, S., Gentry, C., Halevi, S., Sahai, A.: Discrete Gaussian Leftover Hash Lemma over infinite domains. Cryptology ePrint Archive, Report 2012/714 (2012), http://eprint.iacr.org/

[BS03] Boneh, D., Silverberg, A.: Applications of multilinear forms to cryptography. Contemporary Mathematics 324, 71–90 (2003)

[CCK+13] Cheon, J.H., Coron, J.-S., Kim, J., Lee, M.S., Lepoint, T., Tibouchi, M., Yun, A.: Batch fully homomorphic encryption over the integers. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 315–335. Springer, Heidelberg (2013)

[CH12] Cohn, H., Heninger, N.: Approximate common divisors via lattices. In: ANTS X (2012)

[CLT13a] Coron, J.-S., Lepoint, T., Tibouchi, M.: Batch fully homomorphic encryption over the integers. Cryptology ePrint Archive, Report 2013/036 (2013), http://eprint.iacr.org/

[CLT13b] Coron, J.-S., Lepoint, T., Tibouchi, M.: Practical multilinear maps over the integers. Cryptology ePrint Archive, Report 2013/183 (2013), http://eprint.iacr.org/

[CMNT11] Coron, J.-S., Mandal, A., Naccache, D., Tibouchi, M.: Fully homomorphic encryption over the integers with shorter public keys. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 487–504. Springer, Heidelberg (2011)

[CN12] Chen, Y., Nguyen, P.Q.: Faster algorithms for approximate common divisors: Breaking fully-homomorphic-encryption challenges over the integers. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 502–519. Springer, Heidelberg (2012)

[DGHV10] van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (2010)

[Gen09] Gentry, C.: A fully homomorphic encryption scheme. PhD thesis, Stanford University (2009), http://crypto.stanford.edu/craig

[GGH13] Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 1–17. Springer, Heidelberg (2013)

[GH11] Gentry, C., Halevi, S.: Implementing Gentry's fully-homomorphic encryption scheme. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 129–148. Springer, Heidelberg (2011)

[Gt13] Granlund, T., and the GMP development team: GNU MP: The GNU Multiple Precision Arithmetic Library, 5.1.1 edition (2013), http://gmplib.org/

[HILL99] Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM Journal on Computing 28, 12–24 (1999)

[HPS11] Hanrot, G., Pujol, X., Stehlé, D.: Analyzing blockwise lattice algorithms using dynamical systems. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 447–464. Springer, Heidelberg (2011)

[Lep] Lepoint, T.: An Implementation of Multilinear Maps over the Integers. Available under the Creative Commons License BY-NC-SA at https://github.com/tlepoint/multimap

Full Domain Hash from (Leveled) Multilinear Maps and Identity-Based Aggregate Signatures

Susan Hohenberger[1,⋆], Amit Sahai[2,⋆⋆], and Brent Waters[3,⋆⋆⋆]

[1] Johns Hopkins University
susan@cs.jhu.edu
[2] UCLA
sahai@cs.ucla.edu
[3] University of Texas at Austin
bwaters@cs.utexas.edu

Abstract. In this work, we explore building constructions with full domain hash structure, but with standard model proofs that do not employ the random oracle heuristic. The launching point for our results will be the utilization of a "leveled" *multilinear map* setting for which Garg, Gentry, and Halevi (GGH) recently gave an approximate candidate. Our first step is the creation of a standard model signature scheme that exhibits the structure of the Boneh, Lynn and Shacham signatures. In particular, this gives us a signature that admits unrestricted aggregation.

We build on this result to offer the first *identity-based* aggregate signature scheme that admits unrestricted aggregation. In our construction, an arbitrary-sized set of signatures on identity/message pairs can be aggregated into a single group element, which authenticates the entire set. The identity-based setting has important advantages over regular aggregate signatures in that it eliminates the considerable burden of having to store, retrieve or verify a set of verification keys, and minimizes the total cryptographic overhead that must be attached to a set of signer/message

⋆ Supported by the National Science Foundation (NSF) CNS-1154035, CNS-1228443; the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under contract FA8750-11-2-0211, the Office of Naval Research under contract N00014-11-1-0470, and a Microsoft Faculty Fellowship.

⋆⋆ Research supported in part from a DARPA/ONR PROCEED award, NSF grants 1228984, 1136174, 1118096, 1065276, 0916574 and 0830803, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, and an Okawa Foundation Research Grant. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0389.

⋆⋆⋆ Supported by NSF CNS-0915361, CNS-0952692, CNS-1228599; DARPA through the U.S. Office of Naval Research under Contract N00014-11-1-0382, DARPA N11AP20006, a Google Faculty Research Award, an Alfred P. Sloan Fellowship, a Microsoft Faculty Fellowship, and a Packard Foundation Fellowship. Applying to all authors, the views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 494–512, 2013.

pairs. While identity-based signatures are trivial to achieve, their aggregate counterparts are not. To the best of our knowledge, no prior candidate for realizing unrestricted identity-based aggregate signatures exists in either the standard or random oracle models.

A key technical idea underlying these results is the realization of a hash function with a Naor-Reingold-type structure that is publicly computable using repeated application of the multilinear map. We present our results in a generic "leveled" multilinear map setting and then show how they can be translated to the GGH graded algebras analogue of multilinear maps.

1 Introduction

Applying a full domain hash is a common technique in cryptography where a hash function, modeled as a random oracle, is used to hash a string into a set. Originally, the concept referred to a signature scheme where one hashed into the range of a trapdoor permutation [3]. Subsequently, full domain hash has been treated as a more general concept and applied in bilinear map cryptography where typically a hash function $H : \{0,1\}^* \to \mathbb{G}$ is used to hash a string into a bilinear group. (We note that multiple early works [9,11,10] employ this terminology.) Pairing-based applications of Full Domain Hash include: the original Boneh-Franklin [9], short and aggregate signatures [11,10], Hierarchical Identity-Based Encryption [23], and decentralized Attribute-Based Encryption [26]. Typically, proofs of such schemes will use the random oracle heuristic to "program" the output of the hash function in a certain way for which there is no known standard model equivalent (see [24]).

Given that there are well-known issues with random oracle instantiability in general [14] and problems with Full Domain Hash in particular [18,17], there has been a push to find standard model realizations of these applications. These endeavors have been successful in several applications such as signatures [8,36] and (Hierarchical) Identity-Based Encryption [15,6,7,36,21,37]. Despite this progress, the current state is not entirely satisfactory on two fronts. First, each of the standard model examples given above created new cryptographic constructions with fundamentally different structure than the original Full Domain Hash construction. While creating a new structure is a completely valid and novel approach, that path does not necessarily lend insight or further understanding of the original constructions.

Second, there are important applications of the Full Domain Hash method where implementing such a hash using a random oracle introduces significant limitations in the applicability of the Full Domain Hash method. One example concerns aggregate signature schemes and their identity-based counterparts.

An aggregate signature system is one in which a signature σ' on verification key/message pair (VK', M') can be combined with a signature $\tilde{\sigma}$ on $(\tilde{\mathrm{VK}}, \tilde{M})$ producing a new signature σ on the set $S = \{(\mathrm{VK}', M'), (\tilde{\mathrm{VK}}, \tilde{M})\}$. This process can be repeated indefinitely to aggregrate an arbitrary number of signatures together. Crucially, the size of σ should be independent of the number of signatures aggregated, although the description of the set S will grow. The ultimate goal, however, is to minimize the entire transmission size [31].

The need for a public-key infrastructure for verification keys is a major drawback of traditional public-key cryptography, and for this reason identity-based cryptography has flourished [35,9]: In an *identity-based* aggregate signature scheme, verification keys like VK would be replaced with simple identity strings like $\mathcal{I}$ ="harrypotter@hogwarts.edu". This offers a very meaningful savings for protocols such as BGPsec, which require routers to store, retrieve and verify certificates for over 36,000 public keys [16,13]. We note that while identity-based signatures follow trivially from standard signatures, identity-based aggregate signatures are nontrivial (more on this below).

A decade ago, the Boneh, Gentry, Lynn and Shacham (BGLS) [10] aggregate signature scheme was built using the Full Domain Hash methodology. In the original vision of BGLS, aggregation could be performed by any third party on any number of signatures. The authors showed how the Boneh, Lynn and Shacham (BLS) [11] signatures (which are in turn comprised of Boneh-Franklin [9] private IBE keys) can be aggregated in this manner. The BLS construction uses a full domain hash and its security proof is in the random oracle model. However, even though the BGLS scheme was built upon the key mechanism for Boneh-Franklin Identity-Based Encryption, BGLS does *not* support identity-based aggregation. The Full Domain Hash in BGLS is realized using a random oracle, which destroys the structure that would be needed for identity-based aggregate signatures. To the best of our knowledge, no prior solution to identity-based aggregate signatures in either the standard or random oracle models exists. Prior work considered ID-based aggregates restricted to a common nonce [22] (e.g., where signatures can only be aggregated if they were created with the same nonce or time period) or sequential additions [5] (e.g., where a group of signers sequentially form an aggregate by each adding their own signature to the aggregate-so-far).

Our Results in a Nutshell. In this work, we give a new method for implementing the Full Domain Hash method using leveled multilinear maps, including the ones recently proposed by Garg, Gentry, and Halevi (GGH) [19]. We show how to use this method to implement aggregate signatures in the standard model in a way that naturally extends to give the first full solution to the problem of identity-based aggregate signatures (also in the standard model).

Prior Work on Standard Model Aggregate Signatures. All previous work on achieving standard model aggregate signatures did so by departing fundamentally from the Full Domain Hash methodology.

Subsequently to BGLS [10], different standard model solutions were proposed, but with different restrictions on aggregation. These include: constructions [27] where the signatures must be sequentially added in by the signers, multisignatures [27] where aggregation can occur only for the same message M, or where aggregation is limited to signatures associated with the same nonce or time period [1].[1] These restrictions limit their practical applicability.

[1] We remark that these restrictions were considered in other works such as [33,32,29,4,28] prior to the standard model constructions cited above.

In 2009, Rückert and Schröder [34] gave an intriguing vision on how multilinear maps might enable standard model constructions of aggregate signatures, also departing from the Full Domain Hash methodology. They did not discuss or achieve identity-based aggregate signatures. Their proposal came before the Garg, Gentry and Halevi [19] candidate and used the earlier Boneh-Silverberg [12] view of multilinear maps, where a k-linear map would allow the *simultaneous* multiplication of k source group elements into one target group element. The GGH candidate in contrast allows for encodings to exist on multiple levels and a pairing between an encoding on level i and one on level j gives an encoding on level $i + j$ as long as $i + j$ is less than or equal to some k. One drawback of the Rückert and Schröder construction is that the security proof requires access to an interactive (or oracle-type) assumption in order to answer the signature queries where the structure of the oracle output is essentially identical to the signatures required. This property seems to be tightly coupled with the modeling of a multilinear map as a one time multiplication. In contrast, we will exploit the leveling of the GGH abstraction to actually replace the hash function in a BLS-type structure and obtain proofs from non-interactive assumptions.

1.1 Overview of our Aggregate Signature Constructions

We now overview the constructions and their security claims. To simplify the description of the main ideas, we describe the constructions here in terms of leveled multilinear maps. Later on, we give translations to the GGH framework.

The Base Construction. A trusted setup algorithm will take as input security parameter λ and message bit-length ℓ and run a group generator $\mathcal{G}(1^\lambda, k = \ell+1)$ and outputs a sequence of groups $\mathbb{G} = (\mathbb{G}_1, \ldots, \mathbb{G}_k)$ of prime order p.[2] The group sequence will have canonical generators $g = g_1, g_2, \ldots, g_k$ along with a pairing operation that computes $e(g_i^a, g_j^b) = g_{i+j}^{ab}$ for any $a, b \in \mathbb{Z}_p$ and $i + j \leq k$. The setup algorithm will also choose $\boldsymbol{A} = (A_{1,0} = g^{a_{1,0}}, A_{1,1} = g^{a_{1,1}}), \ldots, (A_{\ell,0} = g^{a_{\ell,0}}, A_{\ell,1} = g^{a_{\ell,1}}) \in \mathbb{G}_1^2$. We define $H : \{0,1\}^\ell \to \mathbb{G}_{k-1}$ as $H(M) = g_{k-1}^{\prod_{i\in[1,\ell]} a_{i,m_i}}$, where m_i are the bits of message M. The hash function hashes a message into the group $\mathbb{G}_{k-1}$. It exhibits a Naor-Reingold [30]-type structure and is publicly computable using repeated application of a multilinear map. Since a group element in $\mathbb{G}_{k-1}$ has one pairing left, it intuitively reflects the bilinear map setting. In our scheme a private key contains a random exponent $\alpha \in \mathbb{Z}_p$ and the corresponding verification key VK contains g^α. A signature on a message M is computed as $\sigma = H(M)^\alpha$ and verified by testing $e(\sigma, g) \stackrel{?}{=} e(H(M), g^\alpha)$.

Stepping back, the structure of our scheme very closely resembles BLS signatures. For this reason it is possible to aggregate them in the BGLS fashion by simply multiplying two together. The size of an aggregate signature depends on the security parameter plus message length ℓ (assuming the group representation size increases with $k = \ell + 1$), but is independent of the number of times aggregation is applied. Aggregation is unrestricted and can be done by any third party.

[2] In practice one will perform a CRHF of an arbitrary length message to ℓ bits.

The Rückert and Schröder construction [34] also insightfully uses a Naor-Reingold type function for aggregation. A key distinction is that in the RS method there is a *unique* NR function for each signer and it is privately computed by each signer per each message/input. In our construction the Naor-Reingold function is computed as a *public* hash using the levels of the multilinear map. A signer simply multiplies in his secret exponent after computing the hash. Thus, this mimicks the BLS structure much more closely. One advantage of our structure is that the hash function can be derived from a single common reference string and then public keys are just a single group element. In addition, we will see that our structure is amenable to proofs under non-interactive assumptions and allow us to extend to the identity-based setting. In the aggregation setting, where bandwidth is at a premium, our smaller public keys and the ability to go identity-based is important.

Proofs of Security. We view our aggregate signatures as signatures on a multi-set of message/verification key pairs for full generality. We prove security in a modular way as a two step process. First, we define a weaker "distinct message" variant of security that only considers an attacker successful if the aggregate forgery no two signers sign the same message. We then show how to transform any distinct message secure scheme into one with standard security. The transformation captures the BGLS idea (formalized by Bellare, Namprempre and Neven [2]) of hashing the public key plus message together. Using the transformation we can focus on designing proofs in the distinct message game. We first prove selective security under a natural analog of the CDH assumption we call the k-Multilinear Computational Diffie-Hellman (k-MCDH) assumption. We next show full (a.k.a., adaptive) security using a subexponentially secure version of the assumption. Finally, we show full security with only polynomial factors in the reduction using a non-interactive, but parameterized assumption.

Realizing Identity-Based Aggregation. The authority will run a setup algorithm that takes the message bit-length ℓ and identity bit-length n. It runs a group generator $\mathcal{G}(1^\lambda, k = \ell + n)$ and outputs a sequence of groups $\mathbb{G} = (\mathbb{G}_1, \ldots, \mathbb{G}_k)$ of prime order p. It creates the parameters $\boldsymbol{A}$ as in the prior scheme and $\boldsymbol{B} = (B_{1,0} = g^{b_{1,0}}, B_{1,1} = g^{b_{1,1}}), \ldots, (B_{n,0} = g^{b_{n,0}}, B_{n,1} = g^{b_{n,1}}) \in \mathbb{G}_1^2$. We define $H : \{0,1\}^n \times \{0,1\}^\ell \to \mathbb{G}_{k-1}$ as $H(\mathcal{I}, M) = g_k^{(\prod_{i\in[1,n]} b_{i,\mathrm{id}_i})(\prod_{i\in[1,\ell]} a_{i,m_i})}$, where m_i are the bits of message M and id_i the bits of $\mathcal{I}$. The hash function is publicly computable from the multilinear map. A secret key for identity $\mathcal{I}$ is computed as $\mathrm{SK}_\mathcal{I} = g_{n-1}^{\prod_{i\in[1,n]} b_{i,\mathrm{id}_i}} \in G_{n-1}$. This can be used to produce a signature on message M by computing $(g_{k-1})^{(\prod_{i\in[1,n]} b_{i,\mathrm{id}_i})(\prod_{i\in[1,\ell]} a_{i,m_i})}$ using the multilinear map. Finally, a signature can be verified by checking $e(\sigma, g) \stackrel{?}{=} H(\mathcal{I}, M)$. The signatures will aggregate in the same manner by multiplying together.

The distinct message translation is not required in the identity-based setting, because there is no rogue key problem. We first prove selective security under the k-MCDH assumption, and then show full security using a subexponentially

secure version of the assumption. We provide these proofs in both the generic multilinear and the GGH framework.

Further Applications. Taken altogether we show that multilinear forms provide an opportunity for revisiting cryptographic structures that were strongly associated with the random oracle heuristic. It remains to be seen how widely this direction will apply. One interesting example of an application that currently requires the full domain hash is the decentralized Attribute-Based Encryption system of Lewko and Waters [26]. There is no standard model candidate that has comparable expressiveness. Here performing an analogous transformation to our aggregate signatures hash function gives a candidate construction that we do not immediately see how to break. However, it is less easy to see how our proof techniques would extend to the variant of the Lewko-Waters [26] decentralized ABE scheme.

2 Leveled Multilinear Maps and the GGH Graded Encoding

We give a description of generic, leveled multilinear maps. The assumptions used in this setting are defined inline with their respective security proofs. Basic details of the GGH graded algebras analogue of mulitlinear maps are included where used, and for further details, please refer to [19].

For generic, leveled multilinear maps, we assume the existence of a group generator $\mathcal{G}$, which takes as input a security parameter 1^λ and a positive integer k to indicate the number of allowed pairing operations. $\mathcal{G}(1^\lambda, k)$ outputs a sequence of groups $\boldsymbol{G} = (\mathbb{G}_1, \ldots, \mathbb{G}_k)$ each of large prime order $p > 2^\lambda$. In addition, we let g_i be a canonical generator of $\mathbb{G}_i$ (and is known from the group's description). We let $g = g_1$.

We assume the existence of a set of bilinear maps $\{e_{i,j} : G_i \times G_j \to G_{i+j} \mid i, j \geq 1;\ i + j \leq k\}$. The map $e_{i,j}$ satisfies the following relation:

$$e_{i,j}\left(g_i^a, g_j^b\right) = g_{i+j}^{ab} \ : \ \forall a, b \in \mathbb{Z}_p$$

We observe that one consequence of this is that $e_{i,j}(g_i, g_j) = g_{i+j}$ for each valid i, j.

When the context is obvious, we will sometimes abuse notation and drop the subscripts i, j, For example, we may simply write $e\left(g_i^a, g_j^b\right) = g_{i+j}^{ab}$.

Algorithmic Components of GGH Encodings. While we assume familiarity with the basics of GGH encodings [19], we now review the algorithmic components of the GGH encodings that we will use in our constructions and proofs. The setup algorithm $\mathsf{InstGen}(1^\lambda, 1^k)$ takes as input a security parameter 1^λ and the level of multilinearity 1^k, and outputs the public parameters params needed for using the remaining GGH algorithms, along with a special parameter $\mathbf{p}_{zt}$ to be used for zero testing. The sampling algorithm $\mathsf{samp}(\mathsf{params})$ outputs a level-0 encoding of a randomly chosen element. The canonicalizing encoding $\mathsf{cenc}_e(\mathsf{params}, i, \alpha)$

algorithm takes as input an encoding α of some element a, and outputs a level-i encoding of a, with re-randomization parameter e. This canonicalizing encoding algorithm can re-randomize an encoding for a fixed constant number of re-randomization parameters e. Finally, the zero-testing algorithm $\mathsf{isZero}(\mathbf{p}_{zt}, \alpha)$ takes as input a level-k encoding α, and accepts iff α is an encoding of 0. A more elaborate review of these algorithms can be found elsewhere in these proceedings [20] (omitted here for lack of space).

3 Definitions for Aggregate and ID-Based Aggregate Signatures

We now give our definitions for aggregate signatures. In our setting, each aggregate signature is associated with a *multiset* S over verification key/message pairs (or identity/message pairs in the ID-based setting). A set S is of the form $\{(\mathrm{VK}_1, M_1), \ldots, (\mathrm{VK}_{|S|}, M_{|S|})\}$. Since S is a multiset it is possible to have $(\mathrm{VK}_i, M_i) = (\mathrm{VK}_j, M_j)$ for $i \neq j$. All signatures, including those that come out of the sign algorithm, are considered to be aggregate signatures. The aggregation algorithm is general in that it can take any two aggregate signatures and combine them into a new aggregate signature.

Our definition allows for an initial trusted setup that will generate a set of common public parameters PP. This will define a bit length of all messages (and identities). In practice one could set these fixed lengths to be the output length ℓ of a collision resistant hash function and allow arbitrary-length messages/identities by first hashing them down to ℓ bits. In the ID-based setting, the authority also produces a master secret key used later to run the key generation algorithm.

We emphasize a few features of our setting. First, aggregation is very general in that it allows for the combination of any two aggregate signatures into a single one. Some prior definitions required an aggregate signature to be combined with a single message signature. This is a limitation for applications where an aggregator comes across two aggregate signatures that is wishes to combine. The aggregation operation does not require any secret keys. The multiset structure allows one to combine two aggregate signatures which both include the same message from the same signer.

We begin formally with the ID-based definition, because it is novel to this work, and then discuss its simpler counterpart.

Authority-Setup($1^\lambda, \ell, n$*).* The trusted setup algorithm takes as input the security parameter as well the bit-length ℓ of messages and bit-length n of the identities. It outputs a common set of public parameters PP and master secret key MSK.

KeyGen($\mathrm{MSK}, \mathcal{I} \in \{0,1\}^n$*).* The key generation algorithm is run by the authority. It takes as input the system master secret key and an identity $\mathcal{I}$, and outputs a secret signing key $\mathrm{SK}_{\mathcal{I}}$.

Sign($\mathrm{PP}, \mathrm{SK}_{\mathcal{I}}, \mathcal{I} \in \{0,1\}^n, M \in \{0,1\}^\ell$*).* The signing algorithm takes as input a secret signing key and corresponding identity $\mathcal{I} \in \{0,1\}^n$, the common public parameters as well as a message $M \in \{0,1\}^\ell$. It outputs a signature σ for identity $\mathcal{I}$. *We emphasize that a single signature that is output by this algorithm is considered to also be an aggregate signature.*

Aggregate($\mathrm{PP}, \tilde{S}, S', \tilde{\sigma}, \sigma'$*).* The aggregation algorithm takes as input two multisets $\tilde{S}$ and S' and purported signatures $\tilde{\sigma}$ and σ'. The elements of $\tilde{S}$ consist of identity/message pairs $\{(\tilde{\mathcal{I}}_1, \tilde{M}_1), \ldots, (\tilde{\mathcal{I}}_{|\tilde{S}|}, \tilde{M}_{|\tilde{S}|})\}$ and the elements of S' consist of $\{(\mathcal{I}'_1, M'_1), \ldots, (\mathcal{I}'_{|S'|}, M'_{|S'|})\}$. The process produces a signature σ on the multiset $S = \tilde{S} \cup S'$, where $\cup$ is a multiset union.

Verify(PP, S, σ*).* The verification algorithm takes as input the public parameters, a multiset S of identity and message pairs and an aggregate signature σ. It outputs true or false to indicate whether verification succeeded.

Correctness. The correctness property states that all valid aggregate signatures will pass the verification algorithm, where a valid aggregate is defined recursively as an aggregate signature derived by an application of the aggregation algorithm on two valid inputs or the signing algorithm. More formally, for all integers $\lambda, \ell, n, k \geq 1$, all $\mathrm{PP} \in \text{Authority-Setup}(1^\lambda, \ell, n)$, all $\mathcal{I}_1, \ldots, \mathcal{I}_k \in \{0,1\}^n$, all $\mathrm{SK}_{\mathcal{I}_i} \in \mathrm{KeyGen}(\mathrm{PP}, \mathcal{I}_i)$, $\mathrm{Verify}(\mathrm{PP}, S, \sigma) = 1$, if σ is a *valid* aggregate for multiset S under PP. We say that an aggregate signature σ is *valid* for multiset S if: (1) $S = \{(\mathcal{I}_i, M)\}$ for some $i \in [1, k]$, $M \in \{0,1\}^\ell$ and $\sigma \in \mathrm{Sign}(\mathrm{PP}, \mathrm{SK}_{\mathcal{I}_i}, \mathcal{I}_i, M)$; or (2) there exists multisets $S', \tilde{S}$ where $S = S' \cup \tilde{S}$ and valid aggregate signatures $\sigma', \tilde{\sigma}$ on them respectively such that $\sigma \in \mathrm{Aggregate}(\mathrm{PP}, \tilde{S}, S', \tilde{\sigma}, \sigma')$.

Security Model for Aggregate Signatures. Adapting aggregation [10,2] to the identity-based setting takes some care in considering how keys are handled and which query requests the adversary should be allowed to make. Informally, in the unforgeability game, it should be computationally infeasible for any adversary to produce a forgery implicating an honest identity, even when the adversary can control all other identities involved in the aggregate and can mount a chosen-message attack on the honest identity. This is defined using a game between a challenger and an adversary $\mathcal{A}$ with respect to scheme Π = (Authority-Setup, KeyGen, Sign, Aggregate, Verify).

– **ID-Unforg**$(\Pi, \mathcal{A}, \lambda, \ell, n)$:

Setup. The challenger runs $\text{Authority-Setup}(1^\lambda, \ell, n)$ to obtain PP. It sends PP to $\mathcal{A}$.

Queries. Proceeding adaptively, $\mathcal{A}$ can make three types of requests:

1. Create New Key: The challenger begins with an index $i = 1$ and an empty sequence of index/identity/private key triples T. On input an identity $\mathcal{I} \in \{0,1\}^n$, the challenger runs $\mathrm{KeyGen}(\mathrm{MSK}, \mathcal{I})$ to obtain $\mathrm{SK}_{\mathcal{I}}$. It adds the triple $(i, \mathcal{I}, \mathrm{SK}_{\mathcal{I}})$ to T and then increments i for the next

call. Nothing is returned to the adversary. *We note that the adversary can query this oracle multiple times for the same identity. This will capture security for applications that might release more than one secret key per identity.*

2. Corrupt User: On input an index $i \in [1, |T|]$, the challenger returns to the adversary the triple $(i, \mathcal{I}_i, \mathrm{SK}_{\mathcal{I}_i}) \in T$. It returns an error if T is empty or i is out of range.
3. Sign: On input an index $i \in [1, |T|]$ and a message $M \in \{0,1\}^\ell$, the challenger obtains the triple $(i, \mathcal{I}_i, \mathrm{SK}_{\mathcal{I}_i}) \in T$ (returning an error if it does not exist) and returns the signature resulting from $\mathrm{Sign}(\mathrm{PP}, \mathrm{SK}_{\mathcal{I}_i}, \mathcal{I}_i, M)$ to $\mathcal{A}$.

Response. Finally, $\mathcal{A}$ outputs a multiset S^* of identity/message pairs and a purported aggregate signature σ^*.

We say the adversary "wins" or that the output of this experiment is 1 if: (1) $\mathrm{Verify}(\mathrm{PP}, S^*, \sigma^*) = 1$ and (2) there exists an element $(\mathcal{I}^*, M^*) \in S^*$ such that M^* was not queried for a signature by the adversary on any index corresponding to $\mathcal{I}^*$; i.e., any index i such that $(i, \mathcal{I}^*, \cdot) \in T$. Otherwise, the output is 0. Define $\textbf{ID-Forg}_{\mathcal{A}}$ as the probability that $\textbf{Unforg}(\Pi, \mathcal{A}, \lambda, \ell, n) = 1$, where the probability is over the coin tosses of the Authority-Setup, KeyGen, and Sign algorithms and of $\mathcal{A}$.

Definition 1 (Adaptive Unforgeability). *An ID-based aggregate signature scheme Π is existentially* unforgeable *with respect to adaptive chosen-message attacks if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the function $\textbf{ID-Forg}_A$ is negligible in λ.*

Selective Security. We consider a selective variant to **ID-Unforg** (selective in both the identity and the message) where there is an Init phase before the Setup phase, wherein $\mathcal{A}$ gives to the challenger a forgery identity/message pair $(\mathcal{I}^* \in \{0,1\}^n, M^* \in \{0,1\}^\ell)$. The adversary cannot request a signing key for $\mathcal{I}^*$. (It may request that the challenger create one or more keys for this identity, but it cannot corrupt any user index i associated with $\mathcal{I}^*$.) Moreover, the adversary only "wins" causing the experiment output to be 1 if the normal checks hold (i.e., its signature verifies and it did not request that $\mathcal{I}^*$ sign M^*) and additionally $(\mathcal{I}^*, M^*)$ appears in S^*.

Non-ID-Based Aggregates and the Distinct Message Variant. We provide security definitions for the non-ID-based setting in the full version [25] that follow from [10,2]. We provide adaptive and selective variants. We also identify a weaker "distinct message" security game that is easier to work with. In the full version [25], we describe and prove secure a simple transformation from distinct message security to standard aggregate signature security. The transformation captures the idea of hashing the public key and message together [10,2] in a modular way. Focusing on distinct message security allows one to avoid the "rogue key" attack (see Section 4.2). We do not consider distinct message security in the ID-based setting, because there are no verification keys.

4 Our Base Aggregate Signature Construction

4.1 Generic Multlinear Construction

Setup$(1^\lambda, \ell)$ The trusted setup algorithm takes as input the security parameter as well as the length ℓ of messages. It first runs $\mathcal{G}(1^\lambda, k = \ell + 1)$ and outputs a sequence of groups $\mathbb{G} = (\mathbb{G}_1, \ldots, \mathbb{G}_k)$ of prime order p, with canonical generators $g_1, \ldots, g_k$, where we let $g = g_1$.

Next, it outputs random group elements $(A_{1,0}, A_{1,1}), \ldots, (A_{\ell,0}, A_{\ell,1}) \in \mathbb{G}_1^2$. These will be used to compute a function $H(M) : \{0,1\}^\ell \to \mathbb{G}_{k-1}$, which serves as the analog of the full domain hash function of the BGLS [10] construction. Let $m_1, \ldots, m_\ell$ be the bits of message M. It is computed iteratively as $H_1(M) = A_{1,m_1}$ and for $i \in [2, \ell]$, $H_i(M) = e(H_{i-1}(M), A_{i,m_i})$. We define $H(M) = H_\ell(M)$. The public parameters, PP, consist of the group descriptions plus $(A_{1,0}, A_{1,1}), \ldots, (A_{\ell,0}, A_{\ell,1})$.

KeyGen(PP) The key generation algorithm first chooses random $\alpha \in \mathbb{Z}_p$. It outputs the public verification key as $\mathrm{VK} = g^\alpha$. The secret key SK is $\alpha \in \mathbb{Z}_p$.

Sign$(\mathrm{PP}, \mathrm{SK}, M \in \{0,1\}^\ell)$ The signing algorithm computes the signature as $\sigma = H(M)^\alpha \in G_{k-1}$. This serves as an aggregate signature for the (single element) multiset $S = (\mathrm{VK}, M)$.

Aggregate$(\mathrm{PP}, \tilde{S}, S', \tilde{\sigma}, \sigma')$. The aggregation algorithm simply computes the output signature σ as $\sigma = \tilde{\sigma} \cdot \sigma'$. The serves as a signature on the multiset $S = \tilde{S} \cup S'$, where $\cup$ is a *multiset union*.

Verify(PP, S, σ). The verification algorithm parses S as $\{(\mathrm{VK}_1, M_1), \ldots, (\mathrm{VK}_{|S|}, , M_{|S|})\}$. It then checks that $e(\sigma, g) \stackrel{?}{=} \prod_{i=1,\ldots,|S|} e(H(M_i), \mathrm{VK}_i)$ and accepts if and only if it holds.

Correctness To see correctness, an aggregate σ on $S = \{(\mathrm{VK}_1, M_1), \ldots, (\mathrm{VK}_{|S|}, M_{|S|}\}$ is the product of individual signatures; i.e., $\sigma = \prod_{i=1}^{|S|} H(M_i)^{\alpha_i}$ where $\mathrm{VK}_i = g^{\alpha_i}$, and thus passes the verification equation as:

$$e(\sigma, g) = e(\prod_{i=1}^{|S|} H(M_i)^{\alpha_i}, g) = \prod_{i=1}^{|S|} e(H(M_i)^{\alpha_i}, g) = \prod_{i=1}^{|S|} e(H(M_i), g)^{\alpha_i}$$
$$= \prod_{i=1}^{|S|} e(H(M_i), g^{\alpha_i}) = \prod_{i=1}^{|S|} e(H(M_i), VK_i).$$

Efficiency and Tradeoffs An aggregate signature is one group element in $\mathbb{G}_{k-1}$ independent of the number of messages aggregated. In a multilinear setting, the space to represent a group element might grow with k (which is $\ell + 1$). Indeed, this happens in the GGH [19] graded algebra translation. One way to mitigate

this is to differ the message alphabet size in a tradeoff of computation versus storage. The above construction uses a binary message alphabet. If it used an alphabet of 2^d symbols, then the aggregate signature could resident in the group $G_{\ell/d}$ with $\ell/d - 1$ pairings required to compute it, at the cost of the public parameters requiring $2^d\ell$ group elements in $\mathbb{G}$.

Construction in the GGH Framework We give a translation of the above construction to the GGH [19] framework in the full version of this work [25].

4.2 Security Analysis

Assumption 1 (Multilinear Computational Diffie-Hellman: k-MCDH) *The k-Multilinear Computational Diffie-Hellman (k-MCDH) problem states the following: A challenger runs $\mathcal{G}(1^\lambda, k)$ to generate groups and generators of order p. Then it picks random $c_1, \ldots, c_k \in \mathbb{Z}_p$. The assumption then states that given $g = g_1, g^{c_1}, \ldots, g^{c_k}$ it is hard for any poly-time algorithm to compute $g_{k-1}^{\prod_{j\in[1,k]} c_j}$ with better than negligible advantage (in security parameter λ).*

We say that the k-MCDH assumption holds against subexponential advantage if there exists a universal constant $\epsilon_0 > 0$ such that no polynomial-time algorithm can succeed in the experiment above with probability greater than $2^{-\lambda^{\epsilon_0}}$. In Section 5.3, we will give a variant of the k-MCDH assumption in the approximate multilinear maps setting of GGH [19] that we will call the GGH k-MCDH assumption. We note that the best cryptanalysis available of the GGH framework [19] suggests that the GGH k-MCDH assumption holds against subexponential advantage.

In the full version [25], we show that the basic aggregate signature scheme for message length ℓ in the distinct message unforgeability game is:

- Selectively secure under the (ℓ+1)-Multilinear Computational Diffie-Hellman (MCDH) assumption.
- Fully secure under the ($\ell + 1$)-MCDH assumption against subexponential advantage.
- Fully secure under a non-interactive, parameterized assumption which depends on message length ℓ, the number of adversarial signing queries and the number of messages in the adversary's forgery.

By applying a simple transformation given in the full version [25] which follows from [10,2], the distinct message requirement can be removed. Without this transformation, there is a simple attack where the attacker sets some $\mathrm{VK}' = \mathrm{VK}^{-1}$ and submits the identity element in $\mathbb{G}_{k-1}$ as an aggregate forgery for $S = \{(\mathrm{VK}, M), (\mathrm{VK}', M)\}$ for any message M of its choosing.

5 Our ID-Based Aggregate Signature Construction

5.1 Generic Multilinear Construction

Authority-Setup($1^\lambda, \ell, n$) The trusted setup algorithm is run by the master authority of the ID-based system. It takes as input the security parameter as

well the bit-length ℓ of messages and bit-length n of identities. It first runs $\mathcal{G}(1^\lambda, k = \ell + n)$ and outputs a sequence of groups $\mathbb{G} = (\mathbb{G}_1, \ldots, \mathbb{G}_k)$ of prime order p, with canonical generators $g_1, \ldots, g_k$, where we let $g = g_1$.

Next, it chooses random elements $(A_{1,0} = g^{a_{1,0}}, A_{1,1} = g^{a_{1,1}}), \ldots, (A_{\ell,0} = g^{a_{\ell,0}}, A_{\ell,1} = g^{a_{\ell,1}}) \in \mathbb{G}_1^2$ and random exponents $(b_{1,0}, b_{1,1}), \ldots, (b_{n,0}, b_{n,1}) \in {\mathbb{Z}_p}^2$. It sets $B_{i,\beta} = g^{b_{i,\beta}}$ for $i \in [1, n]$ and $\beta \in \{0, 1\}$. These will be used to define a function $H(\mathcal{I}, M) : \{0,1\}^n \times \{0,1\}^\ell \to \mathbb{G}_k$. Let $m_1, \ldots, m_\ell$ be the bits of message M and $\mathrm{id}_1, \ldots, \mathrm{id}_n$ as the bits of $\mathcal{I}$. It is computed iteratively as

$$H_1(\mathcal{I}, M) = B_{1,\mathrm{id}_1} \quad \text{for } i \in [2, n] \quad H_i(\mathcal{I}, M) = e(H_{i-1}(\mathcal{I}, M), B_{i,\mathrm{id}_i})$$

$$\text{for } i \in [n+1, n+\ell = k] \quad H_i(\mathcal{I}, M) = e(H_{i-1}(\mathcal{I}, M), A_{i-n,m_{i-n}}).$$

We define $H(\mathcal{I}, M) = H_{k=\ell+n}(\mathcal{I}, M)$.

The public parameters, PP, consist of the group sequence description plus:

$$(A_{1,0}, A_{1,1}), \ldots, (A_{\ell,0}, A_{\ell,1}), (B_{1,0}, B_{1,1}), \ldots, (B_{n,0}, B_{n,1})$$

The master secret key MSK includes PP together with the values $(b_{1,0}, b_{1,1}), \ldots, (b_{n,0}, b_{n,1})$.

KeyGen($\mathrm{MSK}, \mathcal{I} \in \{0,1\}^n$*).* The signing key for identity $\mathcal{I}$ is $\mathrm{SK}_\mathcal{I} = g_{n-1}^{\prod_{i\in[1,n]} b_{i,\mathrm{id}_i}} \in G_{n-1}$.

Sign($\mathrm{PP}, \mathrm{SK}_\mathcal{I}, \mathcal{I} \in \{0,1\}^n, M \in \{0,1\}^\ell$*)* The signing algorithm lets temporary variable $D_0 = \mathrm{SK}_\mathcal{I}$. Then for $i = 1$ to ℓ it computes $D_i = e(D_{i-1}, A_{i,m_i}) \in G_{n-1+i}$. The output signature is

$$\sigma = D_\ell = (g_{k-1})^{(\prod_{i\in[1,n]} b_{i,\mathrm{id}_i})(\prod_{i\in[1,\ell]} a_{i,m_i})}.$$

This serves as an ID-based aggregate signature for the (single element) multiset $S = (\mathcal{I}, M)$.

Aggregate($\mathrm{PP}, \tilde{S}, S', \tilde{\sigma}, \sigma'$*).* The aggregation algorithm simply computes the output signature σ as $\sigma = \tilde{\sigma} \cdot \sigma'$. The serves as a signature on the multiset $S = \tilde{S} \cup S'$, where $\cup$ is a multiset union.

Verify(PP, S, σ*).* It parses S as $\{(\mathcal{I}_1, M_1), \ldots, (\mathcal{I}_{|S|}, M_{|S|})\}$. It then accepts if and only if

$$e(\sigma, g) \stackrel{?}{=} \prod_{i=1,\ldots,|S|} H(\mathcal{I}_i, M_i).$$

Correctness and Security. For correctness, an aggregate σ on $S = \{(\mathcal{I}_1, M_1), \ldots, (\mathcal{I}_{|S|}, M_{|S|})\}$ is the product of individual signatures; i.e., σ_i where $e(\sigma_i, g) = H(\mathcal{I}_i, M_i)$, and thus $\prod_{i=1}^{|S|} e(\sigma_i, g) = e(\prod_{i=1}^{|S|} \sigma_i, g) = e(\sigma, g) = \prod_{i=1}^{|S|} H(\mathcal{I}_i, M_i)$. Proof of the following theorem appears in the full version [25] and is similar to the proof for the GGH translation which we provide shortly in Section 5.3.

Theorem 2 (Selective Security of ID-Based Construction). *The ID-based aggregate signature scheme for message length ℓ and identity length n in Section 5.1 is selectively secure in the unforgeability game in Section 3 under the $(\ell + n)$-MCDH assumption.*

5.2 ID-Based Construction in the GGH Framework

We show how to modify our ID-based construction to use the GGH [19] graded algebras analogue of multilinear maps. Please note that we use the same notation developed in [19], with some minor changes: Firstly, we use the canonical encoding function cenc provided by the GGH framework more than once at each level of the encoding, but only a globally fixed constant number of times per level. This is compatible with the GGH encoding [19], and allows for a simpler exposition of our scheme and proof. Also, **for ease of notation on the reader, we suppress repeated params arguments that are provided to every algorithm.** Thus, for instance, we will write $\alpha \leftarrow \mathsf{samp}()$ instead of $\alpha \leftarrow \mathsf{samp}(\mathsf{params})$. Note that in our scheme, there will only ever be a single uniquely chosen value for params throughout the scheme, so there is no cause for confusion. Finally, we use the variant of the GGH framework with "strong" zero-testing, where the zero test statistically guarantees that a vector is a valid encoding of zero if it passes the zero test. For further details on the GGH framework, please refer to [19]. See also [20] in these proceedings.

Authority-Setup($1^\lambda, \ell, n$) The trusted setup algorithm is run by the master authority of the ID-based system. It takes as input the security parameter as well the bit-length ℓ of messages and bit-length n of identities. It then runs $(\mathsf{params}, \mathbf{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^{k=\ell+n})$. Recall that params will be implicitly given as input to all GGH-related algorithms below.

Next, it chooses random encodings $a_{i,\beta} = \mathsf{samp}()$ for $i \in [1, \ell]$ and $\beta \in \{0, 1\}$; and random encodings $b_{i,\beta} = \mathsf{samp}()$ for $i \in [1, n]$ and $\beta \in \{0, 1\}$. Then it assigns $A_{i,\beta} = \mathsf{cenc}_1(1, a_{i,\beta})$ for $i \in [1, \ell]$ and $\beta \in \{0, 1\}$; and it assigns $B_{i,\beta} = \mathsf{cenc}_1(1, b_{i,\beta})$ for $i \in [1, n]$ and $\beta \in \{0, 1\}$.

These will be used to compute a function H mapping $\ell + n$ bit strings to level $k - 1$ encodings. Let $m_1, \ldots, m_\ell$ be the bits of M and $\mathrm{id}_1, \ldots, \mathrm{id}_n$ be the bits of $\mathcal{I}$. It is computed iteratively as

$$H_1(\mathcal{I}, M) = B_{1,\mathrm{id}_1} \quad \text{for } i \in [2, n] \;\; H_i(\mathcal{I}, M) = H_{i-1}(\mathcal{I}, M) \cdot B_{i,\mathrm{id}_i}$$

$$\text{for } i \in [n+1, n+\ell = k] \;\; H_i(\mathcal{I}, M) = H_{i-1}(\mathcal{I}, M) \cdot A_{i-n,m_{i-n}}.$$

We define $H(\mathcal{I}, M) = \mathsf{cenc}_2(k, H_{k=\ell+n}(\mathcal{I}, M))$.

The public parameters, PP, consist of the $\mathsf{params}, \mathbf{p}_{zt}$ plus:

$$(A_{1,0}, A_{1,1}), \ldots, (A_{\ell,0}, A_{\ell,1}), (B_{1,0}, B_{1,1}), \ldots, (B_{n,0}, B_{n,1})$$

Note that params includes a level 1 encoding of 1, which we denote as g.

The master secret key MSK includes PP together with the encodings $(b_{1,0}, b_{1,1}), \ldots, (b_{n,0}, b_{n,1})$.

KeyGen($\mathrm{MSK}, \mathcal{I} \in \{0,1\}^n$*)* The signing key for identity $\mathcal{I}$ is $\mathrm{SK}_{\mathcal{I}} = \mathsf{cenc}_2(n-1, \prod_{i\in[1,n]} b_{i,\mathrm{id}_i})$.

Sign($\mathrm{PP}, \mathrm{SK}_{\mathcal{I}}, \mathcal{I} \in \{0,1\}^n, M \in \{0,1\}^\ell$*)* The signing algorithm lets temporary variable $D_0 = \mathrm{SK}_{\mathcal{I}}$. Then for $i = 1$ to ℓ it computes $D_i = D_{i-1} \cdot A_{i,m_i}$. The output signature is

$$\sigma = \mathsf{cenc}_3(k-1, D_\ell).$$

This serves as an ID-based aggregate signature for the (single element) multiset $S = (\mathcal{I}, M)$.

Aggregate($\mathrm{PP}, \tilde{S}, S', \tilde{\sigma}, \sigma'$*).* The aggregation algorithm simply computes the output signature σ as $\sigma = \tilde{\sigma} + \sigma'$. The serves as a signature on the multiset $S = \tilde{S} \cup S'$, where $\cup$ is a multiset union.

Verify(PP, S, σ*).* The verification algorithm parses S as $\{(\mathcal{I}_1, M_1), \ldots, (\mathcal{I}_{|S|}, M_{|S|})\}$. It rejects if the multiplicity of any identity/message pair is greater than 2^λ.

The algorithm then proceeds to check the signature by setting $\tau = \mathsf{cenc}_2(1, g)$, and testing: :

$$\mathsf{isZero}\left(\mathbf{p}_{zt}, \tau \cdot \sigma - \sum_{i=1,\ldots,|S|} H(\mathcal{I}_i, M_i)\right)$$

and accepts if and only if the zero testing procedure outputs true. Recall that g above is a canonical level 1 encoding of 1 that is included in params, part of the public parameters.

Correctness. Correctness follows from the same argument as for the ID-based aggregate signature scheme in the generic multilinear setting.

5.3 Proof of Security for ID-based Aggregate Signatures in the GGH framework

We now describe how to modify our proof of security for our ID-based construction to use the GGH [19] graded algebras analogue of multilinear maps. As before, for ease of notation on the reader, we suppress repeated params arguments that are provided to every algorithm. For further details, please see [19].

We begin by describing the GGH analogue of the k-MCDH assumption that we will employ:

Assumption 3 *[GGH analogue of k-MCDH: GGH k-MCDH] The GGH k-Multilinear Computational Diffie-Hellman (GGH k-MCDH) problem states the following: A challenger runs* $\mathsf{InstGen}(1^\lambda, 1^k)$ *to obtain* $(\mathsf{params}, \mathbf{p}_{zt})$*. Note that* params *includes a level 1 encoding of 1, which we denote as g. Then it picks random* $c_1, \ldots, c_k$ *each equal to the result of a fresh call to* $\mathsf{samp}()$*.*

The assumption then states that given $\mathsf{params}, \mathbf{p}_{zt}, \mathsf{cenc}_1(1, c_1), \ldots, \mathsf{cenc}_1(1, c_k)$ *it is hard for any poly-time algorithm to compute an integer* $t \in [1, 2^\lambda]$ *and an encoding* z *such that*

$$zTst\left(\mathbf{p}_{zt}, \mathsf{cenc}_2(1, g) \cdot z - \mathsf{cenc}_1(k, t \cdot \prod_{j \in [1,k]} c_j)\right)$$

outputs true.

We say the GGH k-MCDH assumption holds against subexponential advantage if there exists a universal constant $\epsilon_0 > 0$ such that no polynomial-time algorithm can succeed in the experiment above with probability greater than $2^{\lambda^{\epsilon_0}}$. The best cryptanalysis available of the GGH framework [19] suggests that the GGH k-MCDH assumption holds against subexponential advantage.

We establish full security of our ID-based aggregate signature scheme conditioned on the k-MCDH assumption holding against subexponential advantage. This follows immediately from the following theorem and a standard complexity leveraging argument:

Theorem 4 (Selective Security of GGH ID-Based Construction). *The ID-based aggregate signature scheme for message length* ℓ *and identity length* n *in Section 5.2 is selectively secure in the unforgeability game in Section 3 under the GGH* $(\ell + n)$*-MCDH assumption.*

Corollary 1. *The ID-based aggregate signature scheme for message length* ℓ *in Section 5.2 is fully secure in the distinct message unforgeability game under the GGH* $(\ell + n)$*-MCDH assumption against subexponential advantage.*

Proof. This follows immediately from a complexity leveraging argument: the security parameter λ is chosen to ensure that $2^{\lambda^{\epsilon_0}} >> 2^\ell$, where $2^{-\lambda^{\epsilon_0}}$ is the maximum probability of success allowed in the k-MCDH assumption against subexponential advantage. Now, to establish full security, the simulator performs exactly as in the selective security proof, but first it simply guesses the message that will be forged (instead of expecting the adversary to produce this message). Because this guess will be correct with probability at least $2^{-\ell}$, and the security parameter λ is chosen carefully, full security with polynomial advantage (or even appropriately defined subexponential advantage) implies an attacker on the GGH k-MCDH assumption with subexponential advantage.

Proof. **(of Theorem 4)** We show that if there exists a PPT adversary $\mathcal{A}$ that can break the selective security of the ID-based aggregate signature scheme in the unforgettability game with probability ϵ for message length ℓ, identity length n and security parameter λ, then there exists a PPT simulator that can break the GGH $(\ell + n)$-MCDH assumption for security parameter λ with probability ϵ. The simulator takes as input a GGH MCDH instance $\mathsf{params}, \mathbf{p}_{zt}, C_1 = \mathsf{cenc}_1(1, c_1), \ldots, C_k = \mathsf{cenc}_1(1, c_k)$ where $k = \ell + n$. Let m_i denote the ith bit of M and id_i denote the ith bit of $\mathcal{I}$. The simulator plays the role of the challenger in the game as follows.

Init. Let $\mathcal{I}^* \in \{0,1\}^n$ and $M^* \in \{0,1\}^\ell$ be the forgery identity/message pair output by $\mathcal{A}$.

Setup. The simulator chooses random $x_1, \ldots, x_\ell, y_1, \ldots, y_n$ with fresh calls to $\mathsf{samp}()$. For $i = 1$ to ℓ, let $A_{i,m_i^*} = C_{i+n}$ and $A_{i,\bar{m_i^*}} = \mathsf{cenc}_1(1, x_i)$. For $i = 1$ to n, let $B_{i,\mathrm{id}_i^*} = C_i$ and $B_{i,\bar{\mathrm{id}_i^*}} = \mathsf{cenc}_1(1, y_i)$. The parameters are distributed independently and uniformly at random as in the real scheme.

Queries. Conceptually, the simulator will be able to create keys or signatures for the adversary, because his requests will differ from the challenge identity or message in at least one bit. More specifically,

1. Create New Key: The simulator begins with an index $i = 1$ and an empty sequence of index/identity/private key triples T. On input an identity $\mathcal{I} \in \{0,1\}^n$, if $\mathcal{I} = \mathcal{I}^*$, the simulator records $(i, \mathcal{I}^*, \perp)$ in T. Otherwise, the simulator computes the secret key as follows. Let β be the first index such that $\mathrm{id}_i \neq \mathrm{id}_i^*$. Compute $s = \prod_{i=1,\ldots,n \wedge i \neq \beta} B_{i,\mathrm{id}_i}$. Then compute $\mathrm{SK}_{\mathcal{I}} = \mathsf{cenc}_2(n-1, s \cdot y_\beta)$. Record $(i, \mathcal{I}, \mathrm{SK}_{\mathcal{I}})$ in T. Secret keys are well-formed and, due to the rerandomization in the cenc_2 algorithm, are distributed in a manner statistically exponentially close to the keys generated in the real game.
2. Corrupt User: On input an index $i \in [1, |T|]$, the simulator returns to the adversary the triple $(i, \mathcal{I}_i, \mathrm{SK}_{\mathcal{I}_i}) \in T$. It returns an error if T is empty or i is out of range. Recall that i cannot be associated with $\mathcal{I}^*$ in this game.
3. Sign: On input an index $i \in [1, |T|]$ and a message $M \in \{0,1\}^\ell$, the simulator obtains the triple $(i, \mathcal{I}_i, \mathrm{SK}_{\mathcal{I}_i}) \in T$ or returns an error if it does not exist. If $\mathcal{I}_i \neq \mathcal{I}^*$, then the simulator signs M with $\mathrm{SK}_{\mathcal{I}_i}$ in the usual way.

   If $\mathcal{I}_i = \mathcal{I}^*$, then we know $M \neq M^*$. Let β be the first index such that $m_\beta \neq m_\beta^*$. First compute $\sigma' = \prod_{i=1,\ldots,\ell \wedge i \neq \beta} A_{i,m_i}$. Next, compute $\sigma'' = \sigma' \cdot x_i$. Also compute $\gamma = \prod_{i=1,\ldots,n} B_{i,\mathrm{id}_i}$. Finally, compute $\sigma = \mathsf{cenc}_3(k-1, \gamma \cdot \sigma'')$ Return σ to $\mathcal{A}$. Signatures are well-formed and, due to the rerandomization in the cenc_3 algorithm, distributed in a manner statistically exponentially close to the keys generated in the real game.

Response. Eventually, $\mathcal{A}$ outputs an aggregate signature σ^* on multiset S^* where $(\mathcal{I}^*, M^*) \in S^*$. The simulator will extract from this a solution to the MCDH problem. This works by iteratively computing all the other signatures in S^* and then subtracting them out of the aggregate until only one or more signatures on $(\mathcal{I}^*, M^*)$ remain. That is, the simulator takes an aggregate for S^* and computes an aggregate signature for S' where S' has one less verification key/message pair than S at each step. These signatures will be computed as in the query phase.

Eventually, we have an aggregate σ' on $t \geq 1$ instances of $(\mathcal{I}^*, M^*)$. However recall that $H(\mathcal{I}^*, M^*)$ is a level k encoding of $(\prod_{i \in [1,n]} b_{i,\mathrm{id}_i^*})(\prod_{i \in [1,\ell]} a_{i,m_i^*}) = \prod_{i \in [k]} c_i$. Thus verification of the signature

σ' implies that (t, σ') is a solution to the GGH k-MCDH problem, and so the simulator returns (t, σ') to break the GGH k-MCDH assumption.

The responses of the challenger are distributed statistically exponentially closely to the real unforgeability game. The simulator succeeds whenever $\mathcal{A}$ does.

References

1. Ahn, J.H., Green, M., Hohenberger, S.: Synchronized aggregate signatures: new definitions, constructions and applications. In: ACM Conference on Computer and Communications Security, pp. 473–484 (2010)
2. Bellare, M., Namprempre, C., Neven, G.: Unrestricted aggregate signatures. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 411–422. Springer, Heidelberg (2007)
3. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security, pp. 62–73 (1993)
4. Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer, Heidelberg (2002)
5. Boldyreva, A., Gentry, C., O'Neill, A., Yum, D.H.: Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. In: ACM Conference on Computer and Communications Security, pp. 276–285 (2007)
6. Boneh, D., Boyen, X.: Efficient selective-ID secure identity-based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)
7. Boneh, D., Boyen, X.: Secure identity based encryption without random oracles. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 443–459. Springer, Heidelberg (2004)
8. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
9. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. SIAM J. Comput. 32(3), 586–615 (2001); extended abstract in: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–615. Springer, Heidelberg (2001)
10. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and verifiably encrypted signatures from bilinear maps. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, Springer, Heidelberg (2003)
11. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)
12. Boneh, D., Silverberg, A.: Applications of multilinear forms to cryptography. IACR Cryptology ePrint Archive, 80 (2002)
13. Brogle, K., Goldberg, S., Reyzin, L.: Sequential aggregate signatures with lazy verification from trapdoor permutations - (extended abstract). In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 644–662. Springer, Heidelberg (2012)

14. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. J. ACM 51(4), 557–594 (2004)
15. Canetti, R., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 255–271. Springer, Heidelberg (2003)
16. Chi, Y.-J., Oliveira, R., Zhang, L.: Cyclops: The Internet AS-level Observatory. In: ACM SIGCOMM CCR (2008)
17. Dodis, Y., Haitner, I., Tentes, A.: On the instantiability of hash-and-sign RSA signatures. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 112–132. Springer, Heidelberg (2012)
18. Dodis, Y., Oliveira, R., Pietrzak, K.: On the generic insecurity of the full domain hash. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 449–466. Springer, Heidelberg (2005)
19. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 1–17. Springer, Heidelberg (2013)
20. Garg, S., Gentry, C., Halevi, S., Sahai, A., Waters, B.: Attribute-based encryption for circuits from multilinear maps. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 479–499. Springer, Heidelberg (2013)
21. Gentry, C.: Practical identity-based encryption without random oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006)
22. Gentry, C., Ramzan, Z.: Identity-based aggregate signatures. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 257–273. Springer, Heidelberg (2006)
23. Gentry, C., Silverberg, A.: Hierarchical ID-based cryptography. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 548–566. Springer, Heidelberg (2002)
24. Hofheinz, D., Kiltz, E.: Programmable hash functions and their applications. J. Cryptology 25(3), 484–527 (2012)
25. Hohenberger, S., Sahai, A., Waters, B.: Full domain hash from (leveled) multilinear maps and identity-based aggregate signatures. Full version available at the Cryptology ePrint Archive (2013), `http://eprint.iacr.org/`
26. Lewko, A., Waters, B.: Decentralizing attribute-based encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 568–588. Springer, Heidelberg (2011)
27. Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., Waters, B.: Sequential aggregate signatures and multisignatures without random oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 465–485. Springer, Heidelberg (2006)
28. Lysyanskaya, A., Micali, S., Reyzin, L., Shacham, H.: Sequential aggregate signatures from trapdoor permutations. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 74–90. Springer, Heidelberg (2004)
29. Micali, S., Ohta, K., Reyzin, L.: Accountable-subgroup multisignatures: extended abstract. In: ACM Conference on Computer and Communications Security, pp. 245–254 (2001)
30. Naor, M., Reingold, O.: Constructing pseudo-random permutations with a prescribed structure. J. Cryptology 15(2), 97–102 (2002)
31. Neven, G.: Efficient sequential aggregate signed data. IEEE Transactions on Information Theory 57(3), 1803–1815 (2011)
32. Ohta, K., Okamoto, T.: A digital multisignature scheme based on the Fiat-Shamir scheme. In: Matsumoto, T., Imai, H., Rivest, R.L. (eds.) ASIACRYPT 1991. LNCS, vol. 739, pp. 139–148. Springer, Heidelberg (1993)

33. Okamoto, T.: A digital multisignature schema using bijective public-key cryptosystems. ACM Trans. Comput. Syst. 6(4), 432–441 (1988)
34. Rückert, M., Schröder, D.: Aggregate and verifiably encrypted signatures from multilinear maps without random oracles. In: Park, J.H., Chen, H.-H., Atiquzzaman, M., Lee, C., Kim, T.-h., Yeo, S.-S. (eds.) ISA 2009. LNCS, vol. 5576, pp. 750–759. Springer, Heidelberg (2009)
35. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
36. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
37. Waters, B.: Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)

Programmable Hash Functions in the Multilinear Setting

Eduarda S.V. Freire[1,⋆], Dennis Hofheinz[2,⋆⋆], Kenneth G. Paterson[1,⋆⋆⋆], and Christoph Striecks[2]

[1] Royal Holloway, University of London
[2] Karlsruhe Institute of Technology

Abstract. We adapt the concept of a programmable hash function (PHF, Crypto 2008) to a setting in which a multilinear map is available. This enables new PHFs with previously unachieved parameters.

To demonstrate their usefulness, we show how our (standard-model) PHFs can replace random oracles in several well-known cryptographic constructions. Namely, we obtain standard-model versions of the Boneh-Franklin identity-based encryption scheme, the Boneh-Lynn-Shacham signature scheme, and the Sakai-Ohgishi-Kasahara identity-based non-interactive key exchange (ID-NIKE) scheme. The ID-NIKE scheme is the first scheme of its kind in the standard model.

Our abstraction also allows to derive hierarchical versions of the above schemes in settings with multilinear maps. This in particular yields simple and efficient hierarchical generalizations of the BF, BLS, and SOK schemes. In the case of hierarchical ID-NIKE, ours is the first such scheme with full security, in either the random oracle model or the standard model.

While our constructions are formulated with respect to a generic multilinear map, we also outline the necessary adaptations required for the recent "noisy" multilinear map candidate due to Garg, Gentry, and Halevi.

Keywords: programmable hash functions, multilinear maps, identity-based encryption, identity-based non-interactive key exchange, digital signatures.

1 Introduction

Programmable Hash Functions. Programmable hash functions (PHFs) have been proposed in [18] as an abstraction of random oracles that can also be instantiated in the standard model. In a nutshell, a PHF H maps a bitstring X (e.g., a message to be signed) to a group element $\mathsf{H}(X)$; a special trapdoor allows to decompose $\mathsf{H}(X) = c^{ax} h^{bx}$ for previously chosen c, h. In a larger proof,

⋆ Eduarda S.V. Freire was supported by CAPES Foundation/Brazil on grant 0560/09-0 and Royal Holloway, University of London.
⋆⋆ Dennis Hofheinz was supported by a DFG grant (GZ HO 4534/2-1).
⋆⋆⋆ Kenneth G. Paterson was supported by EPSRC Leadership Fellowship EP/H005455/1.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 513–530, 2013.

c will usually be a "challenge element" (e.g., a part of a given Diffie-Hellman challenge), so that $\mathsf{H}(X)$ contains a challenge component if and only if $a_X \neq 0$.

PHFs can be used to employ partitioning strategies: e.g., Waters' CDH-based signature scheme [24] (implicitly) uses a PHF to partition the set of all messages into "signable" and "unsignable" messages. (In his case, a message X is signable iff $a_X \neq 0$.) During the proof of unforgeability, we hope that all messages for which an adversary requests a signature are signable, while the adversary's forgery corresponds to an unsignable message.

Limitations of PHFs. While initially meant as a standard-model replacement for random oracles, many applications require a degree of "programmability" that is not met by current PHF constructions. Technically, we have PHF constructions with $a_X \neq 0$ for most, but not all preimages X. Such PHFs are suitable, e.g., in certain signature or identity-based encryption schemes [24, 18].

However, several prominent schemes that are formulated in the random oracle model (e.g., [23, 4, 6]) would require a PHF with $a_X = 0$ for most (but not all) preimages. (Roughly speaking, in these schemes, adversarial queries X can be handled iff the corresponding hash does not have a challenge component, i.e., if $a_X = 0$.) Unfortunately, a recent result [17] shows that no black-box construction of such a PHF with $a_X = 0$ for most (but not all) X exists.

Our Work. We construct PHFs with $a_X = 0$ for most (but not all) X by slightly adapting the PHF definition to a setting in which a multilinear map is available.[1] We use our PHFs to give standard-model versions of prominent cryptographic schemes whose security has so far only been proven in the random oracle model. Specifically, we give standard-model versions of the Boneh-Franklin (BF) identity-based encryption scheme [4], Boneh-Lynn-Shacham (BLS) signatures [6], and the Sakai-Ohgishi-Kasahara (SOK) identity-based non-interactive key exchange (ID-NIKE) [23]. We also use our PHFs to realise a completely new secure cryptographic functionality: we present the first *fully secure* hierarchical ID-NIKE, with security either in the standard-model or the random oracle model. Our constructions assume the existence of an $\mathbf{O}(k)$-linear map, where k is the security parameter.[2] We use an abstraction of multilinear maps that is compatible with the recent "noisy" candidate for multilinear maps of Garg, Gentry, and Halevi [13].

Some Technical Details. We circumvent the black-box impossibility result [17] by slightly adapting the PHF definition to a setting with multilinear maps. Intuitively, [17] uses that a_X is an exponent that can be viewed as a known function in certain unknown variables. This function is linear, because all involved group elements are from the same group, and only group operations are allowed. But the number of zeros of such a (nontrivial) linear function can be reasonably upper bounded. This contradicts the goal that $a_X = 0$ for many, but not all X.

[1] Concretely, we construct (poly, n)-MPHFs for any constant n. This denotes a slight variant of PHFs in a multilinear setting, with the following property. For any polynomial number of X_i and $Z_1, \ldots, Z_n$ (with $X_i \neq Z_j$), we have $a_{X_i} = 0$ and $a_{Z_j} \neq 0$ for all i, j with significant probability. The X_i, Z_j need not be known during setup.

[2] In fact, our optimizations only require a $\mathbf{O}(k/\log(k))$-linear map.

By moving to a multilinear setting, we essentially allow (a limited number of) multiplications in the exponent. Hence, the exponent a_X is now no longer limited to be a linear function, but can be a *multivariate* polynomial. Such polynomials can have exponentially many zeros. For instance, we could choose secret values $\alpha_{i,b}$ (for $1 \leq i \leq |X|$ and $b \in \{0,1\}$), such that exactly one element of each pair $(\alpha_{i,0}, \alpha_{i,1})$ is nonzero; say $\alpha_{i,b_i} \neq 0$. Then the function

$$a_X = \alpha(X) = \prod_{i=1}^{|X|} \alpha_{i,X_i} \tag{1}$$

(where X_i denotes the i-th bit of X) evaluates to zero everywhere except for $X = (b_1, \ldots, b_{|X|})$. In fact, we implement a suitable variant of the function in (1) in the exponent (in the sense that $\mathsf{H}(X) = c^{a_X} h^{b_X} = c^{\alpha(X)} h^{b_X}$ for a suitable blinding term h^{b_X}) through multilinear maps.[3] In the process, we also recognize and refine an admissible hash function (AHF [3, 8, 1]) implicit in [19]. This yields the – by far – most efficient known AHFs. As a result, we get PHFs in the multilinear setting with $a_X = 0$ for many (but not all) X.

Applications. To demonstrate their power, we use our new PHFs to replace random oracles in three example applications. As one application, we obtain from BLS signatures [6] an existing standard-model signature scheme due to Boneh and Silverberg [5]; as a natural extension, we give a standard-model variant of the Boneh-Franklin IBE scheme [4]. However, our central application is the SOK [23] ID-NIKE scheme; from this scheme, we get the first fully secure ID-NIKE in the standard model.

In all cases, the analysis is completely modular: we prove the security of the PHF-based schemes solely from generic PHF properties. In particular, we can also view (programmable) random oracles as PHFs to obtain the original schemes, with essentially the original proofs.[4] We view these results as strong evidence that PHFs are a useful abstraction of random oracles that also allows for standard-model instantiations.

In addition, we give natural *hierarchical* versions of all schemes in a setting with multilinear maps. (Recall that we require multilinear maps for our PHFs anyway.) Again, we can either use random oracles as PHFs to obtain reasonably efficient new schemes, or use our new PHFs to obtain (somewhat less efficient) standard-model versions.

More on Our ID-NIKE Schemes. In the signature and IBE applications, we mainly explain (and slightly improve) existing schemes through PHFs. While this already hints at the potential of our notion of PHFs, their actual usefulness

[3] We stress that these ideas are not new; essentially the same function in the exponent has been considered by Boneh and Silverberg [5] for a concrete signature scheme, building on work of Lysyanskaya [19]. Our contribution here is an abstraction (along with a few quantitative optimizations) that enables new applications.

[4] The exception is the SOK scheme, for which we only get a proof under a slightly stronger computational assumption.

in building novel cryptographic functionalities is best demonstrated by our application to ID-NIKE.

Loosely speaking, a non-interactive key exchange (NIKE) provides any two parties registered in the system with a unique shared key, *without any interaction*. For NIKE in the identity-based setting, there is a single master public key held by a trusted authority (TA); each party additionally gets an individual user secret key from the TA, and combines its secret key with the identity of the other party to compute the shared key. This primitive is a powerful one. For one thing, it implies secure IBE under a minor technical requirement [20]. More importantly, it has important applications in managing keys and enabling secure communications in mobile *ad hoc* and sensor networks, where the energy cost of communication is at a premium [14, 9]. In the hierarchical setting, H-ID-NIKE allows the same functionality, but also allows the TA's operations to be distributed over a hierarchy, which is well-suited to military and emergency response scenarios. The advantages of ID-NIKE, in terms of reducing communication costs and latency in a realistic adversarial environment, are demonstrated in [9]. For further discussion of applications of NIKE and ID-NIKE, see [14, 12].

However, ID-NIKE has proven surprisingly hard to instantiate in the standard model, even more so in a hierarchical setting. Currently, to the best of our knowledge, there is precisely one efficient, secure ID-NIKE scheme with a proof of security in the random oracle model, namely the SOK scheme [23] (with security models and analysis in [11, 20]). There are no schemes secure in the standard model. One might think that such schemes could easily be obtained from known standard-model-secure IBE schemes, but this is not the case; the essential technical barrier seems to be the randomised key derivation in these IBE schemes.

In the hierarchical setting, Gennaro *et al.* [14] constructed H-ID-NIKE schemes that are secure under certain classes of key exposure, but which do not offer *full security*, the desirable and natural generalisation of the existing ID-NIKE security notion from [20] to the hierarchical setting. Moreover, their schemes do not scale well to large numbers of levels. The same criticisms apply to earlier schemes [2, 21] on which the scheme of Gennaro *et al.* [14] is based. Indeed, one of the open problems left in [14] is to construct a H-ID-NIKE scheme with security against *not only* compromise of any number of leaves, *but also* against any number of nodes at higher levels of the hierarchy.[5]

By substituting the random oracles in the SOK scheme [23] with our new PHFs, we obtain the first secure ID-NIKE schemes in the standard model. Furthermore, our construction extends naturally to the hierarchical setting, yielding the first fully secure H-ID-NIKE schemes. The construction can be instantiated using random oracles to obtain a reasonably efficient scheme, or using PHFs for security in the standard model. In the full version, we also show how multilinear maps can be used to achieve security in the broader scenario of multiple TAs, and for shared keys among whole groups of parties.

[5] We note that there are other papers claiming to solve this open problem (*eg.* [16]), but these can be easily shown to provide insecure schemes.

Note on the Recent Candidate for Multilinear Maps. Recently, Garg, Gentry, and Halevi [13] have announced a candidate for a family of cryptographically interesting multilinear maps. Their candidate is lattice-based, heavily relies on the notion of noise, and thus does not provide groups in the usual sense. We comment on the necessary adaptations of our schemes to their setting inside.

2 Preliminaries

Notation. For $n \in \mathbb{R}$, let $[n] := \{1, \ldots, \lfloor n \rfloor\}$. Throughout the paper, $k \in \mathbb{N}$ denotes the security parameter. For a finite set $\mathcal{S}$, we denote by $s \leftarrow \mathcal{S}$ the process of sampling s uniformly from $\mathcal{S}$. For sets $\mathcal{S}^1, \mathcal{S}^2, \ldots$ and $n \in \mathbb{N}$, we write $\mathcal{S}^{\leq n} := \bigcup_{i=1}^{n} \mathcal{S}^i$. For a probabilistic algorithm A, we write $y \leftarrow A(x)$ for the process of running A on input x with uniformly chosen random coins, and assigning y the result. If A's running time is polynomial in k, then A is called probabilistic polynomial-time (PPT). A function $f : \mathbb{N} \to \mathbb{R}$ is negligible if it vanishes faster than the inverse of any polynomial (i.e., if $\forall c \exists k_0 \forall k \geq k_0 : |f(k)| \leq 1/k^c$). f is significant if it dominates the inverse of some polynomial (i.e., if $\exists c, k_0 \forall k \geq k_0 : f(k) \geq 1/k^c$).

Multilinear Maps. An ℓ-group system consists of ℓ cyclic groups $\mathbb{G}_1, \ldots, \mathbb{G}_\ell$ of prime order p, along with bilinear maps $e_{i,j} : \mathbb{G}_i \times \mathbb{G}_j \to \mathbb{G}_{i+j}$ for all $i, j \geq 1$ with $i + j \leq \ell$. Let g_i be a canonical generator of $\mathbb{G}_i$ (included in the group's description). The map $e_{i,j}$ satisfies $e_{i,j}(g_i^a, g_j^b) = g_{i+j}^{ab}$ (for all $a, b \in \mathbb{Z}_p$). When i, j are clear, we will simply write e instead of $e_{i,j}$. It will also be convenient to abbreviate $e(h_1, \ldots, h_j) := e(h_1, e(h_2, \ldots, e(h_{j-1}, h_j) \ldots))$ for $h_j \in \mathbb{G}_{i_j}$ and $i = (i_1 + i_2 + \ldots + i_j) \leq \ell$. By induction, it is easy to see that this map is j-linear. Additionally, we define $e(g) := g$. Finally, it can also be useful to define the group $\mathbb{G}_0 = \mathbb{Z}^+_{|\mathbb{G}_1|}$ of exponents to which this pairing family naturally extends. In the following, we will assume an ℓ-group system $\mathcal{MPG}_\ell = \{\{\mathbb{G}_i\}_{i \in [\ell]}, p, \{e_{i,j}\}_{i,j \geq 1, i+j \leq \ell}\}$ generated by a *multilinear maps parameter generator* $\mathcal{MG}_\ell$ on input a security parameter 1^k.

The GGH Candidate. We currently do not have candidates for multilinear maps between groups with cryptographically hard problems. However, Garg, Gentry, and Halevi [13] (henceforth GGH) suggest a concrete candidate for an "approximation" of multilinear maps, named *graded encoding systems*. With the GGH candidate, group elements have a randomized (and thus non-unique) representation dubbed "encoding". While it is possible to extract a unique "canonical bitstring" from an encoding, it is not possible to perform further computations with this extracted bitstring. An encoding can be re-randomized (e.g., to hide the sequence of operations that were performed), but only at the cost of introducing an artificial "noise" term in the encoding. Further operations (and re-randomizations) on this group element cause the noise to grow; once this noise grows beyond a certain bound, encodings can no longer be worked with.[6]

[6] We further ignore a (negligible) error probability in most of the GGH procedures. Technically, however, this leads to applications with, e.g., negligible correctness error.

Our Abstraction. For readability and universality, we will generally use the notation from the abstract notion of multilinear maps described above. When instantiated with the GGH candidate, operations are meant to occur on encodings, without implicit re-randomizations. In particular, e.g., g now denotes an encoding (not a group element). Additionally, we will employ the following notation to indicate necessary re-randomizations, extractions, and comparisons when using encodings instead of group elements.

- $g \leftarrow \mathbb{G}_i$ means choosing a random encoding g at level i. (This corresponds to uniformly choosing a group element from $\mathbb{G}_i$.) We assume that encodings g chosen in such a way have a low noise level, say, 1.
- $g \overset{\mathsf{enc}}{=} h$ holds iff the encodings g and h match.
- $g \overset{\mathsf{grp}}{=} h$ holds iff the *group elements* encoded by g and h match, that is, iff the GGH isZero procedure identifies $g^{-1}h$ as the neutral element.[7]
- $\mathsf{reRand}_j(g)$ is the re-randomization of encoding g. This re-randomization increases the noise level to a certain, a-priori fixed bound j. For simplicity, and abstracting, we only consider noise levels $j \in \mathbb{N}$. If g's noise level is already at least j (e.g., because g is the output of reRand_j), then randomization fails. We note that the distributions $\mathsf{reRand}_j(g)$ and $\mathsf{reRand}_j(h)$ are statistically close for any two encodings g, h with $g \overset{\mathsf{grp}}{=} h$ and noise level less than j.
- $\mathsf{ext}(g)$ denotes the canonical bitstring extracted from encoding g. We have $\mathsf{ext}(g) = \mathsf{ext}(h)$ for any g, h with $g \overset{\mathsf{grp}}{=} h$ of sufficiently small noise level.

Like [13], we omit parameters (such as noise bounds) to computations; asymptotic parameters can be derived from the suggestions in [13, Section 4.2].

Hard Problems. The ℓ-MDDH assumption is: given $(g, g^{x_1}, \ldots, g^{x_{\ell+1}})$, (for $g \leftarrow \mathbb{G}_1$ and uniform exponents x_i), the element $e(g^{x_1}, \ldots, g^{x_\ell})^{x_{\ell+1}} \in \mathbb{G}_\ell$ is computationally indistinguishable from a uniform $\mathbb{G}_\ell$-element. The $(\ell+1)$-power assumption is: given (g, g^x) (for $g \leftarrow \mathbb{G}_1$ and uniform x), the element $e(\underbrace{g^x, \ldots, g^x}_{\ell \text{ times}})^x \in \mathbb{G}_\ell$ is computationally indistinguishable from a uniformly chosen $\mathbb{G}_\ell$-element.[8]

3 Programmable Hash Functions in the Multilinear Setting

3.1 Motivation

Programmable hash functions (PHFs) have been defined in [18] as a special type of a group hash function (i.e., a hash function with images in a group).

[7] Technically, the GGH isZero procedure only allows to compare two encodings on the "highest level" ℓ. To compare two level-i encodings (for $i < \ell$), we can first "lift" both to level ℓ by pairing them with a nonzero level-$(\ell - i)$ element.

[8] We note that in the GGH setting, all elements g^{x_i} (resp. g^x), and the challenge $e(g^{x_1}, \ldots, g^{x_\ell})^{x_{\ell+1}})$ (resp. $e(g^x, \ldots, g^x)^x$) are produced with knowledge of the exponents x, x_i as fully randomized but low-noise encodings.

Namely, the image $\mathsf{H}(X)$ of a PHF can always be explained as $\mathsf{H}(X) = c^{a_X} h^{b_X}$ for externally given c, h. Usually, c will be a "challenge element" (e.g., from a Diffie-Hellman-like problem), and h will be a "controlled element" (e.g., with known exponent relative to a fixed group generator) used for blinding purposes. Intuitively, we require that both the events $a_X = 0$ and $a_X \neq 0$ occur with significant probability. Even more, an (m, n)-PHF guarantees that with significant probability, $a_{X_i} = 0$ for *any* m given inputs X_i, while $a_{Z_j} \neq 0$ for *any* n given inputs Z_j (with $X_i \neq Z_j$ of course). This means that the $\mathsf{H}(X_i)$ contain no challenge component, while all $\mathsf{H}(Z_j)$ do.

For our purposes, we will strive to construct efficient (poly, n)-PHFs for constant n (i.e., group hash functions which are $(q(k), n)$-PHFs for any polynomial q). However, there are indications that such PHFs do not exist [17], at least according to the original definition from [18]. Thus, we will adapt the definition of PHFs to the multilinear setting, and construct the "multilinear analog" of a (poly, n)-PHF. Concretely, an (m, n)-PHF maps to a "target" group $\mathbb{G}_\ell$. Here instead of explaining $\mathsf{H}(X)$ as a product $c^{a_X} h^{b_X}$ for c, h in the target group $\mathbb{G}_\ell$ (as the case of PHFs), we will explain $\mathsf{H}(X)$ as a product $e(c_1, \ldots, c_\ell)^{a_X} e(B_X, h)$, for externally given challenges $c_i \in \mathbb{G}_1$ (which means $c = e(c_1, \ldots, c_\ell) \in \mathbb{G}_\ell$) and controlled $h \in \mathbb{G}_1$. Note that the coefficient b_X in the usual definition of a PHF now becomes a preimage $B_X \in \mathbb{G}_{\ell-1}$ under a pairing operation.

3.2 Definitions

Definition 1 (Group hash function). *A group hash function* H *into* $\mathbb{G}$ *consists of two polynomial-time algorithms: the probabilistic algorithm* $\mathsf{HGen}(1^k)$ *outputs a key* hk*, and* $\mathsf{HEval}(hk, X)$ *(for a key* hk *and* $X \in \{0,1\}^k$*) deterministically outputs an image* $\mathsf{H}_{hk}(X) \in \mathbb{G}$*.*

Definition 2 (MPHF). *Assume an* ℓ'*-group system* $\mathcal{MPG}_{\ell'}$ *as generated by* $\mathcal{MG}_{\ell'}(1^k)$*. Let* H *be a group hash function into* $\mathbb{G}_\ell$ *(*$\ell \leq \ell'$*), and let* $m, n \in \mathbb{N}$*. We say that* H *is an* (m, n)*-programmable hash function in the multilinear setting (*(m, n)*-MPHF) if there are PPT algorithms* TGen *and* TEval *as follows.*

- $\mathsf{TGen}(1^k, c_1, \ldots, c_\ell, h)$ *(for* $c_i, h \in \mathbb{G}_1$ *and* $h \overset{\mathsf{grp}}{\neq} 1$*) outputs a key* hk *and a trapdoor* td*. We require that for all* c_i, h*, the distribution of* hk *is statistically close to the output of* HGen*.*[9]
- $\mathsf{TEval}(td, X)$ *(for a trapdoor* td *and* $X \in \{0,1\}^k$*) deterministically outputs* $a_X \in \mathbb{Z}$ *and* $B_X \in \mathbb{G}_{\ell-1}$ *with* $\mathsf{H}_{hk}(X) \overset{\mathsf{grp}}{=} e(c_1, \ldots, c_\ell)^{a_X} \cdot e(B_X, h)$*. We require that there is a polynomial* $p(k)$ *such that for all* hk *and* $X_1, \ldots, X_m$*,* $Z_1, \ldots, Z_n \in \{0,1\}^k$ *with* $\{X_i\}_i \cap \{Z_j\}_j = \emptyset$*,*

$$P_{hk,\{X_i\},\{Z_j\}} := \Pr\left[a_{X_1} = \cdots = a_{X_m} = 0 \wedge a_{Z_1}, \ldots, a_{Z_n} \neq 0\right] \geq 1/p(k), \quad (2)$$

[9] There is a subtlety here: in case of encoded group elements, the output of TGen may consist of group elements whose noise level depends on the noise level of the c_i or h. Hence, we will assume a known a-priori bound on the noise level of the c_i and h. This assumption will be fulfilled in our applications.

where the probability is over possible trapdoors td output by TGen *along with the given hk. Furthermore, we require that* $P_{hk,\{X_i\},\{Z_j\}}$ *is close to statistically independent of hk. (Formally,* $|P_{hk,\{X_i\},\{Z_j\}} - P_{hk',\{X_i\},\{Z_j\}}| \leq \nu(k)$ *for all* hk, hk' *in the range of* TGen*, all* $\{X_i\}, \{Z_j\}$*, and negligible* $\nu(k)$*.)*

We say that H *is a* (poly, n)*-MPHF if it is a* $(q(k), n)$*-MPHF for every polynomial* $q(k)$*, analogously for* (m, poly)*-MPHFs.*

Note that the TEval algorithm of an MPHF into $\mathbb{G}_1$ yields $B_X \in \mathbb{G}_0$, i.e., exponents B_X. In fact, in this case, the MPHF definition coincides with the original PHF definition from [18].

Readers interested only in how to use MPHFs in cryptographic constructions may safely skip the remainder of this section.

3.3 Warmup: Programmable Random Oracles as MPHFs

A *programmable* random oracle RO with images in $\mathbb{G}_1$ can be interpreted as a group hash function in the obvious way. (By "programmable", we mean that during a security proof, we can freely and adaptively determine images of RO, even depending on the inputs of TGen. The only restriction of this programming is that images should appear uniformly and independently distributed to an adversary who sees only public information.) However, note for this modeling to make sense in the first place, we should require that we can hash into $\mathbb{G}_1$.

Theorem 1 (PROs as (poly, n)**-MPHFs).** *A programmable random oracle* RO *(in the above sense) with images in* $\mathbb{G}_1$ *can be programmed to act as a* (poly, n)*-MPHF for any constant* n*.*

Proof (Proof sketch.). Fix a polynomial $q = q(k)$. We show that RO is a (q, n)-MPHF (with empty hk). For each new preimage X, we program $\mathsf{RO}(X) := c^{a_X} h^{B_X}$ for the inputs $c := c_1$ and h to TGen, and a uniformly chosen exponent $B_X \in \mathbb{G}_0 = \mathbb{Z}_{|\mathbb{G}_1|}$. We choose $a_X = 1$ with probability $1/2q$, and $a_X = 0$ otherwise. TEval outputs these a_X, B_X, assigning them as necessary for previously unqueried inputs X. For any pairwise different $X_1, \ldots, X_q, Z_1, \ldots, Z_n$, we thus have

$$\Pr\left[\forall i : a_{X_i} = 0 \wedge \forall j : a_{Z_j} \neq 0\right] = \left(1 - \frac{1}{2q}\right)^q \cdot \left(\frac{1}{2q}\right)^n \geq \frac{1}{2} \cdot \left(\frac{1}{2q}\right)^n,$$

which is significant for polynomial q and constant n. □

3.4 Ingredient: Efficient Admissible Hash Functions

At the heart of our standard-model constructions lies a primitive dubbed "admissible hash function" (AHF) [3]. Unfortunately, the AHFs from [3] are not very efficient (and in fact only achieve a weaker AHF definition, see [8]). However, luckily, an earlier work by Lysyanskaya [19] already contains an implicit and much more efficient AHF.

Intuitively, an AHF can be thought of as a combinatorial counterpart of $(\mathsf{poly}, 1)$-(M)PHFs. An AHF input X is mapped to an image $\mathsf{AHF}(X)$ in a way that X can fall in the set of controlled, $\mathtt{CO}$, inputs (meaning that we know a trapdoor that allows to answer adversary's queries for that input) or uncontrolled, $\mathtt{UN}$, inputs (meaning that we do not know any trapdoor but hope to embed a challenge element). (Unlike with (M)PHFs, however, this is a purely combinatorial property.) An AHF guarantees that for any $X_1, \ldots, X_q, Z$, with significant probability, all X_i are controlled, and Z is uncontrolled.

We now give a definition that is a somewhat simpler variant of the AHF definitions from [8, 1], and then show a result implicit in [19].

Definition 3 (AHF). *For a function* $\mathsf{AHF} : \{0,1\}^k \to R^\ell$ *(with a finite set*[10] R *and polynomial* $\ell = \ell(k)$*) and* $K \in (R \cup \{\perp\})^\ell$*, define the function* $F_K : \{0,1\}^k \to \{\mathtt{CO}, \mathtt{UN}\}$ *through* $F_K(X) = \mathtt{UN} \iff \forall i : K_i = \mathsf{AHF}(X)_i \vee K_i = \perp$*, where* $\mathsf{AHF}(X)_i$ *denotes the* i*-th component of* $\mathsf{AHF}(X)$*.*[11] *We say that* AHF *is* q*-admissible if there exists a PPT algorithm* KGen *and a polynomial* $p(k)$*, such that for all* $X_1, \ldots, X_q, Z \in \{0,1\}^k$ *with* $Z \notin \{X_i\}$*,*

$$\Pr\left[F_K(X_1) = \cdots = F_K(X_q) = \mathtt{CO} \wedge F_K(Z) = \mathtt{UN}\right] \geq 1/p(k), \tag{3}$$

where the probability is over $K \leftarrow \mathsf{KGen}(1^k)$*. We say that* AHF *is an admissible hash function (AHF) if* AHF *is* q*-admissible for all polynomials* $q = q(k)$*.*

Thus, X is controlled (i.e., $F_K(X) = \mathtt{CO}$) if there is an i with $X_i \neq K_i \neq \perp$.

Theorem 2 ([19]). *Assume a family of codes* $\{\mathcal{C}_k\}$ *with* $\mathcal{C}_k : \{0,1\}^k \to R^\ell$ *denoting both the code and its encoding function. Suppose that* $\mathcal{C}_k$ *has minimum distance at least* $c \cdot \ell$ *for a fixed constant* $c > 0$*. (That is,* $X_1 \neq X_2$ *implies that the vectors* $\mathcal{C}_k(X_1)$ *and* $\mathcal{C}_k(X_2)$ *differ in* $\geq c \cdot \ell$ *positions.) Then* $\{\mathcal{C}_k\}$ *is an AHF.*

Proof. Let $q = q(k)$ be a polynomial. We need to devise a PPT algorithm KGen such that (3) holds. $\mathsf{KGen}(1^k)$ sets $d := \lfloor (\ln 2q)/c \rfloor$ (so d is the smallest integer such that $(1-c)^d \leq 1/2q$), and picks K uniformly among all elements from $(R \cup \{\perp\})^\ell$ with exactly d non-$\perp$ components. Hence, the set $I := \{i \mid K_i \neq \perp\}$ is of size d.

Now fix $X_1, \ldots, X_q, Z \in \{0,1\}^k$ with $Z \notin \{X_i\}$. Our choice of K implies $\Pr[F_K(Z) = \mathtt{UN}] = |R|^{-d}$. For any fixed i, we want to upper bound the probability $\Pr[F_K(X_i) = \mathtt{UN} \mid F_K(Z) = \mathtt{UN}]$. (This step loosely corresponds to [19, Lemma 4].) Hence, assume $F_K(Z) = \mathtt{UN}$; note that this conditioning leaves the distribution of I uniform. Now $\mathcal{C}_k(X_i)$ and $\mathcal{C}_k(Z)$ differ in a set $\Delta \subseteq [\ell]$ of positions with $|\Delta| \geq c\ell$. Hence, $F_K(X_i) = \mathtt{UN}$ is equivalent to $I \cap \Delta = \emptyset$. Thus,

$$\begin{aligned} \Pr\left[F_K(X_i) = \mathtt{UN} \mid F_K(Z) = \mathtt{UN}\right] &= \Pr\left[I \cap \Delta = \emptyset \mid F_K(Z) = \mathtt{UN}\right] \\ &\leq (1-c)^d \leq e^{-cd} \leq \frac{1}{2q}. \end{aligned}$$

[10] One should have $R = \{0,1\}$ in mind here. Larger R (e.g., $R = [k]$) lead to slightly less pairing-intensive constructions of MPHFs, see the paragraph before Theorem 4.

[11] That is, for $R = \{0,1\}$, we have $F_K(X) = \mathtt{CO}$ iff there is an i with $K_i = 1 - \mathsf{AHF}(X)_i$.

A union bound over i gives $\Pr[\forall i : F_K(X_i) = \mathtt{UN} \mid F_K(Z) = \mathtt{UN}] \leq 1/2$, so that

$$\Pr[F_K(Z) = \mathtt{UN} \wedge \forall i : F_K(X_i) = \mathtt{CO}] \;\geq\; \frac{1}{2} \cdot |R|^{-d} \;\geq\; \frac{1}{2} \cdot \left(\frac{1}{2q}\right)^{\frac{1}{c \cdot \log_{|R|}(e)}},$$

which is significant. □

3.5 Main Result: MPHFs from Multilinear Maps

Our main result in this section is a simple construction of a (poly, n)-MPHF from an AHF.

Construction 1 (MM). Let $\mathsf{AHF} : \{0,1\}^k \to R^\ell$ be an admissible hash function and assume an ℓ'-group system $\mathcal{MPG}_{\ell'}$. The group hash function MM into $\mathbb{G}_\ell$ ($\ell \leq \ell'$) is given by the following algorithms:

- $\mathsf{HGen}(1^k)$ picks $\tilde{h}_{i,j} \leftarrow \mathbb{G}_1 \backslash \{1\}$ (for $(i,j) \in [\ell] \times R$), sets $h_{i,j} := \mathsf{reRand}_2(\tilde{h}_{i,j})$, and outputs $hk := \{h_{i,j}\}_{i \in [\ell], j \in R}$. [12]
- $\mathsf{HEval}(hk, X)$ computes $(t_1, \ldots, t_\ell) := \mathsf{AHF}(X)$ and outputs $\mathsf{MM}_{hk}(X) := e(h_{1,t_1}, \ldots, h_{\ell,t_\ell})$.

Theorem 3. *The group hash function* MM *above is a* $(\mathsf{poly}, 1)$*-MPHF.*

Proof. Fix a polynomial $q = q(k)$. We need to exhibit TGen and TEval algorithms as in Definition 2. $\mathsf{TGen}(1^k, c_1, \ldots, c_\ell, h)$ invokes $K \leftarrow \mathsf{KGen}(1^k)$ and, for all $(i,j) \in [\ell] \times R$ and uniform exponents $r_{i,j} \neq 0$, it sets up

$$h_{i,j} := \begin{cases} \mathsf{reRand}_2(h^{r_{i,j}}) & \text{if } K_i \neq j \text{ and } K_i \neq \perp, \\ \mathsf{reRand}_2(c_i^{r_{i,j}}) & \text{if } K_i = j \text{ or } K_i = \perp. \end{cases} \tag{4}$$

For now, assume $c_i \overset{\mathsf{grp}}{\neq} 1$ for all i, so our setup yields a perfectly distributed key $hk := \{h_{i,j}\}_{i,j}$ that is in fact independent of K.[13] The trapdoor is $td := ((c_i), h, K, (r_{i,j}))$.

$\mathsf{TEval}(td, X)$ computes $(t_1, \ldots, t_\ell) := \mathsf{AHF}(X)$ and distinguishes two cases:

Case $F_K(X) = \mathtt{CO}$**,** i.e., there is at least an i^* with $K_{i^*} \neq t_{i^*}$ and $K_{i^*} \neq \perp$. If we set $a_X = 0$ and

$$B_X := e(h_{1,t_1}, \ldots, h_{i^*-1,t_{i^*-1}}, h_{i^*+1,t_{i^*+1}}, \ldots, h_{\ell,t_\ell})^{r_{i^*,t_{i^*}}},$$

for any chosen i^*, we can decompose $\mathsf{MM}_{hk}(X) \overset{\mathsf{grp}}{=} e(c_1, \ldots, c_\ell)^{a_X} e(B_X, h)$.

[12] The additional re-randomization step guarantees that the noise levels in scheme and simulation are the same. The concrete noise level of re-randomized elements depends on the maximal noise considered in the arguments of TGen.

[13] In case of randomized encodings, the distribution of hk in the simulation may (e.g., with the GGH candidate) only be statistically close to the one in the scheme.

Case $F_K(X) = \mathtt{UN}$**,** i.e., $K_i = t_i$ or $K_i = \perp$ for all i. This means that $h_{i,t_i} \stackrel{\mathsf{grp}}{=} c_i^{r_{i,t_i}}$ for all i, so $\mathsf{MM}_{hk}(X) \stackrel{\mathsf{grp}}{=} e(c_1,\ldots,c_\ell)^{a_X} e(B_X, h)$ for $a_X = \prod_i r_{i,t_i}$ and $B_X := 1$.

The AHF property (3) implies (2). (Note that $P_{hk,\{X_i\},\{Z\}}$ only depends on K but not on hk.)

Finally, in case $c_i \stackrel{\mathsf{grp}}{=} 1$ for some i, we have $e(c_1,\ldots,c_\ell) \stackrel{\mathsf{grp}}{=} 1$. If we replace *all* c_i in (4) with h, we can explain any image $\mathsf{MM}_{hk}(X) \stackrel{\mathsf{grp}}{=} e(h,\ldots,h)^{\prod_i r_{i,t_i}}$ as $\mathsf{MM}_{hk} \stackrel{\mathsf{grp}}{=} e(c_1,\ldots,c_k)^{a_X} e(B_X,h)$ with *arbitrary* a_X. Adjusting the probability for $a_X \neq 0$ in the order of $1/2q$ (as in the proof of Theorem 1) allows to prove (2) for $p(k) = 2 \cdot (2q)^n$. □

Examples. For $R = \{0,1\}$ and binary codes $\mathcal{C}_k : \{0,1\}^k \to R^\ell$ with large minimum distance, we get the AHF implicit in [19]. This yields MPHFs that use $\mathbf{O}(k)$ groups $\mathbb{G}_i$, and have keys of $2k$ group elements. Larger R give new AHFs that yield MPHFs that use fewer groups, but have larger keys. For instance, with $R = \mathbb{F}_{2^\kappa}$, for $\kappa := \lfloor \log_2(k) \rfloor$, along with MDS codes over R, we obtain MPHFs that use $\mathbf{O}(k/\log_2(k))$ groups, and have keys consisting of k^2 group elements.

Theorem 4. *Let n be a constant, $q = q(k)$ be a polynomial, and let $\mathsf{H} = (\mathsf{HGen}, \mathsf{HEval})$ be a $(q+n-1,1)$-MPHF into $\mathbb{G}_\ell$. Then the group hash function $\mathsf{H}' = (\mathsf{HGen}', \mathsf{HEval}')$ with*

- $\mathsf{HGen}'(1^k)$ *that outputs $hk' = (hk_\nu)_{\nu\in[n]}$ for $hk_\nu \leftarrow \mathsf{HGen}(1^k)$, and*
- $\mathsf{HEval}'(hk', X)$ *that outputs $\mathsf{H}'_{hk'}(X) := \prod_{\nu\in[n]} \mathsf{H}_{hk_\nu}(X)$*

is a (q,n)-MPHF into $\mathbb{G}_\ell$.

Combining Theorems 3 and 4 yields a (poly, n)-MPHF for any constant n.

Proof. We construct suitable TGen' and TEval' algorithms from the respective TGen and TEval algorithms for H:

- $\mathsf{TGen}'(1^k, c_1,\ldots,c_\ell,h)$ runs $(hk_\nu, td_\nu) \leftarrow \mathsf{TGen}(1^k, c_1,\ldots,c_\ell,h)$ for $\nu \in [n]$, and outputs $hk' := (hk_\nu)_{\nu\in[n]}$ and $td' := (td_\nu)_{\nu\in[n]}$.
- $\mathsf{TEval}'(hk', X)$ invokes $(a_{\nu,X}, B_{\nu,X})) \leftarrow \mathsf{TEval}(td_\nu, X)$ and outputs $a_X := \sum_{\nu\in[n]} a_{\nu,X}$ and $B_X := \prod_{\nu\in[n]} B_{\nu,X}$. This output can be justified with

$$\mathsf{H}'_{hk'}(X) \stackrel{\mathsf{grp}}{=} \prod_{\nu\in[n]} \mathsf{H}_{hk_\nu}(X) \stackrel{\mathsf{grp}}{=} \prod_{\nu\in[n]} e(c_1,\ldots,c_\ell)^{a_{\nu,X}} e(B_{\nu,X},h) \stackrel{\mathsf{grp}}{=} e(c_1,\ldots,c_\ell)^{a_X} e(B_X,h).$$

Now fix $X_1,\ldots,X_q,Z_1,\ldots,Z_n$ with $\{X_i\} \cap \{Z_j\} = \emptyset$. For each ν, we hope for the following event: $a_{\nu,X_i} = 0$ for all i, and $a_{\nu,Z_j} = 0$ exactly for $j \neq \nu$. For fixed ν, this event happens with probability at least $1/p(k)$ (over td_ν) for some polynomial p. Since $a_X = \sum_\nu a_{\nu,X}$, we get that with probability at least $(1/p(k))^n$, we have $a_{X_i} = 0$ for all i and $a_{Z_j} = a_{j,Z_j} \neq 0$ for all j. □

4 (Hierarchical) ID-Based Non-interactive Key Exchange

Hierarchical identity-based non-interactive key exchange (H-ID-NIKE) is the natural generalisation of ID-NIKE [23, 11, 20] to the hierarchical setting: a root authority calculates and distributes private keys to sub-authorities, who in turn do the same for sub-sub-authorities, and so on, until leaf nodes are reached. Each node is identified by a vector of identities, and any pair of nodes in the tree should be able to non-interactively compute a common key based on their private keys and identities. We recall from the introduction that H-ID-NIKE schemes are rare, and, to the best of our knowledge, there are not even any ROM constructions that meet all the desirable criteria (efficiency, scalability, and full security in the sense of resilience to arbitrary node compromises).

Formally, an H-ID-NIKE scheme $\mathsf{H\text{-}ID\text{-}NIKE}$ consists of three PPT algorithms (see below), an identity space $\mathcal{ID}$ and shared-key space $\mathcal{SHK}$. The users are organized in a tree of depth L whose root (at level 0) is the trusted authority (TA). The identity of a user at level $d \in [L]$ is represented by a vector $\mathbf{id} = (id_1, \ldots, id_d) \in \mathcal{ID}^d$.

Setup. The setup algorithm $\mathsf{Setup}(1^k, L)$ is run by the TA. Given the security parameter 1^k and a parameter $L \in \mathbb{N}$, it outputs a master public key mpk and a master secret key msk. We also interpret msk as the user secret key usk_ε for the empty identity ε.

Key Delegation. The key delegation algorithm $\mathsf{Del}(mpk, usk_{\mathbf{id}}, \mathbf{id}')$ can be run by any user to generate a secret key for any of its children. Given the master public key mpk, the user secret key $usk_{\mathbf{id}}$ for an identity $\mathbf{id} = (id_1, \ldots, id_d) \in \mathcal{ID}^d$, the algorithm outputs a user secret key $usk_{\mathbf{id}'}$ for any of its children $\mathbf{id}' = (id_1, \ldots, id_\ell, id_{d+1}) \in \mathcal{ID}^{d+1}$ (for $0 \leq d < L$).

Shared Key Generation. Given the master public key mpk, a user secret key $usk_{\mathbf{id}_1}$ for an identity $\mathbf{id}_1 \in \mathcal{ID}^{\leq L}$, and an identity $\mathbf{id}_2 \in \mathcal{ID}^{\leq L}$, $\mathsf{ShK}(mpk, usk_{\mathbf{id}_1}, \mathbf{id}_2)$ outputs either a shared key $K_{\mathbf{id}_1,\mathbf{id}_2} \in \mathcal{SHK}$ or a failure symbol $\perp$. (If $\mathbf{id}_1$ is an ancestor of $\mathbf{id}_2$ (or vice-versa) the algorithm is assumed to always output $\perp$ [14]; here, $\mathbf{id}$ is in particular considered to be an ancestor of itself. Otherwise the output is assumed to be in $\mathcal{SHK}$.)

For correctness, we require that for any $k, L \in \mathbb{N}$, for any $(mpk, msk) \leftarrow \mathsf{Setup}(1^k, L)$, for any pair of identities $(\mathbf{id}_1, \mathbf{id}_2) \in \mathcal{ID}^{d_1} \times \mathcal{ID}^{d_2}$, such that neither is an ancestor of the other, and corresponding user secret keys $usk_{\mathbf{id}_1}$ and $usk_{\mathbf{id}_2}$ generated by repeated applications of Del from $usk_\varepsilon = msk$, we have $\mathsf{ShK}(mpk, usk_{\mathbf{id}_1}, \mathbf{id}_2) = \mathsf{ShK}(mpk, usk_{\mathbf{id}_2}, \mathbf{id}_1)$.

A (non-hierarchical) ID-NIKE scheme is a H-ID-NIKE scheme in which the depth L of the tree is fixed to $L = 1$. (Note that in this case, Del gets as input $usk_\varepsilon = msk$ and outputs user secret keys for level-1 identities. We may thus also speak of extraction of user secret keys.)

[14] If $\mathbf{id}_1$ is an ancestor of $\mathbf{id}_2$, it can always compute the user secret key $usk_{\mathbf{id}_2}$; a key derived from $usk_{\mathbf{id}_2}$ can be used as a shared key between the two users.

4.1 Security Definition for (H-)ID-NIKE

We present a security model for H-ID-NIKE that is the natural generalisation of the *PS* model for ID-NIKE from [20] to the hierarchical setting. The model significantly strengthens the previous model of Gennaro *et al.* [14] by being fully adaptive, allowing arbitrary numbers of node corruptions, and allowing the adversary access to shared keys as well as user secret keys of inner (i.e., non-leaf) nodes. The model is defined in terms of a game between an adversary A and a challenger C. C takes as input the security parameter 1^k and a depth L, runs algorithm Setup of the H-ID-NIKE scheme and gives A the master public key mpk. It keeps the master secret key, msk, to itself. A then makes queries of the following three types:

***Extract*:** A supplies an identity $\mathbf{id} = (id_1, \ldots, id_d) \in \mathcal{ID}^d$ (for $d \in [L]$). C uses Del repeatedly, starting from msk, to derive $usk_{\mathbf{id}}$ and hands $usk_{\mathbf{id}}$ to A.

***Reveal*:** Here A supplies a pair $(\mathbf{id}_1, \mathbf{id}_2) \in \mathcal{ID}^{d_1} \times \mathcal{ID}^{d_2}$. C extracts $usk_{\mathbf{id}_1}$ as above, runs $K_{\mathbf{id}_1,\mathbf{id}_2} \leftarrow \mathsf{ShK}(mpk, usk_{\mathbf{id}_1}, \mathbf{id}_2)$, and hands $K_{\mathbf{id}_1,\mathbf{id}_2}$ to A.

***Test*:** A supplies two *target* identities $(\mathbf{id}_1^*, \mathbf{id}_2^*) \in \mathcal{ID}^{d_1} \times \mathcal{ID}^{d_2}$ such that neither is an ancestor of the other. C computes $K_{\mathbf{id}_1^*,\mathbf{id}_2^*}$ as above, and tosses a coin $b \leftarrow \{0,1\}$. If $b = 0$ then C gives $K_{\mathbf{id}_1^*,\mathbf{id}_2^*}$ to A; otherwise, if $b = 1$, then C gives A a uniform element from $\mathcal{SHK}$.

Finally, A outputs a guess $\hat{b}$ for b. In our security model, the adversary is allowed to make an arbitrary (but polynomial) number of ***Extract*** and ***Reveal*** queries. Furthermore, the adversary is fully adaptive, in the sense that it can compromise nodes (by making ***Extract*** and/or ***Reveal*** queries) in any order. In order to prevent the adversary from trivially winning, we require that the adversary is not allowed to make any ***Extract*** queries on an ancestor of $\mathbf{id}_1^*$ or $\mathbf{id}_2^*$, and no ***Reveal*** query on the pairs $(\mathbf{id}_1^*, \mathbf{id}_2^*)$ and $(\mathbf{id}_2^*, \mathbf{id}_1^*)$. The advantage of an adversary A against a H-ID-NIKE scheme $\mathsf{H\text{-}ID\text{-}NIKE}$ is

$$\begin{aligned} \mathrm{Adv}^{\text{IND-SK}}_{A,\mathsf{H\text{-}ID\text{-}NIKE}}(k) &= \left|\Pr[\hat{b} = b] - 1/2\right| \\ &= 1/2 \left|\Pr\left[\hat{b} = 1 \mid b = 1\right] - \Pr\left[\hat{b} = 1 \mid b = 0\right]\right|. \end{aligned}$$

We say that $\mathsf{H\text{-}ID\text{-}NIKE}$ is IND-SK secure iff $\mathrm{Adv}^{\text{IND-SK}}_{A,\mathsf{H\text{-}ID\text{-}NIKE}}(k)$ is negligible for all PPT adversaries A.

In the non-hierarchical case (i.e., $L = 1$), we recover the definition and security model for (non-hierarchical) ID-NIKE from [20]. Note also that versions of these models in which multiple ***Test*** queries are permitted for a single bit b can be shown to be polynomially equivalent to the versions with a single ***Test*** query using standard hybrid arguments.

4.2 Fully-Secure ID-NIKE from MPHFs

In this section we revisit the ID-NIKE scheme of Sakai, Ohgishi and Kasahara (SOK) [23]. We replace random oracles with $(\mathsf{poly}, 2)$-MPHFs in their scheme

and prove security of the generalized scheme. Using our standard-model MPHFs, this yields the first standard-model ID-NIKE scheme.[15] We then consider a hierarchical generalisation.

We assume a 2ℓ-group system $\mathcal{MPG}_{2\ell} = \{\{\mathbb{G}_i\}_{i\in[2\ell]}, p, \{e_{i,j}\}_{i,j\geq 1, i+j\leq 2\ell}\}$ generated by a multilinear maps parameter generator $\mathcal{MG}_{2\ell}(1^k)$, and a $(\mathsf{poly}, 2)$-MPHF $\mathsf{H} = (\mathsf{HGen}, \mathsf{HEval})$ with input length in $\{0,1\}^k$ and output in $\mathbb{G}_\ell$. The component algorithms of our ID-NIKE scheme $\mathsf{IDNIKE}_{\mathsf{MPHF}}$ are then defined in Figure 1. (For compatibility with existing notation, we present an extraction algorithm Ext instead of an equivalent delegation algorithm.) Correctness of the scheme is easy to verify.

Algorithm $\mathsf{Setup}(1^k)$
$\mathcal{MPG}_{2\ell} \leftarrow \mathcal{MG}_{2\ell}(1^k)$
$x \leftarrow \mathbb{Z}_p$, $hk \leftarrow \mathsf{HGen}(1^k)$
$mpk := (\mathcal{MPG}_{2\ell}, hk)$, $msk := x$
return (mpk, msk)

Algorithm $\mathsf{Ext}(mpk, msk, id)$
$usk_{id} \leftarrow \mathsf{reRand}_3(\mathsf{H}_{hk}(id)^{msk})$
return usk_{id}

Algorithm $\mathsf{ShK}(mpk, usk_{id_1}, id_2)$
$K_{id_1,id_2} := \mathsf{ext}(e(usk_{id_1}, \mathsf{H}_{hk}(id_2)))$
return K_{id_1,id_2}

Fig. 1. The ID-NIKE scheme $\mathsf{IDNIKE}_{\mathsf{MPHF}}$

Theorem 5 (Security of the MPHF-based ID-NIKE scheme). *Assume* H *is a* $(\mathsf{poly}, 2)$*-MPHF into* $\mathbb{G}_\ell$*. Then* $\mathsf{IDNIKE}_{\mathsf{MPHF}}$ *is IND-SK secure under the* $(2\ell+1)$*-power assumption.*

Proof. See the full version (`http://eprint.iacr.org/2013/354.pdf`).

A Variant Secure under a Weaker Assumption. We can also construct an ID-NIKE scheme in the standard model using two instances (with keys hk_1, hk_2) of a $(\mathsf{poly}, 1)$-MPHF instead of a single instance of a $(\mathsf{poly}, 2)$-MPHF. Shared keys are computed as $K := \mathsf{ext}(e(\mathsf{H}_{hk_1}(id_1)^{msk}, \mathsf{H}_{hk_2}(id_2)))$; user secret keys are of the form $usk_{id} = (\mathsf{reRand}_3(\mathsf{H}_{hk_1}(id)^{msk}), \mathsf{reRand}_3(\mathsf{H}_{hk_2}(id)^{msk}))$. The benefit of this variant is that it is possible to prove security under the 2ℓ-MDDH assumption (as opposed to the potentially stronger $(2\ell+1)$-power assumption we use above).

4.3 Extension to H-ID-NIKE

We can extend our ID-NIKE scheme to a H-ID-NIKE scheme of constant depth L. To this end, we work in a $2\ell L$-group system $\mathcal{MPG}_{2\ell L}$, and use L instances of a $(\mathsf{poly}, 2)$-MPHF H into $\mathbb{G}_\ell$. The resulting H-ID-NIKE scheme, denoted by

[15] If we instantiate the MPHFs again with random oracles (using Theorem 1), we retrieve the original SOK scheme in pairing-friendly groups, along with a security proof. However, we note that our security proof uses a different, seemingly stronger computational assumption.

$\mathsf{HIDNIKE}_{\mathsf{MPHF}}$, is given in Figure 2. In that description, and in the following, we write $\mathbf{id}_{\lceil i} := (id_1, \ldots, id_i)$ for an identity $\mathbf{id} = (id_1, \ldots, id_d)$ and $i \leq d$. We assume that all involved identities (including "shortened identities" $\mathbf{id}_{\lceil i}$) can be uniquely encoded as k-bit strings. (If this is not the case, we can always first apply a collision-resistant hash function.)

Algorithm $\mathsf{Setup}(1^k, L)$
$\mathcal{MPG}_{2\ell L} \leftarrow \mathcal{MG}_{2\ell L}(1^k)$
$x \leftarrow \mathbb{Z}_p, \tilde{u} \leftarrow \mathbb{G}_\ell, u \leftarrow \mathsf{reRand}_2(\tilde{u})$
$hk_i \leftarrow \mathsf{HGen}(1^k)(i \in [L])$; $msk := x$
$mpk := (\mathcal{MPG}_{2\ell L}, \{hk_i\}_{i\in[L]}, u)$
return (mpk, msk)

Algorithm $\mathsf{Del}(mpk, usk_{\mathbf{id}}, \mathbf{id}')$
parse $\mathbf{id}' =: (id_1, \ldots, id_{d+1})$
if $\mathbf{id} \neq (id_1, \ldots, id_d)$ return $\perp$
$usk_{\mathbf{id}'} \leftarrow \mathsf{reRand}_{d+2}(e(usk_{\mathbf{id}}, \mathsf{H}_{hk_{d+1}}(\mathbf{id}'_{\lceil d+1}))))$
return $usk_{\mathbf{id}'}$

Algorithm $\mathsf{ShK}(mpk, usk_{\mathbf{id}_1}, \mathbf{id}_2)$
$Y_{\mathbf{id}_2} := e(\mathsf{H}_{hk_1}(\mathbf{id}_{2,\lceil 1}), \ldots, \mathsf{H}_{hk_{d_2}}(\mathbf{id}_{2,\lceil d_2}))$
$K_{\mathbf{id}_1,\mathbf{id}_2} := \mathsf{ext}(e(usk_{\mathbf{id}_1}, Y_{\mathbf{id}_2}, \underbrace{u, \ldots, u}_{2L-d_1-d_2 \text{ times}}))$
return $K_{\mathbf{id}_1,\mathbf{id}_2}$

Fig. 2. The H-ID-NIKE scheme $\mathsf{HIDNIKE}_{\mathsf{MPHF}}$

Note. $msk = usk_\varepsilon = x \in \mathbb{Z}_p = \mathbb{G}_0$, so Del can be used to derive level-1 user secret keys from msk. (Recall that our definition of e is consistent with the implicit exponent group $\mathbb{G}_0 = \mathbb{Z}_p$; e.g., $e(x, g) = g^x$ for $x \in \mathbb{G}_0$.)

Theorem 6 (Security of the MPHF-based H-ID-NIKE scheme). *Let* H *be a* $(\mathsf{poly}, 2)$*-MPHF into* $\mathbb{G}_\ell$*. For fixed depth* $L \in \mathbb{N}$*,* $\mathsf{HIDNIKE}_{\mathsf{MPHF}}$ *is secure under the* $(2\ell L + 1)$*-power assumption.*

Proof. See the full version (`http://eprint.iacr.org/2013/354.pdf`).

A More Efficient Variant in the Random Oracle Model. We can replace the $2\ell L$-group system with a $2L$-group system and the L different MPHFs with a random oracle hashing into $\mathbb{G}_1$ in the above scheme $\mathsf{HIDNIKE}_{\mathsf{MPHF}}$ to obtain a second H-ID-NIKE scheme which can be proven secure in the random oracle model. In this case, the $2L$-group system can be instantiated with smaller parameters than the $2\ell L$-group system required in our standard model scheme.

Security with Multiple TAs and Group-ID-NIKE. We can also achieve security in the more general setting of multiple trusted authorities and shared keys that can be computed by groups of parties instead of just pairs. The details can be found in the full version.

5 IBE and Signature Schemes from MPHFs

Identity-Based Encryption. An identity-based encryption (IBE) scheme IBE with identity space $\mathcal{ID}$ and message space $\mathcal{M}$ consists of four PPT algorithms:

$\mathsf{Gen}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec}$. Key generation $\mathsf{Gen}(1^k)$, on input a security parameter 1^k, outputs a master public key mpk and a master secret key msk. Key extraction $\mathsf{Ext}(msk, id)$, given msk and an identity $id \in \mathcal{ID}$, outputs a user secret key usk_{id}. Encryption $\mathsf{Enc}(mpk, id, M)$, given mpk, an identity $id \in \mathcal{ID}$, and a message $M \in \mathcal{M}$, outputs a ciphertext C. Decryption $\mathsf{Dec}(usk_{id}, C)$, given usk_{id} and a ciphertext C, outputs a message $M \in \mathcal{M} \cup \{\perp\}$. For correctness, we require that for any $k \in \mathbb{N}$, all $(mpk, msk) \leftarrow \mathsf{Gen}(1^k)$, all $id \in \mathcal{ID}$, all $usk_{id} \leftarrow \mathsf{Ext}(msk, id)$, all $M \in \mathcal{M}$, and all $C \leftarrow \mathsf{Enc}(mpk, id, M)$, Dec satisfies $\mathsf{Dec}(usk_{id}, C) = M$.

IBE-IND-CPA Security. An IBE scheme IBE as above is IBE-IND-CPA secure iff every PPT adversary A succeeds in the following experiment with probability at most negligibly larger than $1/2$. First, A gets an honestly generated master public key mpk; in all of the following, A has access to an $\mathsf{Ext}(msk, \cdot)$ oracle for the corresponding msk. Next, A selects an identity $id^* \in \mathcal{ID}$ and two equal-length messages $M_0, M_1 \in \mathcal{M}$. The experiment then computes $C^* \leftarrow \mathsf{Enc}(mpk, id^*, M_b)$ for uniformly chosen $b \leftarrow \{0, 1\}$ and sends C^* to A. Finally, A outputs a guess b' and succeeds iff $b = b'$ and it has not queried Ext with id^*.

IBE from $(\mathsf{poly}, 1)$-MPHFs. Figure 3 depicts $\mathsf{IBE}_{\mathsf{MPHF}}$, which is the Boneh-Franklin IBE scheme [4], implemented with $(\mathsf{poly}, 1)$-MPHFs. Message and identity space are $\mathcal{M} = \mathcal{ID} = \{0,1\}^k$. We assume an $(\ell+1)$-group system $\mathcal{MPG}_{\ell+1} = \{\{\mathbb{G}_i\}_{i \in [\ell+1]}, p, \{e_{i,j}\}_{i,j \geq 1, i+j \leq \ell+1}\}$ generated by a multilinear maps parameter generator $\mathcal{MG}_{\ell+1}(1^k)$, and a $(\mathsf{poly}, 1)$-MPHF H into $\mathbb{G}_\ell$. If we take a random oracle as $(\mathsf{poly}, 1)$-MPHF (as in Theorem 1), then $\ell = 1$, and we get the original BF scheme. Correctness of $\mathsf{IBE}_{\mathsf{MPHF}}$ is easy to verify.

Algorithm $\mathsf{Gen}(1^k)$
$\mathcal{MPG}_{\ell+1} \leftarrow \mathcal{MG}_{\ell+1}(1^k)$
$hk \leftarrow \mathsf{HGen}(1^k), h \leftarrow \mathbb{G}_1, x \leftarrow \mathbb{Z}_p$
$mpk := (\mathcal{MPG}_{\ell+1}, hk, h, \mathsf{reRand}_2(h^x))$
$msk := (hk, x)$
return (mpk, msk)

Algorithm $\mathsf{Ext}(msk, id)$
parse $msk =: (hk, x)$
return $usk_{id} := \mathsf{reRand}_3(\mathsf{H}_{hk}(id)^x)$

Algorithm $\mathsf{Enc}(mpk, id, M)$
parse $mpk =: (\mathcal{MPG}_{\ell+1}, hk, h, \tilde{h})$
$r \leftarrow \mathbb{Z}_p$
$C :=$
$(\mathsf{reRand}_2(h_1^r), \mathsf{ext}(e(\mathsf{H}_{hk}(id), \tilde{h})^r) \oplus M)$
return C

Algorithm $\mathsf{Dec}(usk_{id}, C)$
parse $C =: (C_1, C_2)$
return $M := C_2 \oplus \mathsf{ext}(e(usk_{id}, C_1))$

Fig. 3. The IBE scheme from $(\mathsf{poly}, 1)$-MPHFs

Theorem 7. *Assume $\mathsf{IBE}_{\mathsf{MPHF}}$ is implemented in an $(\ell+1)$-group system, and with a $(\mathsf{poly}, 1)$-MPHF H into $\mathbb{G}_\ell$. Then, under the $(\ell+1)$-MDDH assumption, $\mathsf{IBE}_{\mathsf{MPHF}}$ is IBE-IND-CPA-secure.*

Proof. See the full version (`http://eprint.iacr.org/2013/354.pdf`).

Extension to HIBE. We can extend the above IBE scheme to a hierarchical IBE (HIBE) scheme of constant depth D. This generalization works similarly as in the ID-NIKE case. Hence, due to space constraints, we postpone a more detailed exposition to the full version of this paper.

(Hierarchical) Signatures from $(\mathsf{poly}, 1)$**-MPHFs.** We can convert any (H)IBE scheme into a (hierarchical) signature scheme using the techniques of [4, 15, 10]. If we apply this transformation to $\mathsf{IBE}_{\mathsf{MPHF}}$ above, we obtain an abstraction of BLS signatures [7]. Indeed, if we instantiate the involved MPHF with a random oracle, we get the original BLS scheme. On the other hand, if we use the standard-model MPHF from Theorem 3, we obtain (a slight variant of) the signature scheme of Boneh and Silverberg [5]. In fact, with suitable parameters (i.e., a larger R, see Section 3.5), we obtain a signature scheme that uses only $\mathbf{O}(k/\log(k))$ groups and multilinear operations (as opposed to $\mathbf{O}(k)$ groups and multilinear operations in the Boneh-Silverberg scheme). It seems natural to expect that, using the techniques of [22], this also yields an aggregatable signature scheme.

References

[1] Abdalla, M., Fiore, D., Lyubashevsky, V.: From selective to full security: Semi-generic transformations in the standard model. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 316–333. Springer, Heidelberg (2012)

[2] Blundo, C., Santis, A.D., Herzberg, A., Kutten, S., Vaccaro, U., Yung, M.: Perfectly secure key distribution for dynamic conferences. Inf. Comput. 146(1), 1–23 (1998)

[3] Boneh, D., Boyen, X.: Secure identity based encryption without random oracles. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 443–459. Springer, Heidelberg (2004)

[4] Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, p. 213. Springer, Heidelberg (2001)

[5] Boneh, D., Silverberg, A.: Applications of multilinear forms to cryptography. Contemporary Mathematics 324, 71–90 (2002)

[6] Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)

[7] Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. J. Cryptology 17(4), 297–319 (2004)

[8] Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010)

[9] Çapar, Ç., Goeckel, D., Paterson, K.G., Quaglia, E.A., Towsley, D., Zafer, M.: Signal-flow-based analysis of wireless security protocols. Information and Computation 226, 37–56 (2013)

[10] Cui, Y., Fujisaki, E., Hanaoka, G., Imai, H., Zhang, R.: Formal security treatments for signatures from identity-based encryption. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 218–227. Springer, Heidelberg (2007)

[11] Dupont, R., Enge, A.: Provably secure non-interactive key distribution based on pairings. Discrete Applied Mathematics 154(2), 270–276 (2006)
[12] Freire, E.S.V., Hofheinz, D., Kiltz, E., Paterson, K.G.: Non-interactive key exchange. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 254–271. Springer, Heidelberg (2013)
[13] Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 1–17. Springer, Heidelberg (2013), http://eprint.iacr.org/2012/610
[14] Gennaro, R., Halevi, S., Krawczyk, H., Rabin, T., Reidt, S., Wolthusen, S.D.: Strongly-resilient and non-interactive hierarchical key-agreement in MANETs. In: Jajodia, S., Lopez, J. (eds.) ESORICS 2008. LNCS, vol. 5283, pp. 49–65. Springer, Heidelberg (2008)
[15] Gentry, C., Silverberg, A.: Hierarchical ID-based cryptography. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 548–566. Springer, Heidelberg (2002)
[16] Guo, H., Mu, Y., Li, Z., Zhang, X.: An efficient and non-interactive hierarchical key agreement protocol. Computers & Security 30(1), 28–34 (2011)
[17] Hanaoka, G., Matsuda, T., Schuldt, J.C.N.: On the impossibility of constructing efficient key encapsulation and programmable hash functions in prime order groups. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 812–831. Springer, Heidelberg (2012)
[18] Hofheinz, D., Kiltz, E.: Programmable hash functions and their applications. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 21–38. Springer, Heidelberg (2008)
[19] Lysyanskaya, A.: Unique signatures and verifiable random functions from the DH-DDH separation. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 597–612. Springer, Heidelberg (2002)
[20] Paterson, K.G., Srinivasan, S.: On the relations between non-interactive key distribution, identity-based encryption and trapdoor discrete log groups. Des. Codes Cryptography 52(2), 219–241 (2009)
[21] Ramkumar, M., Memon, N., Simha, R.: A hierarchical key pre-distribution scheme. In: 2005 IEEE International Conferenceon on Electro Information Technology (May 2005)
[22] Rückert, M., Schröder, D.: Aggregate and verifiably encrypted signatures from multilinear maps without random oracles. In: Park, J.H., Chen, H.-H., Atiquzzaman, M., Lee, C., Kim, T.-h., Yeo, S.-S. (eds.) ISA 2009. LNCS, vol. 5576, pp. 750–759. Springer, Heidelberg (2009)
[23] Sakai, R., Ohgishi, K., Kasahara, M.: Cryptosystems based on pairing. In: SCIS 2000, Okinawa, Japan (January 2000)
[24] Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)

On the Indifferentiability of Key-Alternating Ciphers

Elena Andreeva[1], Andrey Bogdanov[2], Yevgeniy Dodis[3], Bart Mennink[1], and John P. Steinberger[4]

[1] KU Leuven and iMinds
{elena.andreeva,bart.mennink}@esat.kuleuven.be
[2] Technical University of Denmark
a.bogdanov@mat.dtu.dk
[3] New York University
dodis@cs.nyu.edu
[4] Tsinghua University
jpsteinb@gmail.com

Abstract. The Advanced Encryption Standard (AES) is the most widely used block cipher. The high level structure of AES can be viewed as a (10-round) *key-alternating* cipher, where a t-round key-alternating cipher KA_t consists of a small number t of fixed permutations P_i on n bits, separated by key addition:

$$\mathrm{KA}_t(K, m) = k_t \oplus P_t(\dots k_2 \oplus P_2(k_1 \oplus P_1(k_0 \oplus m))\dots),$$

where $(k_0, \dots, k_t)$ are obtained from the master key K using some key derivation function.

For $t = 1$, KA_1 collapses to the well-known Even-Mansour cipher, which is known to be *indistinguishable* from a (secret) random permutation, if P_1 is modeled as a (public) random permutation. In this work we seek for stronger security of key-alternating ciphers — *indifferentiability from an ideal cipher* — and ask the question under which conditions on the key derivation function and for how many rounds t is the key-alternating cipher KA_t indifferentiable from the ideal cipher, assuming $P_1, \dots, P_t$ are (public) random permutations?

As our main result, we give an affirmative answer for $t = 5$, showing that the 5-*round key-alternating cipher* KA_5 *is indifferentiable from an ideal cipher*, assuming $P_1, \dots, P_5$ are five independent random permutations, and the key derivation function sets all rounds keys $k_i = f(K)$, where $0 \leq i \leq 5$ and f is modeled as a random oracle. Moreover, when $|K| = |m|$, we show we can set $f(K) = P_0(K) \oplus K$, giving an n-bit block cipher with an n-bit key, making only six calls to n-bit permutations $P_0, P_1, P_2, P_3, P_4, P_5$.

Keywords: Even-Mansour, ideal cipher, key-alternating cipher, indifferentiability.

1 Introduction

Block Ciphers. A block cipher $E : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$ takes a κ-bit key K and an n-bit input x and returns an n-bit output y. Moreover, for each key

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 531–550, 2013.

K the map $E(K, \cdot)$ must be a permutation, and come with an efficient inversion procedure $E^{-1}(K, \cdot)$. Block ciphers are central primitives in cryptography. Most importantly, they account for the bulk of data encryption and data authentication occurring in the field today, as well as play a critical role in the design of "cryptographic hash functions" [1–4].

Indistinguishability. The standard security notion for block ciphers is that of (computational) *indistinguishability* from a random permutation, which states that no computationally bounded distinguisher $\mathcal{D}$ can tell apart having oracle access to the block cipher $E(K, \cdot)$ or its inverse $E^{-1}(K, \cdot)$ for a *random key* K from having oracle access to a (single) truly random permutation P and its inverse P^{-1}. This security notion is relatively well understood in the theory community, and is known to be implied by the mere existence of one-way functions, through a relatively non-trivial path: from one-way functions to pseudorandom generators [5], to pseudorandom functions (PRFs) [6], to pseudorandom permutations (PRPs) [7], where the latter term is also a "theory synonym" for the "practical notion" of a block cipher. Among these celebrated results, we explicitly note the seminal work of Luby-Rackoff [7], who proved that four (independently keyed) rounds of the Feistel network $(L', R') = (R, f(K, R) \oplus L)$, also known as the "Luby-Rackoff construction", are enough to obtain a PRP $E((K_1, K_2, K_3, K_4), (L_0, R_0))$ on n-bit inputs/outputs from four $n/2$-to-$n/2$-bits PRFs $f(K_1, R_0), \ldots, f(K_4, R_3)$. In fact, modulo a few exceptions mentioned below, the Luby-Rackoff construction and its close relatives were the *only theoretically-analyzed* ways to build a block cipher.

Is Indistinguishability Enough? Despite this theoretical success, practical ciphers — including the current block cipher standard AES — are built using very different means. One obvious reason is that the theoretical feasibility results above are generally too inefficient to be of practical use (and, as one may argue, were not meant to be). However, a more subtle but equally important reason is that a practitioner — even the one who understands enough theory to know what a PRP is — would not think of a block cipher as a synonym of a PRP, but as something *much stronger*!

For example, the previous U.S. block cipher standard DES had the following so called "key complementary" property $E(\bar{K}, \bar{x}) = \overline{E(K, x)}$, where $\bar{y}$ stands for the bitwise complement of the string y. Although such an equality by itself does not contradict the PRP property, though effectively reducing the key space by a half, it was considered undesirable and typically used as an example of something that a "good" block cipher design should definitely avoid. Indeed, AES is not known to have any simple-to-express relations between its inputs/outputs on related keys. Generally speaking though, related-key attacks under more complex related-key relations (using nonlinear functions on the master key) for AES were identified and received a lot of attention in the cryptanalytic community several years ago [8–10], despite not attacking the standard PRP security. In fact, the recent biclique cryptanalysis of the full AES cipher [11] in the single-key setting implicitly uses the similarity of AES computation under related keys.

Indeed, one of the reasons that practical block ciphers are meant to have stronger-than-PRP properties is that various applications (e.g. [2–4, 12–19]) critically rely on such "advanced properties", which are far and beyond the basic indistinguishability property. Perhaps the most important such example comes in the area of building good "hash functions", as many cryptographic hash functions, including the most extensively used SHA-1/2 and MD5 functions, use the famous block-cipher-based Davies-Meyer compression function $f(K, x) = E(K, x) \oplus x$ in their design.[1] This compression function f is widely believed to be collision-resistant (CR) if E is a "good-enough" block cipher (see more below), but this obviously does not follow from the basic PRP property. For example, modifying any good block cipher E to be the identity permutation on a single key K' clearly does not affect it PRP security much (since, w.h.p., a random key $K \neq K'$), but then $f(K', x) = x \oplus x = 0$ for all x, which is obviously not CR. While the example above seems artificial, we could instead use a natural and quite popular Even-Mansour (EM) [14] cipher $E(K, x) = P(K \oplus x) \oplus K$, where P is some "good-enough" public permutation. As we mention below, the EM cipher is known to be indistinguishable [14] assuming P is a public "random permutation", and, yet, the composed Davies-Meyer hash function $f(K, x) = E(K, x) \oplus x = P(K \oplus x) \oplus (K \oplus x)$ is certainly not CR, as any pair $(K, x) \neq (K', x')$ satisfying $K \oplus x = K' \oplus x'$ yields a collision.

Ideal Cipher Model. Motivated by these (and other) considerations, practitioners view a good block cipher as something much closer to an *ideal cipher* than a mere PRP, much like they view a good hash function much closer to a *random oracle* than a one-way (or collision-resistant) function. In other words, many important applications of block ciphers (sometimes implicitly) assume that E "behaves" like a family $\mathcal{IC}$ of 2^κ completely random and independent permutations $P_1, \ldots, P_{2^\kappa}$. More formally, an analysis in the ideal cipher model assumes that all parties, including the adversary, can make (a bounded number of) both encryption and decryption queries to the ideal block cipher $\mathcal{IC}$, for any given key K (not necessarily random!). Indeed, under such an idealistic assumption one can usually *prove* the security of most of the above mentioned applications of block ciphers [2–4, 13–19], such as a simple and elegant proof that the Davies-Meyer compression function $f(K, x) = E(K, x) \oplus x$ is CR in the ideal cipher model (ICM) [19].

Of course, the ideal cipher model is ultimately a heuristic, and one can construct artificial schemes that are secure in the ICM, but insecure for any concrete block cipher [22]. Still, a proof in the ideal cipher model seems useful because it shows that a scheme is secure against generic attacks, that do not exploit specific weaknesses of the underlying block cipher. Even more important than potential applications, the ICM gives the block cipher designers a much "higher-than-PRP" *goal* that they should strive to achieve in their proposed designs, even though this goal is, theoretically-speaking, impossible to achieve. This raises

[1] Where E is some particular block cipher; e.g., in the case if SHA-1/2, it was called SHACAL [20, 21].

an important question to the theory community if it is possible to offer some theoretical framework within which one might be able to evaluate the design of important block ciphers, such as AES, in terms of being "close" to an ideal cipher or, at least, resisting generic "structure-abusing" attacks.

Indifferentiability. One such framework is the so-called *indifferentiability* framework of Maurer et al. [23], popularized by Coron et al. [24] as a clean and elegant way to formally assess security of various idealized constructions of hash functions and block ciphers. Informally, given a construction of one (possibly) idealized primitive B (i.e., block cipher) from *another idealized* primitive A (i.e., random oracle), the indifferentiability framework allows one to formally argue the security of B in terms of (usually simpler) A. Thus, although one does not go all the way to building B from scratch, the indifferentiability proof illustrates the lack of "generic attacks" on B, and shows that any concrete attack must use something about the internals of any candidate implementation of A. Moreover, the indifferentiability framework comes with a powerful composition theorem [23] which means that most natural (see [25]) results shown secure in the "ideal-B" model can safely use the construction of B using A instead, and become secure in the "ideal-A" model.

For example, we already mentioned that the design of popular hash functions, such as SHA-1/2 and MD5, could be generically stated in terms of some underlying block cipher E. Using the indifferentiability framework, one can *formally* ask if the resulting hash function is indifferentiable from a random oracle if E is an ideal cipher. Interestingly, Coron et al. [24] showed a negative answer to this question. Moreover, this was not a quirk of the model, but came from a well-known (and serious) "extension" attack on the famous Merkle-Damgård domain extension [26, 27]. Indeed, an attack on indifferentiability usually leads to a serious real-world attack for some applications, and, conversely, the security proof usually tells that the high-level design of a given primitive (in this case hash function) does not have structural weaknesses. Not surprisingly, all candidates for the recently concluded SHA-3 competition were strongly encouraged to come with a supporting indifferentiability proof in some model (as we will expand on shortly).

Random Oracle vs. Ideal Cipher. Fortunately, Coron et al. [24] also showed that several simple tweaks (e.g., truncating the output or doing prefix-free input encoding) make the resulting hash function construction indifferentiable from a random oracle. Aside from formally showing that the ICM model "implies" the random oracle model (ROM) in theory, these (and follow-up [28, 29]) positive results showed that (close relatives of) *practically used* constructions are "secure" (in the sense of resisting *generic* attacks, as explained above).

From the perspective of this work, where we are trying to validate the design principle behind existing block ciphers, the opposite direction (of building an ideal cipher from a random oracle) is much more relevant. Quite interestingly, it happened to be significantly more challenging than building a PRP out of a PRF. Indeed, the most natural attempt is to use the already mentioned Feistel construction, that uses the given random oracles f to implement the required round

functions.[2] However, unlike the standard PRF-PRP case, where four rounds were already sufficient [7], in the indifferentiability setting even five rounds are provably insecure [24, 30, 31]. On a high-level, the key issue is that in the latter framework the distinguisher can have direct access to all the intermediate round functions, which was provably impossible in the more restricted indistinguishability framework. As a step towards overcoming this difficulty, Dodis and Puniya [31] considered a variant of the indifferentiability framework called "honest-but-curious" (HBC) indifferentiability, where the adversary can only query the global Feistel construction, and get all the intermediate results, but cannot directly query the round functions. In this model, which turns out to be *incomparable* to "standard" indifferentiability [30], they showed that the Feistel construction with a super-logarithmic number of rounds (with random oracle round functions) is HBC-indifferentiable from a fixed ideal permutation. The elegant work of Coron et al. [30] (and later Seurin [32]) conjectured and attempted a "standard" indifferentiability proof for the Feistel construction with six rounds. Unfortunately, while developing several important techniques, the proof contained some non-trivial flaws. Fortunately, this result was later fixed by Holenstein et al. [33], who succeeded in proving that a fourteen-round Feistel construction can be used to build an ideal cipher from a random oracle.

Key-Alternating Ciphers. Despite this great theoretical success showing the equivalence between the random oracle and the ideal cipher models, the above results of [30, 32, 33] only partially address our main motivation of theoretically studying the soundness of the design of *existing* block ciphers. In particular, we notice that (from a high level) there are two major design principles for block ciphers. The "old school" approach is indeed Feistel-based, with many prominent ciphers such as DES, Blowfish, Camellia, FEAL, Lucifer, and MARS. However, it appears that all such ciphers use *rather weak* (albeit non-trivial) round functions, and (in large part) get their security by using *many more* rounds than theoretically predicted. So, while the theoretical soundness of the Feistel network is important philosophically, it is unclear that random oracle modeling of the round functions is realistic.

In fact, we already mentioned a somewhat paradoxical fact: while, in theory, the random oracle model appears much more basic and minimal than the highly structured ideal cipher model (much like a one-way function is more basic than a one-way permutation), in practice, the implication appears to be *totally reversed*. In particular, in practice it appears much more accurate to say that hash functions (or "random oracles") are built from block ciphers (or "ideal ciphers") than the other way around. Indeed, in addition to the widely used SHA-1/2 and

[2] The most natural modeling would give a *single* n-to-n-bit permutation from several $n/2$-to-$n/2$-bit random oracles. However, by prepending the *same* κ-bit key K to each such RO, one gets a candidate block cipher. We notice, though, that unlike the secret-key setting, it is (clearly) *not* secure to prepend several *independent* keys to each round function. We will come back to this important point when discussing the importance of key derivation in the indifferentiability proofs.

MD5 examples, other prominent block-cipher-based hash functions are recent SHA-3 finalists BLAKE [34] and Skein [35].

Perhaps most importantly for us, the current block cipher standard AES, as well as a few other "new school" ciphers (e.g., 3-Way, SHARK, Serpent, Present, and Square), are *not Feistel-based*. Instead, such ciphers are called *key-alternating ciphers*, and their design goes back to Daemen [36–38]. In general, a key-alternating cipher KA_t consists of a small number t of fixed permutations P_i on n bits, separated by key addition:

$$\mathrm{KA}_t(K, m) = k_t \oplus P_t(\dots k_2 \oplus P_2(k_1 \oplus P_1(k_0 \oplus m)) \dots),$$

where the round keys $k_0, \dots, k_t$ are derived from the master key K using some *key derivation* (aka "key schedule") function. For one round $t = 1$, the construction collapses to the well-known Even-Mansour (EM) [14] cipher. Interestingly, already in the standard "PRP indistinguishability" model, the analysis of the EM [14] (and more general key-alternating ciphers [39]) seems to require the modeling of P as a *random permutation* (but, on the other hand, does not require another computational assumption such as a PRF). With this idealized modeling, one can show that the Even-Mansour cipher is indistinguishable [14], and, in fact, its exact indistinguishablity security increases beyond the "birthday bound" as the number of round increases to 2 and above [39, 40].

Our Main Question. Motivated by the above discussion, we ask the main question of our work:

> *Under which conditions on the key derivation function and for how many rounds t is the key-alternating cipher KA_t indifferentiable from the ideal cipher, assuming $P_1, \dots, P_t$ are random permutations?*

As we mentioned, one motivation for this question comes from the actual design of the AES cipher, whose design principles we are trying to analyze. The second motivation comes from the importance of having the composition theorem guaranteed by the indifferentiability framework. Indeed, we already saw a natural example where using the Even-Mansour cipher to instantiate the classical Davies-Meyer compression function gave a totally insecure construction, despite the fact that the Davies-Meyer construction was known to be collision-resistant in the ideal cipher model [19], and the EM cipher indistinguishable in the random permutation model [14]. The reason for that is the fact that the EM cipher is easily seen to be not indifferentiable from an ideal cipher. In contrast, if we were to use a variant of the key alternating cipher which *is* provably indifferentiable, we would be *guaranteed* that the composed Davies-Meyer function remains collision-resistant (now, in the random permutation model).

The third motivation comes from the fact that the direct relationship between the random permutation (RP) model and the ideal cipher model is interesting in its own right. Although we know that these primitives are equivalent through the chain "IC $\Rightarrow$ RP (trivial) $\Rightarrow$ RO [41, 42] $\Rightarrow$ IC [30, 33]", a direct "RP $\Rightarrow$ IC" implication seems worthy of study in its own right (and was mentioned as

an open problem in [43]).[3] More generally, we believe that the random permutation model (RPM) actually deserves its own place alongside the ROM and the ICM. The reason is that both the block cipher standard AES and the new SHA-3 standard Keccak [44] (as well as several other prominent SHA-3 finalists Grøstl [45] and JH [46]) are most cleanly described using a (constant number of) *permutation(s)*. The practical reason appears to be that it seems easier to ensure that the permutation design does not lose any entropy (unlike an ad-hoc hash function), or would not have some non-trivial relationship among different keys (unlike an ad-hoc block cipher). Thus, we find the indifferentiability analyses in the RPM very relevant both in theory and in practice. Not surprisingly, there has been an increased number of works as of late analyzing various constructions in the RPM [39, 41, 42, 47–50].

Our Main Result. As our main result, we show the following theorem.

Theorem 1. *The 5-round key-alternative cipher KA_5 is indifferentiable from an ideal cipher, assuming $P_1, \ldots, P_5$ are five independent random permutations, and the key derivation function sets all rounds keys $k_i = f(K)$, where $0 \le i \le 5$ and f is modeled as a κ-to-n-bits random oracle.*

A more detailed statement appears in Theorem 3. In particular, our indifferentiability simulator has provable security $O(q^{10}/2^n)$, running time $O(q^3)$, and query complexity $O(q^2)$ to answer q queries made by the distinguisher. Although (most likely) far from optimal, our bounds are (unsurprisingly) much better than the $O(q^{16}/2^{n/2})$ and $O(q^8)$ provable bounds achieved by following the indirect "random-oracle route" [33].

We also show a simple attack illustrating that a one- or even two-round KA_t construction is never indifferentiable from the ideal cipher (in the full version of this paper [51]). This should be contrasted with the simpler indistinguishability setting, where the 1-round Even-Mansour construction is already secure [14]. Indeed, as was the case with Merkle-Damgård based hash function design and the "extension attack", the Davies-Meyer composition fiasco of the 1-round EM cipher demonstrated that this lack of indifferentiability indeed leads to a serious real-world attack on this cipher.

Finally, we give some justification of why we used 5 rounds, by attacking several "natural" simulators for the 4-round construction.

Importance of Key Derivation. Recall, in the secret-key indistinguishability case, the key derivation function was only there for the sake of minimizing the key length, and having $t+1$ independent keys $k_0, \ldots, k_t$ resulted in the best security analysis. Here, the key K is *public* and controlled by the attacker. In particular, it is trivial to see that having $t+1$ independent keys is like having a one-round construction (as then the attacker can simply fix all-but-one-keys k_i), which we know is trivially insecure. Thus, in the indifferentiability setting it is very important that the keys are somehow correlated (e.g., equal).

[3] Indeed, our efficiency and security below are much better than following the indirect route through random oracle.

Another important property for the key derivation functions, at least if one wants to optimize the number of rounds, appears to be its *invertibility*. Very informally, this means that the only way to compute a valid round k_i is to "honestly compute" a key derivation function f on some key K first. In particular, in our analysis we use a random oracle as such a non-invertible key derivation function. We give some evidence of the importance of invertibility for understanding the indifferentiability-security of key-alternating ciphers by (1) critically using such non-invertibility in our analysis; and (2) showing several somewhat surprising attacks for the 3-round construction with certain natural "invertible" key schedules (e.g., all keys k_i equal to K for $\kappa = n$). We stress that our results do not preclude the use of invertible key schedules for a sufficiently large number of rounds (say, 10-12), but only indicate why having non-invertible key schedules is very helpful in specific analyses (such as ours) and also for avoiding specific attacks (such as our 3-round attacks). Indeed, subsequent to our work, Lampe and Seurin [52] showed that the 12-round key alternating cipher will all keys $k_i = K$ (for $\kappa = n$) is indeed indifferentiable from an ideal cipher, with security $O(q^{12}/2^n)$ and simulator query complexity $O(q^4)$ to answer q queries made by the distinguisher. Although using substantially more rounds and achieving noticeably looser exact security than this work, their result is closer to the actual design of the AES cipher, whose key schedule f is indeed easily invertible.

Instantiating the Key Derivation Function. Although we use a random oracle as a key derivation function (see above), in principle one can easily (and efficiently!) build the required random oracle from a random permutation [41, 42], making the whole construction entirely permutation-based. For example, the most optimized "enhanced-CBC" construction from [41] will use only a single additional random permutation and make $\frac{2\kappa}{n} + O(1)$ calls to this permutation to build a κ-to-n-bit random oracle f.[4] Unsurprisingly, this instantiation will result in a cipher making a lot fewer calls to the random permutation (by a large constant factor) than following the indirect RP-to-RO-to-IC cycle.

Moreover, we can further optimize the most common case $\kappa = n$ as follows. First, [41] showed that $f(K) = P(K) \oplus P^{-1}(K)$ is $O(q^2/2^n)$-indifferentiable from an n-to-n-bit random oracle, which already results in a very efficient block cipher construction with 7 permutation calls. Second, by closely examining our proof, we observe that we do not need the full power of the random oracle f for key derivation. Instead, our proof only uses the "preimage awareness" [53] of the random oracle[5] and the fact that random oracle avoids certain simple combinatorial relations among different derived keys. In particular, we observe that the "unkeyed Davies-Meyer" function [41] $f(K) = P(K) \oplus K$ is enough for our analysis to go through. This gives the following result for building an n-bit ideal cipher with n-bit key, using only six random permutation calls.

[4] The indifferentiability security of this construction to handle q queries is "only" $O(q^4/2^n)$, but this is still much smaller than the bound in Theorem 3, and will not affect the final asymptotic security.

[5] Informally, at any point of time the simulator knows the list of all input-output pairs to f "known" by the distinguisher.

Theorem 2. *The following n-bit cipher with n-bit key is indifferentiable from an ideal cipher:*

$$E(K, m) = k \oplus P_5(k \oplus P_4(k \oplus P_3(k \oplus P_2(k \oplus P_1(k \oplus m))))),$$

where $k = P_0(K) \oplus K$ and $P_0, P_1, P_2, P_3, P_4, P_5$ are random permutations.

Overall, our results give the first theoretical evidence for the design soundness of key-alternative ciphers — including AES, 3-Way, SHARK, Serpent, Present, and Square — from the perspective of indifferentiability.[6]

2 Preliminaries

For a domain $\{0,1\}^m$ and a range $\{0,1\}^n$, a random oracle $\mathcal{R} : \{0,1\}^m \to \{0,1\}^n$ is a function drawn uniformly at random from the set of all possible functions that map m to n bits. For two sets $\{0,1\}^\kappa$ and $\{0,1\}^n$, an ideal cipher $\mathcal{IC} : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$ is taken randomly from the set of all block ciphers with key space $\{0,1\}^\kappa$ and message and ciphertext space $\{0,1\}^n$. A random permutation $\pi : \{0,1\}^n \to \{0,1\}^n$ is a function drawn randomly from the set of all n-bit permutations.

Key-Alternating Ciphers. A key-alternating cipher KA_t consists of a small number t of fixed permutations P_i on n bits separated by key addition:

$$\mathrm{KA}_t(K, m) = k_t \oplus P_t(\ldots k_2 \oplus P_2(k_1 \oplus P_1(k_0 \oplus m)) \ldots),$$

where the round keys $k_0, \ldots, k_t$ are derived from the master key K using some key schedule f: $(k_0, \ldots, k_t) = f(K)$. The notion of key-alternating ciphers itself goes back to Daemen [36–38] and was used in the design of AES. However, it was Knudsen [55] who proposed to instantiate multiple-round key-alternating ciphers with randomly drawn, fixed and public permutations (previously, a single-round key-alternating construction was proposed by Even-Mansour [14]).

Indifferentiability. We use the notion of indifferentiability [23, 24] in our proofs to show that if a construction $\mathcal{C}^\mathcal{P}$ based on an ideal subcomponent $\mathcal{P}$ is indifferentiable from an ideal primitive $\mathcal{R}$, then $\mathcal{C}^\mathcal{P}$ can replace $\mathcal{R}$ in any system. As noticed in [25] the latter statement must be qualified with some fine print: since the adversary must eventually incorporate the simulator, the indifferentiability composition theorem only applies in settings where the adversary comes from a computational class that is able to "swallow" the simulator (e.g., the class of polynomial-time, polynomial-space algorithms); see [25, 56] for more details on the limitations of indifferentiability.

Definition 1. *A Turing machine $\mathcal{C}$ with oracle access to an ideal primitive $\mathcal{P}$ is called $(t_D, t_S, q, \varepsilon)$-indifferentiable from an ideal primitive $\mathcal{R}$ if there exists a*

[6] We also mention a complementary recent work of [54], who mainly looked at "weaker-than-indistinguishability" properties which can be proven about AES design.

simulator $\mathcal{S}$ with oracle access to $\mathcal{R}$ and running in time t_S, such that for any distinguisher D running in time at most t_D and making at most q queries, it holds that:

$$\mathrm{Adv}^{\mathrm{indif}}_{\mathcal{C},\mathcal{R},\mathcal{S}}(D) = \left|\Pr\left[D^{\mathcal{C}^{\mathcal{P}},\mathcal{P}} = 1\right] - \Pr\left[D^{\mathcal{R},\mathcal{S}^{\mathcal{R}}} = 1\right]\right| < \varepsilon.$$

Distinguisher D can query both its *left oracle* (either $\mathcal{C}$ or $\mathcal{R}$) and its *right oracle* (either $\mathcal{P}$ or $\mathcal{S}$). We refer to $\mathcal{C}^{\mathcal{P}}, \mathcal{P}$ as the *real world*, and to $\mathcal{R}, \mathcal{S}^{\mathcal{R}}$ as the *simulated world*.

3 Indifferentiability of KA_5

In this section we discuss our main result, namely that KA_5 with an RO key schedule is indifferentiable from an ideal cipher. In the statement below, KA_5 stands for a 5-round key-alternating cipher implemented with round functions $P_1, \ldots, P_5$ and key scheduling function f, with the round functions, their inverses, and the key scheduling function all being available for oracle queries by the adversary (and thus, also, all being implemented as interfaces by the simulator).

Theorem 3. *Let $P_1, \ldots, P_5$ be independent random n-bit permutations, and f be a random κ-to-n-bits function. Let D be an arbitrary information-theoretic distinguisher that makes at most q queries. Then there exists a simulator $\mathcal{S}$ such that*

$$\mathrm{Adv}^{\mathrm{indif}}_{KA_5,\mathcal{IC},\mathcal{S}}(D) \leq 320 \cdot 6^{10}\left(\frac{q^{10}}{2^n} + \frac{q^4}{2^n}\right) = O\left(\frac{q^{10}}{2^n}\right),$$

where $\mathcal{S}$ makes at most $2q^2$ queries to the ideal cipher $\mathcal{IC}$ and runs in time $O(q^3)$.

Our 5-round simulator $\mathcal{S}$ is given by the pseudocode in game G_1 (see Figures 1–4), and more precisely by the public functions f, P1, $\mathrm{P1}^{-1}$, P2, $\mathrm{P2}^{-1}$, ..., P5, $\mathrm{P5}^{-1}$ within G_1. Here f emulates the key scheduling random oracle, whereas P1, $\mathrm{P1}^{-1}$ emulate the random permutation P_1 and its inverse P_1^{-1}, and so on. Since the pseudocode of game G_1 is not easy to assimilate, a high-level description of our simulator is likely welcome. Furthermore, because the simulator is rather complex, we also try to argue the necessity of its complex behavior by discussing why some simpler classes of simulators might not work.

To describe the simulator-distinguisher interaction we use expressions such as "D makes the query $\mathrm{f}(K) \to k$" to mean that the distinguisher D queries f (which is implemented by the simulator) on input K, and receives answer k as a result. The set of values k for which the adversary has made a query of the form $\mathrm{f}(K) \to k$ for some $K \in \{0,1\}^\kappa$ is denoted $\mathcal{Z}$ (thus $\mathcal{Z}$ is a time-dependent set). If $\mathrm{f}(K) \to k$ then we also write K as "$f^{-1}(k)$"; here f and f^{-1} are internal tables maintained by the simulator to keep track of scheduled keys and their preimages (see procedure $\mathrm{f}(K)$ in Figure 1 for more details).

Game G_1

Random tapes: $p_1, \ldots, p_5, \{p_E[K] : K \in \{0,1\}^\kappa\}, r_f$

private procedure ReadTape($Table, x, p$)
  $y \leftarrow p(x)$
  if ($Table(x) \neq \perp$) **then abort**
  if ($Table^{-1}(y) \neq \perp$) **then abort**
  $Table(x) \leftarrow y$
  $Table^{-1}(y) \leftarrow x$
  return y

public procedure E(K, x)
  if ($ETable[K](x) \neq \perp$) **return** $ETable[K](x)$
  $y \leftarrow$ ReadTape($ETable[K], x, p_E[K](\rightarrow, \cdot)$)
  return y

public procedure $\mathrm{E}^{-1}(K, y)$
  if ($ETable[K]^{-1}(y) \neq \perp$) **return** $ETable[K]^{-1}(y)$
  $x \leftarrow$ ReadTape($ETable[K]^{-1}, y, p_E[K](\leftarrow, \cdot)$)
  return x

public procedure f(K)
  if $f(K) \neq \perp$ **return** $f(K)$
  $k \leftarrow r_f(K)$
  $\mathcal{Z} \leftarrow \mathcal{Z} \cup k$
  $f(K) \leftarrow k$
  $f^{-1}(k) \leftarrow K$
  $KeyQueries \leftarrow KeyQueries \cup \{(K, k, {+}{+}qnum)\}$
  if ($qnum > 6q^2 + q$) **then abort**
  return $f(K)$

public procedure P1(x)
  if ($P_1(x) \neq \perp$) **return** $P_1(x)$
  $y \leftarrow$ ReadTape($P_1, x, p_1(\rightarrow, \cdot)$)
  AddQuery($1, x, y, \rightarrow$)
  return $P_1(x)$

public procedure $\mathrm{P1}^{-1}(y)$
  $\mathrm{PrivateP1}^{-1}(y)$
  Cleanup()
  return $P_1^{-1}(y)$

Game G_1 (continued)

private procedure $\mathrm{PrivateP1}^{-1}(y)$
  if ($P_1^{-1}(y) \neq \perp$) **return** $P_1^{-1}(y)$
  $x \leftarrow$ ReadTape($P_1^{-1}, y, p_1(\leftarrow, \cdot)$)
  AddQuery($1, x, y, \leftarrow$)
  FreezeLeftValues($x, \perp$)
  $LeftQueue \leftarrow LeftQueue \cup (1^+, y)$
  return $P_1^{-1}(y)$

public procedure P2(x)
  if ($P_2(x) \neq \perp$) **return** $P_2(x)$
  $y \leftarrow$ ReadTape($P_2, x, p_2(\rightarrow, \cdot)$)
  AddQuery($2, x, y, \rightarrow$)
  return $P_2(x)$

public procedure $\mathrm{P2}^{-1}(y)$
  if ($P_2^{-1}(y) \neq \perp$) **return** $P_2^{-1}(y)$
  $x \leftarrow$ ReadTape($P_2^{-1}, y, p_2(\leftarrow, \cdot)$)
  AddQuery($2, x, y, \leftarrow$)
  return $P_2^{-1}(y)$

public procedure P3(x)
  PrivateP3(x)
  Cleanup()
  return $P_3(x)$

private procedure PrivateP3(x)
  if ($P_3(x) \neq \perp$) **return** $P_3(x)$
  $y \leftarrow$ ForcedP3($3^-, x$)
  if ($y \neq \perp$) **then**
    if ($y \in$ range(P_3)) **then abort**
    $P_3(x) \leftarrow y$
    $P_3^{-1}(y) \leftarrow x$
    AddQuery($3, x, y, \perp$)
    $RightQueue \leftarrow RightQueue \cup (3^+, y)$
  else
    $y \leftarrow$ ReadTape($P_3, x, p_3(\rightarrow, \cdot)$)
    AddQuery($3, x, y, \rightarrow$)
  end if
  $LeftQueue \leftarrow LeftQueue \cup (3^-, x)$
  return $P_3(x)$

Fig. 1. The simulated world (first of four sets of procedures)

A triple (i, x, y) such that D has made the query $\mathrm{P}i(x) \rightarrow y$ or $\mathrm{P}i^{-1}(y) \rightarrow x$ is called an *i-query*, $i \in \{1, 2, 3, 4, 5\}$. Moreover, when the simulator "internally defines" a query $\mathrm{P}i(x) = y$, $\mathrm{P}i^{-1}(y) = x$ we also call the associated triple (i, x, y) an i-query, even though the adversary might not be aware of these values yet. (While this might seem a little informal, we emphasize that this section is, indeed, meant mainly as an informal overview.) A pair of queries (i, x_i, y_i), $(i+1, x_{i+1}, y_{i+1})$ such that $y_i \oplus k = x_{i+1}$ for some $k \in \mathcal{Z}$ is called *k-adjacent*. We also say that a pair of queries $(1, x_1, y_1)$, $(5, x_5, y_5)$ is *k-adjacent* if $k \in \mathcal{Z}$ and $\mathrm{E}(f^{-1}(k), x_1 \oplus k) = y_5 \oplus k$, where $\mathrm{E}(K, x)$ is the ideal cipher (and $\mathrm{E}^{-1}(K, y)$ its inverse). (Since $\mathcal{Z}$ is time-dependent, a previously non-adjacent pair of queries might become adjacent later on; of course, this is unlikely.) A sequence of queries

$$(1, x_1, y_1), (2, x_1, y_2), \ldots, (5, x_5, y_5)$$

Game G_1 (continued)

public procedure $\mathrm{P3}^{-1}(y)$
  $\mathrm{PrivateP3}^{-1}(y)$
  Cleanup()
  return $P_3^{-1}(y)$

private procedure $\mathrm{PrivateP3}^{-1}(y)$
  if $(P_3^{-1}(y) \neq \perp)$ **return** $P_3^{-1}(y)$
  $x \leftarrow \mathrm{ForcedP3}(3^+, y)$
  if $(x \neq \perp)$ **then**
    if $(x \in \mathrm{domain}(P_3))$ **then abort**
    $P_3(x) \leftarrow y$
    $P_3^{-1}(y) \leftarrow x$
    $\mathrm{AddQuery}(3, x, y, \perp)$
    $\mathit{LeftQueue} \leftarrow \mathit{LeftQueue} \cup (3^-, x)$
  else
    $\mathrm{ReadTape}(P_3^{-1}, y, p_3(\leftarrow, \cdot))$
    $\mathrm{AddQuery}(3, x, y, \leftarrow)$
  end if
  $\mathit{RightQueue} \leftarrow \mathit{RightQueue} \cup (3^+, y)$
  return $P_3^{-1}(y)$

public procedure $\mathrm{P4}(x)$
  if $(P_4(x) \neq \perp)$ **return** $P_4(x)$
  $y \leftarrow \mathrm{ReadTape}(P_4, x, p_4(\rightarrow, \cdot))$
  $\mathrm{AddQuery}(4, x, y, \rightarrow)$
  return $P_4(x)$

public procedure $\mathrm{P4}^{-1}(y)$
  if $(P_4^{-1}(y) \neq \perp)$ **return** $P_4^{-1}(y)$
  $x \leftarrow \mathrm{ReadTape}(P_4^{-1}, y, p_4(\leftarrow, \cdot))$
  $\mathrm{AddQuery}(4, x, y, \leftarrow)$
  return $P_4^{-1}(y)$

public procedure $\mathrm{P5}(x)$
  $\mathrm{PrivateP5}(x)$
  Cleanup()
  return $P_5(x)$

private procedure $\mathrm{PrivateP5}(x)$
  if $(P_5(x) \neq \perp)$ **return** $P_5(x)$
  $y \leftarrow \mathrm{ReadTape}(P_5, x, p_5(\rightarrow, \cdot))$
  $\mathrm{AddQuery}(5, x, y, \rightarrow)$
  $\mathrm{FreezeRightValues}(y, \perp)$
  $\mathit{LeftQueue} \leftarrow \mathit{LeftQueue} \cup (5^-, x)$
  return $P_5(x)$

Game G_1 (continued)

public procedure $\mathrm{P5}^{-1}(y)$
  if $(P_5^{-1}(y) \neq \perp)$ **return** $P_5^{-1}(y)$
  $x \leftarrow \mathrm{ReadTape}(P_5^{-1}, y, p_5(\leftarrow, \cdot))$
  $\mathrm{AddQuery}(5, x, y, \leftarrow)$
  return $P_5^{-1}(y)$

private procedure $\mathrm{FreezeLeftValues}(x_1, k^\star)$
  forall $k \in \mathcal{Z} \backslash \{k^\star\}$ **do**
    if $(x_1 \oplus k \in \mathit{LeftFreezer})$ **then abort**
    $\mathit{LeftFreezer} \leftarrow \mathit{LeftFreezer} \cup \{x_1 \oplus k\}$
  end forall

private procedure $\mathrm{FreezeRightValues}(y_5, k^\star)$
  ... // (symmetric to FreezeLeftValues)

private procedure $\mathrm{ForcedP3}(i, z)$
  if $(i = 3^-)$ **then**
    $x_3 \leftarrow z$
    $\mathit{candidate} \leftarrow \emptyset$
    forall $k \in \mathcal{Z}$ **do**
      if $(x_3 \oplus k \notin \mathrm{range}(P_2))$ **continue**
      $y_1 \leftarrow P_2^{-1}(x_3 \oplus k) \oplus k$
      if $(y_1 \notin \mathrm{range}(P_1))$ **continue**
      $x_1 \leftarrow P_1^{-1}(y_1)$
      if $(x_1 \oplus k \in \mathit{LeftFreezer})$ **continue**
      if $(\mathit{candidate} \neq \emptyset)$ **then abort**
      $\mathit{candidate} \leftarrow (k, x_1 \oplus k)$
    end forall // (k)
    if $(\mathit{candidate} = \emptyset)$ **return** $\perp$
    $(k, x) \leftarrow \mathit{candidate}$
    $y_5 \leftarrow \mathrm{E}(f^{-1}(k), x) \oplus k$
    $\mathrm{TallyEQuery}(f^{-1}(k), x, \rightarrow)$
    if $(y_5 \notin \mathrm{range}(P_5))$ **return** $\perp$
    $y_4 \leftarrow P_5^{-1}(y_5) \oplus k$
    return $\mathrm{P4}^{-1}(y_4) \oplus k$
  end if
  if $(i = 3^+)$ **then**
    ... // (symmetric to case $(i = 3^-)$)
  end if
  return $\perp$

Fig. 2. The simulated world (second of four sets of procedures)

for which there exists a $k \in \mathcal{Z}$ such that each adjacent pair is k-adjacent and such that the first and last queries are also k-adjacent is called a *completed k-path* or *completed k-chain*.

Consider first the simplest attack that a distinguisher D might carry out: D chooses a random $x \in \{0,1\}^n$ and a random $K \in \{0,1\}^\kappa$ (where $\{0,1\}^\kappa$ is the key space), queries $\mathrm{E}(K, x) \rightarrow y$ (to its left oracle), then queries $\mathrm{f}(K) \rightarrow k$, $\mathrm{P1}(x \oplus k) \rightarrow y_1$, $\mathrm{P2}(y_1 \oplus k) \rightarrow y_2$, $\mathrm{P3}(y_2 \oplus k) \rightarrow y_3$, ..., $\mathrm{P5}(y_4 \oplus k) \rightarrow y_5$ to the simulator, and finally checks that $y_5 \oplus k = y$. The simulator, having itself answered the query $\mathrm{f}(K)$, can already anticipate the distinguisher's attack when the query $\mathrm{P2}(y_1 \oplus k)$ is made, since it sees that a k-adjacency is about to be formed between a 1-query and a 2-query. At this point, a standard strategy

```
Game G1 (continued)

private procedure ExistsPath(i, z, k)
  if (i = 1+) then
    y1 ← z
    if (y1 ∉ range(P1)) return false
    x1 ← P1^{-1}(y1)
    (ℓ, x) ← ProbeForward(2, 5, y1 ⊕ k, k)
    if (ℓ ≠ 5 ∨ x ∉ domain(P5)) return false
    if (E(f^{-1}(k), x1 ⊕ k) ≠ P5(x) ⊕ k) then abort
    TallyEQuery(f^{-1}(k), x1 ⊕ k, →)
    return true
  end if
  if (i = 3−) then
    x3 ← z
    (ℓ1, y) ← ProbeBackward(2, 1, x3 ⊕ k, k)
    (ℓ2, x) ← ProbeForward(3, 5, x3, k)
    if (ℓ1 ≠ 1 ∨ y ∉ range(P1)) return false
    if (ℓ2 ≠ 5 ∨ x ∉ domain(P5)) return false
    if (E(f^{-1}(k), P1^{-1}(y)⊕k) ≠ P5(x)⊕k) then abort
    TallyEQuery(f^{-1}(k), P1^{-1}(y) ⊕ k, →)
    return true
  end if
  if (i = 3+) then
    ... // (symmetric to case (i = 3−))
  end if
  if (i = 5−) then
    ... // (symmetric to case (i = 1+))
  end if

private procedure ProbeForward(i, j, xi, k)
  // (i, j ∈ {1, 2, 3, 4, 5}, i < j)
  while i < j do
    if (Pi(xi) = ⊥) break
    xi ← Pi(xi) ⊕ k
    i ← i + 1
  end
  return (i, xi)

private procedure ProbeBackward(i, j, yi, k)
  // (i, j ∈ {1, 2, 3, 4, 5}, i > j)
  while i > j do
    if (Pi^{-1}(yi) = ⊥) break
    yi ← Pi^{-1}(yi) ⊕ k
    i ← i − 1
  end
  return (i, yi)
```

```
Game G1 (continued)

private procedure EmptyQueue()
  do
    while ¬LeftQueue.empty()
      (i, z) ← LeftQueue.pop()
      if (i = 1+) then ProcessNew1Edge(z)
      if (i = 3−) then ProcessNew3−Edge(z)
    end while
    while ¬RightQueue.empty()
      (i, z) ← RightQueue.pop()
      if (i = 3+) then ProcessNew3+Edge(z)
      if (i = 5−) then ProcessNew5Edge(z)
    end while
  while (¬LeftQueue.empty())

private procedure ProcessNew1Edge(y1)
  forall k ∈ Z
    if (ExistsPath(1+, y1, k)) then continue
    if (y1 ⊕ k ∉ domain(P2)) then continue
    CompletePath1+(y1, k)
  end forall

private procedure ProcessNew3−Edge(x3)
  forall k ∈ Z
    if (ExistsPath(3−, x3, k)) then continue
    if (x3 ⊕ k ∉ range(P2)) then continue
    CompletePath3−(x3, k)
  end forall

private procedure ProcessNew3+Edge(y3)
  ... // (symmetric to ProcessNew3−Edge)

private procedure ProcessNew5Edge(x5)
  ... // (symmetric to ProcessNew1Edge)

private procedure Cleanup()
  EmptyQueue()
  LeftFreezer ← ∅
  RightFreezer ← ∅

private procedure AddQuery(i, x, y, dir)
  Queries ← Queries ∪ {(i, x, y, dir, ++qnum)}
  if (qnum > 6q^2 + q) then abort
```

Fig. 3. The simulated world (third of four sets of procedures)

would be for the simulator to pre-emptively[7] complete a k-chain by answering (say) the queries $\mathrm{P3}(y_2 \oplus k)$ and $\mathrm{P4}(y_3 \oplus k)$ randomly itself, and setting the value of $\mathrm{P5}(y_4 \oplus k)$ to $\mathrm{E}(f^{-1}(k), x) \oplus k$ by querying E.

The distinguisher might vary this attack by building a chain "from the right" (by choosing a random y and querying $\mathrm{P5}^{-1}(y \oplus k) \to x_5$, $\mathrm{P4}^{-1}(x_5 \oplus k) \to x_4$, etc) or by building a chain "from the inside" (e.g., by choosing a random x_3 and querying $\mathrm{P3}(x_3) \to y_3$, $\mathrm{P2}^{-1}(x_3 \oplus k)$, $\mathrm{P4}(y_3 \oplus k) \to y_4$, ...) or even by building a chain "from the left and right" simultaneously (the two sides meeting

[7] Pre-emption is generally desirable in order for the simulator to avoid becoming "trapped" in an over-constrained situation.

Game G_1 (continued)

private procedure CompletePath1$^+(y_1, k)$
  $x_1 \leftarrow P_1^{-1}(y_1)$
  $x_3 \leftarrow P_2(y_1 \oplus k) \oplus k$
  $x_4 \leftarrow$ PrivateP3$(x_3) \oplus k$
  $x_5 \leftarrow$ P4$(x_4) \oplus k$
  FinishPath1$^+$3$^-(x_1, x_5, k)$

private procedure CompletePath3$^-(x_3, k)$
  $x_2 \leftarrow P_2^{-1}(x_3 \oplus k)$
  $x_1 \leftarrow$ PrivateP1$^{-1}(x_2 \oplus k)$
  $x_4 \leftarrow P_3(x_3) \oplus k$
  $x_5 \leftarrow$ P4$(x_4) \oplus k$
  FinishPath1$^+$3$^-(x_1, x_5, k)$

private procedure FinishPath1$^+$3$^-(x_1, x_5, k)$
  if $(x_1 \oplus k \in$ *LeftFreezer*$)$ **then**
    fresh $\leftarrow$ **true**
    LeftFreezer $\leftarrow$ *LeftFreezer*$\backslash\{x_1 \oplus k\}$
  else
    fresh $\leftarrow$ **false**
  end if
  $y_5 \leftarrow k \oplus \mathrm{E}(f^{-1}(k), x_1 \oplus k)$
  TallyEQuery$(f^{-1}(k), x_1 \oplus k, \rightarrow)$
  if $(x_5 \in \mathrm{domain}(P_5))$ **then abort**
  if $(y_5 \in \mathrm{range}(P_5))$ **then abort**
  $P_5(x_5) \leftarrow y_5$
  $P_5^{-1}(y_5) \leftarrow x_5$
  AddQuery$(5, x_5, y_5, \perp)$
  RightQueue $\leftarrow$ *RightQueue* $\cup\ (5^-, x_5)$
  if (*fresh*) **then**
    FreezeRightValues(y_5, k)
  end if

Game G_1 (continued)

private procedure CompletePath3$^+(y_3, k)$
  ... // (symmetric to CompletePath3$^-$)

private procedure CompletePath5$^-(x_5, k)$
  ... // (symmetric to CompletePath1$^+$)

private procedure FinishPath5$^-$3$^+(y_5, y_1, k)$
  ... // (symmetric to FinishPath1$^+$3$^-$)

private procedure TallyEQuery(K, z, dir)
  if $(dir = \rightarrow)$ **then**
    if $(TallyETable[K](z) = \perp)$ **then** $++Eqnum$
    $TallyETable[K](z) \leftarrow t \leftarrow \mathrm{E}(K, z)$
    $TallyETable[K]^{-1}(t) \leftarrow z$
  end if
  if $(dir = \leftarrow)$ **then**
    ... // (symmetric to case $dir = \rightarrow$)
  end if
  if $(Eqnum > 2q^2)$ **then abort**

Fig. 4. The simulated world (fourth of four sets of procedures)

up somewhere in the middle). Given all these combinations, a natural strategy is to have the simulator complete chains whenever it detects *any* k-adjacency. We call this type of simulator *naïve*. The difficulty with the naïve simulator is that, as the path-completion strategy is applied recursively to queries created by the simulator itself, some uncontrollable chain reaction might occur that causes the simulator to create a superpolynomial number of queries, and, thus, lead to an unacceptable simulator running time and to an unacceptably watered-down security bound. Even if such a chain reaction cannot occur, the burden of showing so is on the prover's shoulders, which is not necessarily an easy task. We refer to the general problem of showing that runaway chain reactions do not occur as the problem of *simulator termination*.[8]

To overcome the naïve simulator's problematic termination, we modify the naïve simulator to be more restrained and to complete fewer chains. For this we

[8] Naturally, since the simulator can only create finitely many different i-queries, the simulator is, in general, guaranteed to terminate. Thus "simulator termination" refers, more precisely, to the problem of showing that the simulator only creates polynomially many queries per adversarial query. We prefer the term "termination" to "efficiency" because it seems to more picturesquely capture the threat of an out-of-control chain reaction.

use the "tripwire" concept. Informally, a tripwire is an ordered pair of the form $(i, i+1)$ or $(i+1, i)$ or $(1, 5)$ or $(5, 1)$ (for a 5-round cipher). "Installing a tripwire (i, j)" means the simulator will complete paths for k-adjacencies detected between positions i and j *and for which the j-query is made after the i-query.* (Thus, tripwires are "directed".) As long as no tripwires are triggered, the simulator does nothing; when a tripwire is triggered, the simulator completes the relevant chain(s), and recurses to complete chains for other potentially triggered tripwires, etc. The "naïve" simulator then corresponds to a tripwire simulator with all possible tripwires installed. The tripwire paradigm is essentially due to Coron et al. [30] even while the terminology is ours.

Restricting ourselves to the (fairly broad) class of tripwire simulators, conflicting goals emerge: to install enough tripwires so that the simulator cannot be attacked, while installing few enough tripwires (or in clever enough positions) that a termination argument can be made. Before presenting our own 5-round solution to this dilemma, we briefly justify our choice of five rounds.

Firstly, *no* tripwire simulator with 3 rounds is secure, since it turns out that the naïve 3-round simulator (i.e., with all possible tripwires) can already be attacked. Hence, regardless of termination issues, any 3-round tripwire simulator is insecure. Secondly, we focused on 4-round simulators with four tripwires, as proving termination for five or more tripwires seemed a daunting task. A particularly appealing simulator, here, is the 4-tripwire simulator

$$(1, 4), (4, 1), (2, 3), (3, 2)$$

whose termination can easily be proved by modifying Holenstein et al. termination argument [33], itself adapted from an earlier termination argument of Seurin [32]. Unfortunately it turns out this simulator can be attacked, making it useless. This attack as well as the above-mentioned attack on the 3-round naïve simulator can be found in the full version of this paper [51], where some other attacks on tripwire simulators are also sketched.

Ultimately, the only 4-round, 4-tripwire simulator for which we didn't find an attack is the simulator with the (asymmetric) tripwire configuration

$$(1, 2), (3, 2), (3, 4), (1, 4)$$

(and its symmetric counterpart). However, since we could not foresee a manageable termination argument for this simulator, we ultimately reverted to five rounds. Our 5-round simulator has tripwires

$$(2, 1), (2, 3), (4, 3), (4, 5)$$

(and no tripwires of the form $(1, 5)$ or $(5, 1)$), as sketched in Figure 5. This simulator has the advantage of having a clean (though combinatorially demanding) termination argument, and, as previously discussed, of having excellent efficiency and also better security than the state-of-the-art in "indifferentiable blockcipher" constructions.

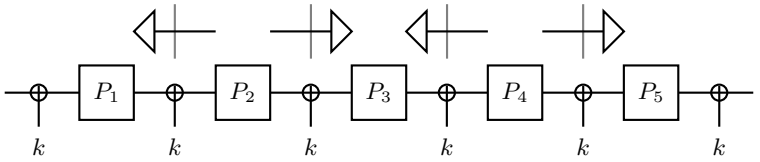


Fig. 5. Tripwire positions for our 5-round simulator. A directed arrow from column P_i to column P_j indicates a tripwire (i, j). The tripwires are $(2, 1)$, $(2, 3)$, $(4, 3)$ and $(4, 5)$.

Some more high-level description of the 5-round simulator. We have already mentioned that our 5-round simulator has tripwires

$$(2, 1), (2, 3), (4, 3), (4, 5).$$

To complete the simulator's description it (mainly) remains to describe how the simulator completes chains, once a tripwire is triggered, since there is some degree of freedom as to which i-query is "adapted" to fit E, etc. Quickly and informally, when a newly created 1-query or 3-query triggers respectively the $(2, 1)$ or $(2, 3)$ tripwire, the relevant path(s) that are completed have their 5-query adapted to fit E. (We note the same query may trigger the completion of several new paths.) Symmetrically, when a newly created 3-query or 5-query triggers a $(4, 3)$ or $(4, 5)$ tripwire, the completed paths have their 1-query adapted to fit E. We note that new 2-queries and 4-queries can never trigger a tripwire, due to the tripwire structure. Moreover, 2- and 4-queries are never adapted, and always have at least one "random endpoint". The latter property turns out to be crucial for various arguments in the proof. It also makes the implementation of the procedures $\mathrm{P2}()$, $\mathrm{P2}^{-1}()$, $\mathrm{P4}()$ and $\mathrm{P4}^{-1}()$ particularly simple, since these do nothing else than lazy sample and return.

The above "quick and informal" summary of the path-completion process is over-simplified because 3-queries can also, in specific situations, be adapted to complete a path. To gain some preliminary intuition about 3-queries, consider a distinguisher D that chooses values x and K and then makes the queries $\mathrm{f}(K) \to k$, $\mathrm{P1}(x \oplus k) \to y_1$, $\mathrm{P2}(y_1 \oplus k) \to y_2$, $\mathrm{E}(K, x) \to y$, $\mathrm{P5}^{-1}(y \oplus k) \to x_5$ and $\mathrm{P4}^{-1}(x_5 \oplus k) \to x_4$. So far, no tripwires have been triggered, but the adversary already knows (e.g., in the real world) that $\mathrm{P3}(y_2 \oplus k) = x_4 \oplus k$, even while the simulator has not yet defined anything internally about P3. Typically, such a situation where the adversary "already knows" something the simulator doesn't are dangerous for the simulator and can lead to attacks; in this case, it turns out the distinguisher cannot use this private knowledge to fool the simulator. It does mean, however, that the simulator needs to be on the lookout for such "pre-defined" 3-queries whenever it answers queries to $\mathrm{P3}()$, $\mathrm{P3}^{-1}()$ or, more generally, whenever it makes a new 3-query internally.

In fact the code used by the simulator to answer 3-queries is altogether rather cautious and sophisticated, even slightly more so than the previous discussion might suggest. To gain further insight into the simulator's handling of 3-queries, consider a distinguisher D' that similarly chooses values x and K and then

makes the queries $\mathrm{f}(K) \to k$, $\mathrm{P1}(x \oplus k) \to y_1$, $\mathrm{P2}(y_1 \oplus k) \to y_2$, $\mathrm{E}(K, x) \to y$ and $\mathrm{P5}^{-1}(y \oplus k) \to x_5$. (So D' makes all the same queries as the distinguisher D above except for the final query $\mathrm{P4}^{-1}(x_5 \oplus k)$, which is *not* made by D'.) At this point, the value $\mathrm{P3}(y_2 \oplus k)$ is not yet pre-defined by E and by the previous queries, since the query $\mathrm{P4}^{-1}(x_5 \oplus k)$ hasn't been made; if D' queries $\mathrm{P3}(y_2 \oplus k) \to y_3$, the simulator might conceivably sample y_3 randomly, and later use the freedom afforded by the missing P4 query to adapt the chain. If the simulator did this, however, the simulator would create a "non-random" 4-query (i.e., a 4-query that doesn't have at least one non-adapted, "random endpoint"), which would wreak havoc within the proof. Instead, when faced with the query $\mathrm{P3}(y_2 \oplus k)$, the simulator detects the situation above and starts by making the "missing" query $\mathrm{P4}^{-1}(x_5 \oplus k) \to x_4$ internally, thus giving the P4-query its required "random endpoint" (at x_4), and finally adapts $\mathrm{P3}(y_2 \oplus k)$ to $x_4 \oplus k$. It so turns out that, with high probability, the simulator is never caught trying to adapt P3() to two different values in this way.

The sets *LeftQueue* and *RightQueue* mentioned in the pseudocode are two queues of queries maintained by the simulator for the purpose of tripwire detection. When a new i-query is created, $i \in \{1, 3\}$, that the simulator believes might set off the $(2, 1)$ or $(2, 3)$ tripwire, the simulator puts this i-query into *LeftQueue*, to be checked later; similarly for $i \in \{3, 5\}$, the simulator puts a newly created i-query into *RightQueue* if it believes this new query might set off a $(4, 3)$ or $(4, 5)$ tripwire. (The same 3-query might end up in both *LeftQueue* and *RightQueue*.) As evidenced by the procedure EmptyQueue() in Fig. 3, *LeftQueue* and *RightQueue* are emptied sequentially and separately, which we choose to do mostly because it offers conceptual advantages within the proof. In the full version of this paper [51] we further discuss how the simulator might come to believe that a newly created i-query will likely *not* set off a tripwire (and thus not put this i-query into the relevant queue(s)), as well give a more detailed discussion of the pseudocode of the simulator.

Due to the space constraints, we similarly leave to the full version [51] a full formal indifferentiability proof of our construction, as well as all other results mentioned in the introduction (e.g., our attacks and the proof of Theorem 2).

References

1. Brachtl, B., Coppersmith, D., Hyden, M., Matyas, S., Meyer, C., Oseas, J., Pilpel, S., Schilling, M.: Data authentication using modification detection codes based on a public one-way encryption function, U.S.Patent No 4.908.861 (1990)
2. Hirose, S.: Some Plausible Constructions of Double-Block-Length Hash Functions. In: Robshaw, M. (ed.) FSE 2006. LNCS, vol. 4047, pp. 210–225. Springer, Heidelberg (2006)
3. Lai, X., Massey, J.L.: Hash Functions Based on Block Ciphers. In: Rueppel, R.A. (ed.) EUROCRYPT 1992. LNCS, vol. 658, pp. 55–70. Springer, Heidelberg (1993)
4. Preneel, B., Govaerts, R., Vandewalle, J.: Hash Functions Based on Block Ciphers: A Synthetic Approach. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 368–378. Springer, Heidelberg (1994)

5. Impagliazzo, R., Levin, L.A., Luby, M.: Pseudo-random Generation from one-way functions. In: ACM Symposium on Theory of Computing, STOC, Seattle, Washington, USA, pp. 12–24. ACM (1989)
6. Goldreich, O., Goldwasser, S., Micali, S.: How to Construct Random Functions. In: 25th Annual Symposium on Foundations of Computer Science, FOCS, West Palm Beach, Florida, USA, pp. 464–479. IEEE Computer Society (1984)
7. Luby, M., Rackoff, C.: How to construct pseudorandom permutations from pseudorandom functions. SIAM Journal of Computing 17, 373–386 (1988)
8. Biryukov, A., Dunkelman, O., Keller, N., Khovratovich, D., Shamir, A.: Key Recovery Attacks of Practical Complexity on AES-256 Variants with up to 10 Rounds. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 299–319. Springer, Heidelberg (2010)
9. Biryukov, A., Khovratovich, D.: Related-Key Cryptanalysis of the Full AES-192 and AES-256. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 1–18. Springer, Heidelberg (2009)
10. Biryukov, A., Khovratovich, D., Nikolić, I.: Distinguisher and Related-Key Attack on the Full AES-256. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 231–249. Springer, Heidelberg (2009)
11. Bogdanov, A., Khovratovich, D., Rechberger, C.: Biclique Cryptanalysis of the Full AES. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 344–371. Springer, Heidelberg (2011)
12. Black, J., Rogaway, P., Shrimpton, T.: Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 320–335. Springer, Heidelberg (2002)
13. Desai, A.: The Security of All-or-Nothing Encryption: Protecting against Exhaustive Key Search. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 359–375. Springer, Heidelberg (2000)
14. Even, S., Mansour, Y.: A Construction of a Cipher from a Single Pseudorandom Permutation. In: Matsumoto, T., Imai, H., Rivest, R.L. (eds.) ASIACRYPT 1991. LNCS, vol. 739, pp. 201–224. Springer, Heidelberg (1993)
15. Granboulan, L.: Short Signatures in the Random Oracle Model. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 364–378. Springer, Heidelberg (2002)
16. Jonsson, J.: An OAEP Variant With a Tight Security Proof. Cryptology ePrint Archive. Report 2002/034 (2002)
17. Kilian, J., Rogaway, P.: How to Protect DES against Exhaustive Key Search (An Analysis of DESX). Journal of Cryptology 14, 17–35 (2001)
18. Phan, D.H., Pointcheval, D.: Chosen-Ciphertext Security without Redundancy. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 1–18. Springer, Heidelberg (2003)
19. Winternitz, R.S.: A Secure One-Way Hash Function Built from DES. In: IEEE Symposium on Security and Privacy, pp. 88–90. IEEE Computer Society (1984)
20. Handschuh, H., Naccache, D.: SHACAL. Submission to the NESSIE Project (2000)
21. Handschuh, H., Naccache, D.: SHACAL: A Family of Block Ciphers. Submission to the NESSIE Project (2002)
22. Black, J.: The Ideal-Cipher Model, Revisited: An Uninstantiable Blockcipher-Based Hash Function. In: Robshaw, M. (ed.) FSE 2006. LNCS, vol. 4047, pp. 328–340. Springer, Heidelberg (2006)
23. Maurer, U., Renner, R., Holenstein, C.: Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 21–39. Springer, Heidelberg (2004)

24. Coron, J.S., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-Damgård Revisited: How to Construct a Hash Function. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 430–448. Springer, Heidelberg (2005)
25. Ristenpart, T., Shacham, H., Shrimpton, T.: Careful with Composition: Limitations of the Indifferentiability Framework. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 487–506. Springer, Heidelberg (2011)
26. Damgård, I.: A Design Principle for Hash Functions. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 416–427. Springer, Heidelberg (1990)
27. Merkle, R.C.: One Way Hash Functions and DES. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 428–446. Springer, Heidelberg (1990)
28. Bellare, M., Ristenpart, T.: Multi-Property-Preserving Hash Domain Extension and the EMD Transform. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 299–314. Springer, Heidelberg (2006)
29. Chang, D., Lee, S., Nandi, M., Yung, M.: Indifferentiable Security Analysis of Popular Hash Functions with Prefix-Free Padding. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 283–298. Springer, Heidelberg (2006)
30. Coron, J.S., Patarin, J., Seurin, Y.: The Random Oracle Model and the Ideal Cipher Model Are Equivalent. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 1–20. Springer, Heidelberg (2008)
31. Dodis, Y., Puniya, P.: On the Relation Between the Ideal Cipher and the Random Oracle Models. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 184–206. Springer, Heidelberg (2006)
32. Seurin, Y.: Primitives et protocoles cryptographiques à sécurité prouvée. PhD thesis, Université de Versailles Saint-Quentin-en-Yvelines, France (2009)
33. Holenstein, T., Künzler, R., Tessaro, S.: The equivalence of the random oracle model and the ideal cipher model, revisited. In: ACM Symposium on Theory of Computing, STOC, San Jose, CA, USA, pp. 89–98. ACM (2011)
34. Aumasson, J., Henzen, L., Meier, W., Phan, R.: SHA-3 proposal BLAKE. Submission to NIST's SHA-3 Competition (2010)
35. Ferguson, N., Lucks, S., Schneier, B., Whiting, D., Bellare, M., Kohno, T., Callas, J., Walker, J.: The Skein Hash Function Family. Submission to NIST's SHA-3 Competition (2010)
36. Daemen, J., Govaerts, R., Vandewalle, J.: Correlation Matrices. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 275–285. Springer, Heidelberg (1995)
37. Daemen, J., Rijmen, V.: The Wide Trail Design Strategy. In: Honary, B. (ed.) Cryptography and Coding 2001. LNCS, vol. 2260, pp. 222–238. Springer, Heidelberg (2001)
38. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Springer (2002)
39. Bogdanov, A., Knudsen, L.R., Leander, G., Standaert, F.X., Steinberger, J., Tischhauser, E.: Key-Alternating Ciphers in a Provable Setting: Encryption Using a Small Number of Public Permutations. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 45–62. Springer, Heidelberg (2012)
40. Dunkelman, O., Keller, N., Shamir, A.: Minimalism in Cryptography: The Even-Mansour Scheme Revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 336–354. Springer, Heidelberg (2012)
41. Dodis, Y., Pietrzak, K., Puniya, P.: A New Mode of Operation for Block Ciphers and Length-Preserving MACs. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 198–219. Springer, Heidelberg (2008)

42. Dodis, Y., Reyzin, L., Rivest, R., Shen, E.: Indifferentiability of Permutation-Based Compression Functions and Tree-Based Modes of Operation, with Applications to MD6. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 104–121. Springer, Heidelberg (2009)
43. Coron, J.S., Dodis, Y., Mandal, A., Seurin, Y.: A Domain Extender for the Ideal Cipher. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 273–289. Springer, Heidelberg (2010)
44. Bertoni, G., Daemen, J., Peeters, M., Assche, G.: The KECCAK sponge function family. Submission to NIST's SHA-3 Competition (2011)
45. Gauravaram, P., Knudsen, L., Matusiewicz, K., Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.: Grøstl – a SHA-3 candidate. Submission to NIST's SHA-3 Competition (2011)
46. Wu, H.: The Hash Function JH. Submission to NIST's SHA-3 Competition (2011)
47. Rogaway, P., Steinberger, J.P.: Constructing Cryptographic Hash Functions from Fixed-Key Blockciphers. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 433–450. Springer, Heidelberg (2008)
48. Rogaway, P., Steinberger, J.P.: Security/Efficiency Tradeoffs for Permutation-Based Hashing. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 220–236. Springer, Heidelberg (2008)
49. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: On the Indifferentiability of the Sponge Construction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 181–197. Springer, Heidelberg (2008)
50. Lee, J., Hong, D.: Collision Resistance of the JH Hash Function. IEEE Transactions on Information Theory 58, 1992–1995 (2012)
51. Andreeva, E., Bogdanov, A., Dodis, Y., Mennink, B., Steinberger, J.P.: On the Indifferentiability of Key-Alternating Ciphers. In: Micciancio, D., Peikert, C. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 526–545. Springer, Heidelberg (2013)
52. Lampe, R., Seurin, Y.: How to Construct an Ideal Cipher from a Small Set of Public Permutations. Cryptology ePrint Archive. Report 2013/255 (2013)
53. Dodis, Y., Ristenpart, T., Shrimpton, T.: Salvaging Merkle-Damgård for Practical Applications. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 371–388. Springer, Heidelberg (2009)
54. Miles, E., Viola, E.: Substitution-Permutation Networks, Pseudorandom Functions, and Natural Proofs. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 68–85. Springer, Heidelberg (2012)
55. Knudsen, L.: Block Ciphers - The Basics, ECRYPT II Summer School on Design and Security of Cryptographic Algorithms and Devices (2011) (invited talk)
56. Demay, G., Gaži, P., Hirt, M., Maurer, U.: Resource-Restricted Indifferentiability. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 664–683. Springer, Heidelberg (2013)

Plain versus Randomized Cascading-Based Key-Length Extension for Block Ciphers

Peter Gaži

ETH Zurich, Switzerland
Department of Computer Science
peter.gazi@inf.ethz.ch

Abstract. Cascading-based constructions represent the predominant approach to the problem of key-length extension for block ciphers. Besides the plain cascade, existing works also consider its modification containing key-whitening steps between the invocations of the block cipher, called randomized cascade or XOR-cascade. We contribute to the understanding of the security of these two designs by giving the following attacks and security proofs, assuming an underlying ideal block cipher with key length κ and block length n:

- For the plain cascade of odd (resp. even) length ℓ we present a generic attack requiring roughly $2^{\kappa+\frac{\ell-1}{\ell+1}n}$ (resp. $2^{\kappa+\frac{\ell-2}{\ell}n}$) queries, being a generalization of both the meet-in-the-middle attack on double encryption and the best known attack on triple cascade.
- For XOR-cascade of odd (resp. even) length ℓ we prove security up to $2^{\kappa+\frac{\ell-1}{\ell+1}n}$ (resp. $2^{\kappa+\frac{\ell-2}{\ell}n}$) queries and also an improved bound $2^{\kappa+\frac{\ell-1}{\ell}n}$ for the special case $\ell \in \{3,4\}$ by relating the problem to the security of key-alternating ciphers in the random-permutation model.
- Finally, for a natural class of *sequential* constructions where block-cipher encryptions are interleaved with key-dependent permutations, we show a generic attack requiring roughly $2^{\kappa+\frac{\ell-1}{\ell}n}$ queries. Since XOR-cascades are sequential, this proves tightness of our above result for XOR-cascades of length $\ell \in \{3,4\}$ as well as their optimal security within the class of sequential constructions.

These results suggest that XOR-cascades achieve a better security/efficiency trade-off than plain cascades and should be preferred.

Keywords: Provable security, block ciphers, key-length extension, ideal-cipher model, cascade, XOR-cascade.

1 Introduction

1.1 Block Ciphers and the Key-Length Extension Problem

It is beyond question that block ciphers play a pivotal role in cryptographic practice, being the basic building block for most constructions in the realm

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 551–570, 2013.

of symmetric cryptography. The first standardized block cipher achieving huge popularity and wide-spread use was DES [1], nowadays being replaced by the current standard AES [4].

Formally, a block cipher with keyspace $\{0,1\}^\kappa$ and message space $\{0,1\}^n$ is simply a family of efficiently computable (and invertible) permutations E_k on the set of n-bit strings indexed by a κ-bit key k, which is often emphasized by referring to it as a (κ, n)-block cipher. For example, $n = 64$ and $\kappa = 56$ for DES, and $n = 128$ and $\kappa \in \{128, 192, 256\}$ for AES.

In most applications that employ a block cipher as its underlying primitive, it is assumed (and required) that it behaves as a *pseudorandom permutation* (PRP), i.e., if used with a random secret key, it cannot be efficiently distinguished from a uniformly random permutation. To capture this notion, the PRP security level of a block cipher is defined as the complexity required to distinguish it from a random permutation with non-negligible advantage.

Key-Length Extension. The key length κ is a crucial security parameter of every block cipher E. An attacker, given some plaintext-ciphertext pairs, can easily identify the secret key being used by a brute-force attack if he is capable of performing roughly 2^κ evaluations of E. This key-recovery attack can be also transformed into a PRP distinguishing attack, implying that the bound of 2^κ evaluations limits the PRP security of every block cipher. This represents a problem for existing block ciphers with small key length κ for which 2^κ operations can no longer be considered beyond the available computational power of a potential attacker.

A prominent example of such a design is the former standard DES, which however, apart from its insufficient key size, is believed to contain no significant structural weaknesses. It also remains attractive thanks to its short block length which allows enciphering short inputs and explains the wide-spread use of DES-based constructions in the financial industry even today (see e.g. [6] for the EMV standard).

Due to the above reasons, there exists a practical demand for constructions transforming any (κ, n)-block cipher E into a (κ', n)-block cipher C^E while increasing both the key length (i.e., $\kappa' > \kappa$) and the generic security achieved (i.e., the PRP security of C^E should be significantly higher than 2^κ assuming that E itself contains no non-generic weaknesses). This is known as the *key-length extension problem* for block ciphers and in this paper we contribute to the understanding and analysis of several cascading-based constructions addressing this problem. Note that even though the case of DES constituted the initial motivation for the study of key-length extension, we focus on generic constructions that are applicable to any block cipher, making our results attractive also from a theoretic perspective.

Ideal-Cipher Model. To assess the security level achieved by the key-length extension constructions themselves, we assume the absence of any weaknesses of the underlying block cipher by modelling it as the *ideal* block cipher $\mathbf{E}$ providing an independent uniformly random permutation for each key. We consider a distinguisher $\mathbf{D}$ that is allowed to issue two types of queries:

- *block-cipher queries* to evaluate the block cipher $\mathbf{E}$ under any key and on any input block (both in the encryption and the decryption direction).
- *construction queries* to evaluate either the key-length extending construction $\mathsf{C}^{\mathbf{E}}_{K'}$ used with the block cipher $\mathbf{E}$ and a uniformly random secret κ'-bit key K'; or a uniform random permutation $\mathbf{P}$ independent of $\mathbf{E}$ (again, both query directions are allowed).

Hence, the distinguisher is either given to interact with the combined system $(\mathbf{E}, \mathsf{C}^{\mathbf{E}}_{K'})$ or with $(\mathbf{E}, \mathbf{P})$ and its goal is to decide which of these two situations has occurred. Its complexity is determined solely by the sum of its queries of both types, leading to results of information-theoretic nature. Note that the security of *any* key-length extension construction in this model can be upper-bounded by $2^{\kappa+n}$ which corresponds to the trivial attack asking all possible block-cipher and construction queries. This model has already been employed numerous times to analyze the security of key-length extending constructions, e.g. in [19,10,16,18].

1.2 Plain and Randomized Cascades

Arguably the most natural way to approach the key-length extension problem is to simply apply the block cipher several times using an independent key at each step – an approach known as *cascading*. Its security has been a subject of extensive study in various models, including the information-theoretic ideal-cipher model described above. It is well known that a cascade of length two does not substantially increase security due to the meet-in-the-middle attack [12], even though a security increase in terms of distinguishing advantage is achieved for low attack complexities, as shown in [7]. This makes triple encryption the shortest cascade with a potential for significant security gain, resulting into its widespread usage as the *Triple-DES* (3DES) standard [2,3,5]. Given keys $k_1, k_2, k_3 \in \{0,1\}^{56}$, 3DES encrypts a 64-bit message m as

$$3\mathrm{DES}_{k_1,k_2,k_3}(m) = \mathrm{DES}_{k_3}(\mathrm{DES}_{k_2}(\mathrm{DES}_{k_1}(m))) \; .$$

3DES was formally studied by Bellare and Rogaway [10], showing its security up to roughly $2^{\kappa+\min\{\kappa,n\}/2}$ queries when DES is replaced by an ideal block cipher. Gaži and Maurer [16] showed that the security lower bound increases further with the length of the cascade for block ciphers where $\kappa \leq n$, reaching roughly $2^{\min\left\{\frac{2\ell\kappa}{\ell+1}, \kappa+\frac{n}{2}\right\}}$ queries for a cascade of odd length ℓ; with increasing ℓ this term approaches $2^{\min\left\{2\kappa, \kappa+\frac{n}{2}\right\}}$. Recently it was shown by Lee [22] that the security of the cascade actually approaches the value $2^{\kappa+\min\{\kappa,n\}}$ with increasing ℓ, however his result only gives useful bounds for large ℓ (say $\ell \geq 16$). On the negative side, Lucks [23] presented an attack on triple encryption that, once cast into the ideal-cipher model, constitutes the best such attack known in this model by requiring roughly $2^{\kappa+n/2}$ queries.

An alternative approach to the keylength-extension problem is inspired by the key-whitening technique, first employed in the DESX construction due to Rivest. Here, the input and output of the block cipher is masked ("whitened")

by an XOR with additional key material as follows: given a key tuple $(k_i, k_o, k) \in \left(\{0,1\}^{64}\right)^2 \times \{0,1\}^{56}$ a message m is mapped to

$$\mathrm{DESX}_{k_i,k_o,k}(m) = k_o \oplus \mathrm{DES}_k(k_i \oplus m).$$

The generalization of DESX for arbitrary κ, n was shown to be secure up to $2^{\frac{\kappa+n}{2}}$ queries by Kilian and Rogaway [19] even if the same key is used in both whitening steps.

In an attempt to combine cascading and key whitening, Gaži and Tessaro [18] proposed the so-called 2-XOR-cascade (or randomized cascade) construction. It consists of a cascade of length 2 interleaved with two whitening steps, mapping each n-bit message m under a key $(k, z) \in \{0,1\}^k \times \{0,1\}^n$ to

$$2\mathrm{XOR}_{k,z}(m) = E_{\widetilde{k}}(E_k(m \oplus z) \oplus z)$$

where $\widetilde{k}$ is derived from k in a deterministic way (e.g. by flipping a single bit). They prove 2-XOR-cascade to be secure up to $2^{\kappa+n/2}$ queries and also show that this bound is tight. The recent independent work by Lee [22] considers the general case of XOR-cascade of length ℓ (with independent keys and an XOR step at the end) and proves that its security approaches the optimal bound $2^{\kappa+n}$, while again giving useful statements only for large ℓ.

Other Models. There is a vast amount of literature on the security properties of different cascading-based constructions for block ciphers in various security models, in the information-theoretic setting [14,25,31,26,27,17] as well as in the computational setting [28,30,13]. The models employed in these works are however orthogonal to ours and hence the results are not directly comparable.

1.3 Our Contributions

Cascades. We start our investigation by looking at the case of a plain cascade construction of a general length ℓ (see Fig. 2). As a complement to the above-mentioned positive results given in [16,22], in Section 3 we present a generic attack on ℓ-cascade in our model that requires roughly $2^{\kappa+\frac{\ell-2}{\ell}n}$ queries ($2^{\kappa+\frac{\ell-1}{\ell+1}n}$ queries) for even (odd) ℓ. The well-known meet-in-the-middle attack [12] and the attack of Lucks [23] turn out to be special cases of our attack for $\ell = 2$ and $\ell = 3$, respectively. To the best of our knowledge, our result also constitutes the first formal analysis of the advantage achieved by the often-cited attack on triple encryption [23].

XOR-Cascades. After upper-bounding the security of the seemingly simplest possible construction — the cascade — we turn our attention to the more involved ℓ-XOR-cascade constructions of arbitrary length ℓ (see Fig. 4) which are a generalization of the 2-XOR-cascade proposed in [18].

In Section 4 we give a general method to reduce the security of XOR-cascades in our model to the security of so-called *key-alternating ciphers* in the random-permutation model. A key-alternating cipher (KAC) is a block cipher designed

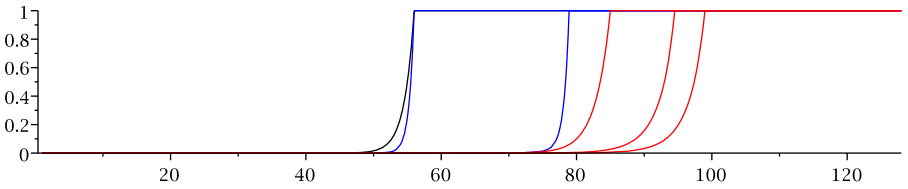


Fig. 1. Upper bounds on distinguishing advantage versus $\log_2 q$ (where q is the number of queries) for plain (blue) and randomized (red) cascades of lengths 2–4, using $\kappa = 56$ and $n = 64$. Curves from left to right: (1) single encryption for reference; (2) 2-cascade; (3) 3- and 4-cascade (same bound); (4) 2-XOR-cascade; (5) 3-XOR-cascade; (6) 4-XOR-cascade.

to alternate keyed XOR operations with fixed publicly known permutations (see Fig. 5). Since AES represents a prominent practical example of this design paradigm, its security has been extensively studied [11,29,20,8,21]. However, despite the seeming closeness to the structure of XOR-cascades, these two topics were never related to each other explicitly.

Our reduction relates the security of an XOR-cascade to the security of one step shorter KAC, allowing for more modular security analysis of XOR-cascades. By combining it with recent lower bounds on the security of KAC [11,29,20] we obtain a proof that 3-XOR-cascade and 4-XOR-cascade are secure up to $2^{\kappa+\frac{2}{3}n}$ and $2^{\kappa+\frac{3}{4}n}$ queries, respectively; and finally, that a general ℓ-XOR-cascade of odd (even) length is secure at least up to $2^{\kappa+\frac{\ell-1}{\ell+1}n}$ queries ($2^{\kappa+\frac{\ell-2}{\ell}n}$ queries), respectively.

The latter result implies that the security of XOR-cascades with increasing length approaches the optimum $2^{\kappa+n}$. While this also follows from the independent concurrent work [22], the bound presented in [22] is not applicable to small, practical values of ℓ (exceeding the trivial security 2^κ only for $\ell \geq 5$). Using the DES parameters for illustration, our bound for 3-XOR-cascade is roughly matched by the bound in [22] for 12-XOR-cascade. We also see the modular approach of our proof as an advantage, exhibiting a general reduction to the recently studied security of KAC. On the other hand, the result from [22] gives a better bound for large values of ℓ (e.g. $\ell > 22$ for DES parameters).

Contrasting our results with the generic attacks on plain cascades given in Section 3, we see that a 3-XOR-cascade is provably at least as secure as a 6-cascade and a 4-XOR-cascade is at least as secure as an 8-cascade, while providing much better efficiency. This gives us a more robust argument in favor of XOR-cascades as constructions providing security and efficiency at the same time; a view that was already advocated in [18]. Note that here we are comparing security lower bounds (for XOR-cascades) to best known attacks (for plain cascades), making an even stronger case for the randomization. Alternatively, one can compare the upper bounds on distinguishing advantages for the constructions considered, we present one such comparison in Fig. 1.

Table 1. Best known security lower bounds and generic attacks for various key-length extension schemes. Each given term is a logarithm of the respective number of queries and is parameterized by the key length κ and block size n of the underlying block cipher. References and further details to all depicted bounds are given in the text. Results denoted by $(\star)$ come from this paper.

ℓ	ℓ-cascade		ℓ-XOR-cascade	sequential ℓ-query construction
	security	attack	security	attack
2	$\min\{\kappa, n\}$	κ	$\kappa + \frac{n}{2}$	$\kappa + \frac{n}{2}$
3	$\kappa + \min\left\{\frac{\kappa}{2}, \frac{n}{2}\right\}$	$\kappa + \frac{n}{2}$	$\kappa + \frac{2}{3}n$ $(\star)$	$\kappa + \frac{2}{3}n$ $(\star)$
4	$\kappa + \min\left\{\frac{\kappa}{2}, \frac{n}{2}\right\}$	$\kappa + \frac{n}{2}$ $(\star)$	$\kappa + \frac{3}{4}n$ $(\star)$	$\kappa + \frac{3}{4}n$ $(\star)$
odd ≥ 5[1]	$\min\left\{\frac{2\ell\kappa}{\ell+1}, \kappa + \frac{n}{2}\right\}$	$\kappa + \frac{\ell-1}{\ell+1}n$ $(\star)$	$\kappa + \frac{\ell-1}{\ell+1}n$ $(\star)$	$\kappa + \frac{\ell-1}{\ell}n$ $(\star)$
even	$\min\left\{\frac{2(\ell-1)\kappa}{\ell}, \kappa + \frac{n}{2}\right\}$	$\kappa + \frac{\ell-2}{\ell}n$	$\kappa + \frac{\ell-2}{\ell}n$	

Sequential Constructions. Motivated by the question of tightness of the above-mentioned bounds for XOR-cascades, we proceed by investigating generic attacks on a particular class of key-length extending constructions that include them. In Section 5 we look at constructions issuing ℓ queries to the block cipher while working in a sequential way: they consist of ℓ block-cipher encryptions interleaved with applications of arbitrary permutations that only depend on the key being used. For this class of constructions that we call *sequential* we exhibit an attack requiring approximately $2^{\kappa+\frac{\ell-1}{\ell}n}$ queries. Since XOR-cascades clearly belong to the class of sequential constructions, an ℓ-XOR-cascade cannot be secure beyond $2^{\kappa+\frac{\ell-1}{\ell}n}$ queries. This shows that the obtained security bounds for $\ell \in \{3, 4\}$ are tight and moreover, the ℓ-XOR-cascades of this length are optimally secure among the class of all sequential constructions, emphasizing that the extremely cheap XOR operation is sufficient to achieve the full potential of sequential constructions. This was previously only shown for $\ell = 2$ in [18].

Summary. Table 1 summarizes the results of this paper in the context of previously known results. To serve as an overview, most bounds are presented in a simplified form. For numerical illustration, let us again consider the DES parameters and the case $\ell = 3$. For 3-cascade we have a security lower bound of $2^{78.4}$ [10,16] and an upper bound of $2^{89.6}$ queries due to the attack [23] analyzed here. For 3-XOR-cascade the lower and upper bounds are $2^{93.0}$ and $2^{102.4}$ queries, respectively, both obtained in this paper. In both cases, lower bounds are threshold values where the advantage bound reaches $1/2$, upper bounds are numbers of queries required by our attacks resulting in advantage at least $1/2$.

[1] For large ℓ (e.g. $\ell > 16$ for cascade and $\ell > 22$ for XOR-cascade, considering DES parameters) the security lower bounds are superseded by the results in [22].

Finally, note that all generic attacks presented in this paper can be mounted even if the distinguisher is only allowed to ask forward construction queries. Moreover, these queries can be chosen arbitrarily, resulting in *known-plaintext attacks*. In contrast, our security proofs are valid also with respect to an adaptive adversary allowed to ask also inverse construction queries (CCA adversary).

2 Preliminaries

2.1 Basic Notation

We typically denote sets by calligraphic letters $\mathcal{X}, \mathcal{Y}, \ldots$, and by $|\cdot|$ we denote their cardinalities. The set of all k-tuples $x^k = (x_1, \ldots, x_k)$ of elements of $\mathcal{X}$ is denoted by $\mathcal{X}^k$. The symbols $\mathrm{Func}(m, \ell)$ and $\mathrm{Perm}(n)$ refer to the sets of all functions from $\{0,1\}^m$ to $\{0,1\}^\ell$ and of all permutations of $\{0,1\}^n$, respectively; while $id \in \mathrm{Perm}(n)$ represents the identity mapping when n is implicit. All logarithms are understood to the base 2.

Random variables and concrete values they can take are usually denoted by upper-case letters $X, Y, \ldots$ and lower-case letters $x, y, \ldots$, respectively. For events A and B and random variables U and V with ranges $\mathcal{U}$ and $\mathcal{V}$, respectively, we denote by $\mathsf{P}_{UA|VB}$ the corresponding conditional probability distribution, seen as a (partial) function $\mathcal{U} \times \mathcal{V} \to [0,1]$. The value $\mathsf{P}_{UA|VB}(u,v) = \mathsf{P}[U = u \wedge A | V = v \wedge B]$ is well-defined for all $u \in \mathcal{U}$ and $v \in \mathcal{V}$ such that $\mathsf{P}_{VB}(v) > 0$ and undefined otherwise. Two probability distributions P_U and $\mathsf{P}_{U'}$ on the same set $\mathcal{U}$ are equal, denoted $\mathsf{P}_U = \mathsf{P}_{U'}$, if $\mathsf{P}_U(u) = \mathsf{P}_{U'}(u)$ for all $u \in \mathcal{U}$. Conditional probability distributions are equal if the equality holds for all arguments for which both of them are defined. To emphasize the random experiment $\mathcal{E}$ in consideration, we sometimes write it in the superscript, e.g. $\mathsf{P}^{\mathcal{E}}_{U|V}(u,v)$. The expected value of a discrete random variable X is denoted by $\mathsf{E}(X) = \sum_{x \in \mathcal{X}} (x \cdot \mathsf{P}[X = x])$. The complement of an event A is denoted by $\overline{A}$.

2.2 Random Systems

To present our results we make use of Maurer's random systems framework [24], which we now introduce in a self-contained exposition sufficient to follow the rest of the paper.

We start by observing that the input-output behavior of any kind of reactive discrete system with inputs in $\mathcal{X}$ and outputs in $\mathcal{Y}$ can be described by an infinite family of functions specifying, for each $i \geq 1$, the probability distribution of the system's i-th output $Y_i \in \mathcal{Y}$ given the values of the first i inputs $X^i \in \mathcal{X}^i$ and the previous $i-1$ outputs $Y^{i-1} \in \mathcal{Y}^{i-1}$. Using this viewpoint, we say that an $(\mathcal{X}, \mathcal{Y})$*-(random) system* $\mathbf{F}$ is an infinite sequence of functions $\mathsf{p}^{\mathbf{F}}_{Y_i|X^iY^{i-1}} \colon \mathcal{Y} \times \mathcal{X}^i \times \mathcal{Y}^{i-1} \to [0,1]$ such that $\sum_{y_i} \mathsf{p}^{\mathbf{F}}_{Y_i|X^iY^{i-1}}(y_i, x^i, y^{i-1}) = 1$ for all $i \geq 1$, $x^i \in \mathcal{X}^i$ and $y^{i-1} \in \mathcal{Y}^{i-1}$. Note that $\mathsf{p}^{\mathbf{F}}_{Y_i|X^iY^{i-1}}$ by itself does not represent a (conditional) probability distribution in any particular random experiment with well-defined random variables Y_i, X^i, Y^{i-1} until the system is connected to a

distinguisher (see below), in which case these random variables will exist and take the role of the transcript. We shall typically define discrete systems by a high level description, as long as the resulting conditional probability distributions could be derived easily from this description. A system $\mathbf{F}$ is *deterministic* if the range of $\mathsf{p}^{\mathbf{F}}_{Y_i|X^iY^{i-1}}$ is $\{0,1\}$ for all $i \geq 1$. Moreover, it is *stateless* if the probability distribution of each output depends only on the current input, i.e., if there exists a distribution $\mathsf{p}_{Y|X} : \mathcal{Y} \times \mathcal{X} \to [0,1]$ such that $\mathsf{p}^{\mathbf{F}}_{Y_i|X^iY^{i-1}}(y_i, x^i, y^{i-1}) = \mathsf{p}_{Y|X}(y_i, x_i)$ for all y_i, x^i and y^{i-1}.

A system $\mathbf{F}$ might often be used as a component (subsystem) in a construction $\mathsf{C}^{(\cdot)}$, resulting in the composed system $\mathsf{C}^{\mathbf{F}}$. While a construction $\mathsf{C}^{(\cdot)}$ does not define a random system by itself, $\mathsf{C}^{\mathbf{F}}$ does define a random system. The notions of being deterministic and of being stateless naturally extend to constructions.[2] Two (possibly dependent) systems $\mathbf{F}$ and $\mathbf{G}$ can also be composed in parallel, denoted $(\mathbf{F}, \mathbf{G})$, which simply results in a system that allows queries to both systems $\mathbf{F}$ and $\mathbf{G}$.

Examples. A special case of a random system is a *random function* $\mathbf{F} : \mathcal{X} \to \mathcal{Y}$ that implements a function f initially chosen according to some distribution on the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$.[3] In particular, the *uniform random function (URF)* $\mathbf{R} : \{0,1\}^m \to \{0,1\}^\ell$ realizes a uniformly chosen function $f \in \mathrm{Func}(m,\ell)$, and the *uniform random permutation (URP)* $\mathbf{P} : \{0,1\}^n \times \{+,-\} \to \{0,1\}^n$ realizes a uniformly chosen permutation $\pi \in \mathrm{Perm}(n)$ allowing both forward queries of the form $(x,+)$ returning $\pi(x)$ as well as backward queries $(y,-)$ returning $\pi^{-1}(y)$. Throughout this paper we meet the convention that any system realizing a random function (possibly by means of a construction) which is a permutation will *always* allow both forward and backward queries. Furthermore, by $\mathbf{E} : \{0,1\}^\kappa \times \{0,1\}^n \times \{+,-\} \to \{0,1\}^n$ we denote the random function realizing an *ideal block cipher* that provides an independent uniform random permutation $\mathbf{E}_k \in \mathrm{Perm}(n)$ for each key $k \in \{0,1\}^\kappa$, allowing both forward and backward queries to each $\mathbf{E}_k$. Finally, note that with some abuse of notation, we often write $\mathbf{E}_k$ or $\mathbf{P}$ to refer to the randomly chosen permutation P implemented by the system $\mathbf{E}_k$ or $\mathbf{P}$, respectively.

Distinguishing Random Systems. A *distinguisher* $\mathbf{D}$ for an $(\mathcal{X},\mathcal{Y})$-random system asking q queries is a $(\mathcal{Y},\mathcal{X})$-random system which is "one query ahead:" its input-output behavior is defined by the conditional probability distributions of its queries $\mathsf{p}^{\mathbf{D}}_{X_i|X^{i-1}Y^{i-1}}$ for all $1 \leq i \leq q$. (Its first query is determined by $\mathsf{p}^{\mathbf{D}}_{X_1}$.) After the distinguisher asks all q queries, it outputs a bit W_q depending on the transcript (X^q, Y^q). Given a random system $\mathbf{F}$ and a distinguisher $\mathbf{D}$, we denote by $\mathbf{DF}$ the random experiment where $\mathbf{D}$ interacts with $\mathbf{F}$, with the distributions of the transcript (X^q, Y^q) and of the bit W_q being uniquely defined by their conditional probability distributions. For two $(\mathcal{X},\mathcal{Y})$-random systems

[2] We dispense with a formal definition. However, we point out that we allow a stateless construction to keep a state during invocations of its subsystem.

[3] As for the notion of a random variable or a random system, the word "random" does not imply any uniformity of the distribution.

$\mathbf{F}$ and $\mathbf{G}$, the *distinguishing advantage* of $\mathbf{D}$ in distinguishing systems $\mathbf{F}$ and $\mathbf{G}$ by q queries is the quantity $\Delta^{\mathbf{D}}(\mathbf{F},\mathbf{G}) = |\mathsf{P}^{\mathbf{DF}}_{W_q}(1) - \mathsf{P}^{\mathbf{DG}}_{W_q}(1)|$ and the maximal distinguishing advantage over all distinguishers asking q queries is denoted by $\Delta_q(\mathbf{F},\mathbf{G}) = \max_{\mathbf{D}} \Delta^{\mathbf{D}}(\mathbf{F},\mathbf{G})$ (with $\mathbf{D}$ ranging over all such distinguishers).

If a detailed description of some distinguisher's internal workings is needed, we use standard pseudocode notation (see e.g. Fig. 3). To capture that the distinguisher issues a query x to a system $\mathbf{F}$ and stores the response as y we always use the explicit notation "**query** $y := \mathbf{F}(x)$".

Monotone Conditions. For a random system $\mathbf{F}$, we often consider an internal *monotone condition* defined on it. Such a condition is initially satisfied (true), but once it gets violated, it cannot become true again (hence the name monotone). We use such conditions to capture whether the behavior of the system meets some additional requirement (e.g. distinct outputs, consistent outputs) or this was already violated during the interaction that occurred so far. A monotone condition is formalized by a sequence of events $\mathcal{A} = A_0, A_1, \ldots$ such that A_0 always holds, and A_i holds if the condition holds after answering the i-th query. The probability that a distinguisher $\mathbf{D}$ issuing q queries to $\mathbf{F}$ makes a monotone condition $\mathcal{A}$ fail in the random experiment $\mathbf{DF}$ is denoted by $\nu^{\mathbf{D}}(\mathbf{F},\overline{A_q}) = \mathsf{P}^{\mathbf{DF}}(\overline{A_q})$ and maximum over all such distinguishers is denoted by $\nu(\mathbf{F},\overline{A_q}) = \max_{\mathbf{D}} \nu^{\mathbf{D}}(\mathbf{F},\overline{A_q})$.

For any random system $\mathbf{F}$ with a monotone condition $\mathcal{A}$ defined on it, following [27] we define $\mathbf{F}$ *blocked by* $\mathcal{A}$ to be a new random system that behaves exactly like $\mathbf{F}$ as long as the condition $\mathcal{A}$ is satisfied; but once $\mathcal{A}$ is violated, it only outputs a special blocking symbol $\perp$ not contained in the output alphabet of $\mathbf{F}$. We will make use of the following helpful claims on random systems proven in previous works.

Lemma 1. *Let $\mathsf{C}^{(\cdot)}$ and $\mathsf{C}'^{(\cdot)}$ be two constructions invoking a subsystem, and let $\mathbf{F}$ and $\mathbf{G}$ be random systems. Let $\mathcal{A}$ and $\mathcal{B}$ be two monotone conditions defined on $\mathbf{F}$ and $\mathbf{G}$, respectively.*

(i) [16, Lemma 2] Let $\mathbf{F}^{\perp}$ denote the random system $\mathbf{F}$ blocked by $\mathcal{A}$ and let $\mathbf{G}^{\perp}$ denote $\mathbf{G}$ blocked by $\mathcal{B}$. Then for every distinguisher $\mathbf{D}$ asking q queries we have $\Delta^{\mathbf{D}}(\mathbf{F},\mathbf{G}) \leq \Delta_q(\mathbf{F}^{\perp},\mathbf{G}^{\perp}) + \nu^{\mathbf{D}}(\mathbf{F},\overline{A_q})$.

(ii) [24, Lemma 5] $\Delta_q(\mathsf{C}^{\mathbf{F}},\mathsf{C}^{\mathbf{G}}) \leq \Delta_{q'}(\mathbf{F},\mathbf{G})$, where q' is the maximum number of invocations of any internal system $\mathbf{H}$ for any sequence of q queries to $\mathsf{C}^{\mathbf{H}}$, if such a value is defined.

(iii) [16, Lemma 3] There exists a fixed permutation $S \in \mathrm{Perm}(n)$ (represented by a deterministic stateless system) such that $\Delta_q(\mathsf{C}^{\mathbf{P}},\mathsf{C}'^{\mathbf{P}}) \leq \Delta_q(\mathsf{C}^{S},\mathsf{C}'^{S})$.

3 Plain Cascades

We start by investigating the security of the plain cascade construction. Given the lower bounds on the security of plain cascades given in [16,22], it is natural to approach the question from the opposite direction and explore generic attacks on the cascade construction in our model. In this section we describe such an

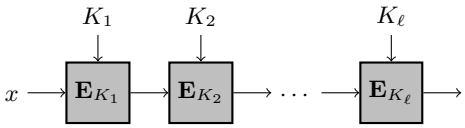


Fig. 2. The cascade construction realized by $\mathsf{Casc}^{\mathbf{E}}_{\ell,\bar{K}}$

attack for the general case of a cascade of arbitrary length $\ell \geq 2$. It shows that, roughly speaking, plain cascade of length ℓ can be attacked in $2^{\kappa+\frac{\ell-2}{\ell}n}$ queries ($2^{\kappa+\frac{\ell-1}{\ell+1}n}$ queries) for even (odd) ℓ.

Let $\mathsf{Casc}^{(\cdot)}_{\ell}\colon (\{0,1\}^{\kappa})^{\ell} \times \{0,1\}^{n} \times \{+,-\} \to \{0,1\}^{n}$ denote a (deterministic stateless) construction which expects a subsystem $\mathbf{E}\colon \{0,1\}^{\kappa} \times \{0,1\}^{n} \times \{+,-\} \to \{0,1\}^{n}$ realizing a block cipher. $\mathsf{Casc}^{\mathbf{E}}_{\ell}$ then realizes cascaded encryption of length ℓ using the block cipher $\mathbf{E}$ and the keys given, i.e., $\mathsf{Casc}^{\mathbf{E}}_{\ell}$ answers each forward query $(k_1,\ldots,k_\ell,x,+)$ by $\mathbf{E}_{k_\ell}(\cdots\mathbf{E}_{k_1}(x)\cdots)$ and each backward query $(k_1,\ldots,k_\ell,y,-)$ by $\mathbf{E}^{-1}_{k_1}(\cdots\mathbf{E}^{-1}_{k_\ell}(y)\cdots)$. Moreover, we let $\mathsf{Casc}^{\mathbf{E}}_{\ell,\bar{K}}$ be the system that chooses a uniformly random (secret) key tuple $\bar{K} = (K_1,\ldots,K_\ell) \in (\{0,1\}^{\kappa})^{\ell}$ and then gives access to the permutation $\mathsf{Casc}^{\mathbf{E}}_{\ell}(\bar{K},\cdot)$ in both directions (i.e., takes inputs from $\{0,1\}^{n}\times\{+,-\}$). The evaluation of a forward query by $\mathsf{Casc}^{\mathbf{E}}_{\ell,\bar{K}}$ is depicted in Fig. 2.

Theorem 1. *For the cascade construction* $\mathsf{Casc}^{(\cdot)}_{\ell,\bar{K}}$ *of even length* $\ell \geq 2$ *using an ideal block cipher* $\mathbf{E}$ *and for any*[4] *parameter* $0 < t < 2^{2n/\ell-1}$ *there exists a distinguisher* $\mathbf{D}$ *such that*

$$\Delta^{\mathbf{D}}\left((\mathbf{E},\mathsf{Casc}^{\mathbf{E}}_{\ell,\bar{K}}),(\mathbf{E},\mathbf{P})\right) \geq 1 - \frac{2}{t} - 2^{\ell\kappa - t(n-1)}$$

and $\mathbf{D}$ *asks at most* $\ell \cdot 2^{\kappa+\frac{\ell-2}{\ell}n}$ *queries to* $\mathbf{E}$ *and* $2t \cdot 2^{\frac{\ell-2}{\ell}n}$ *forward queries to either of* $\mathsf{Casc}^{\mathbf{E}}_{\ell,\bar{K}}$ *and* $\mathbf{P}$. *For odd-length cascades,* $\mathbf{D}$ *requires at most* $\ell \cdot 2^{\kappa+\frac{\ell-1}{\ell+1}n}$ *queries to* $\mathbf{E}$ *and* $2t \cdot 2^{\frac{\ell-1}{\ell+1}n}$ *forward queries to either of* $\mathsf{Casc}^{\mathbf{E}}_{\ell,\bar{K}}$ *and* $\mathbf{P}$.

Our proof relies on the following technical lemma proven in the full version of this paper. Let E, Var and Cov denote the usual notions of expected value, variance and covariance, respectively.

Lemma 2. *Let* $\mathcal{U}$ *be a set such that* $|\mathcal{U}| = N$ *and for* $m \in \mathbb{N}$ *let* $\mathcal{A}_1,\cdots,\mathcal{A}_m$ *be sets of size* $a_1,\ldots,a_m$ *respectively, such that each* A_i *for* $i \geq 2$ *is chosen independently uniformly at random from all subsets of* $\mathcal{U}$ *having* a_i *elements;* A_1 *may be chosen arbitrarily. If the random variable* X *denotes the number of elements of the intersection* $\mathcal{A}_1 \cap \cdots \cap \mathcal{A}_m$ *then we have* $\mathsf{E}(X) = (\prod_{i=1}^{m} a_i)/N^{m-1}$ *and* $\mathsf{Var}(X) \leq (\prod_{i=1}^{m} a_i)/N^{m-1}$.

[4] For some intuition about the bound obtained, consider e.g. $\kappa \approx n$ and $t :\approx \ell + 1$.

Distinguisher $\mathbf{D}(\mathbf{E}, \mathbf{S})$: where $\mathbf{S} \in \{\mathsf{Casc}^{\mathbf{E}}_{\ell,\bar{K}}, \mathbf{P}\}$

1: **choose arbitrary** $\mathcal{S}_0 \subseteq \{0,1\}^n$ **s.t.** $|\mathcal{S}_0| = 2t \cdot 2^{\frac{\ell-2}{\ell}n}$
2: **for** $i := 1$ **to** $\ell/2 - 1$ **do**
3: **choose uniformly at random** $\mathcal{S}_{2i} \subseteq \{0,1\}^n$ **s.t.** $|\mathcal{S}_{2i}| = 2^{\frac{\ell-2}{\ell}n}$
4: **for all** $x \in \mathcal{S}_0$ **do**
5: **query** $y(x) := \mathbf{S}(x, +)$
6: $\mathcal{S}_\ell := \{y(x) \mid x \in \mathcal{S}_0\}$
7: **for all** $x \in \mathcal{S}_0 \cup \mathcal{S}_2 \cup \cdots \cup \mathcal{S}_{\ell-2}$ **do**
8: **for all** $k \in \{0,1\}^\kappa$ **do**
9: **query** $e_k(x) := \mathbf{E}(k, x, +)$
10: **for all** $y \in \mathcal{S}_2 \cup \mathcal{S}_4 \cup \cdots \cup \mathcal{S}_\ell$ **do**
11: **for all** $k \in \{0,1\}^\kappa$ **do**
12: **query** $e_k^{-1}(y) := \mathbf{E}(k, y, -)$
13: **for all** $\bar{k} = (k_1, \ldots, k_\ell) \in (\{0,1\}^\kappa)^\ell$ **do**
14: **choose** $\mathcal{I} \subseteq \mathcal{S}_0$ **s.t.** $|\mathcal{I}| = t$ **and** $\forall x \in \mathcal{I}, \forall i \in \{1, \ldots, \ell\}$: $e_{k_i}(\cdots e_{k_1}(x))$ **is known from lines** 9 **and** 12
15: **if** $\mathcal{I}$ exists $\wedge\ \forall x \in \mathcal{I} : y(x) = e_{k_\ell}(\cdots e_{k_1}(x))$ **then**
16: **return** 1
17: **return** 0

Fig. 3. Distinguisher $\mathbf{D}$ for the proof of Theorem 1 for the case of ℓ being even

Proof (of Theorem 1). Assume ℓ is even, we give the description of the distinguisher $\mathbf{D}$ in Fig. 3. It first chooses an arbitrary set $\mathcal{S}_0 \subseteq \{0,1\}^n$ and independent random sets $\mathcal{S}_2, \mathcal{S}_4, \ldots, \mathcal{S}_{\ell-2} \subseteq \{0,1\}^n$ of the given sizes and issues $2t \cdot 2^{\frac{\ell-2}{\ell}n}$ queries to the construction (cascade or random permutation – let us denote it $\mathbf{S}$) to obtain $\mathcal{S}_\ell := \mathbf{S}(S_0)$. Each $\mathcal{S}_i$ will represent the subset of values $\{0,1\}^n$ that $\mathbf{D}$ "cares about" after i steps of the cascade. Then $\mathbf{D}$ issues $\ell \cdot 2^{\kappa + \frac{\ell-2}{\ell}n}$ block-cipher queries to obtain all the values

$$\mathbf{E}_k(\mathcal{S}_0), \mathbf{E}_k^{-1}(\mathcal{S}_2), \mathbf{E}_k(\mathcal{S}_2), \ldots, \mathbf{E}_k^{-1}(\mathcal{S}_{\ell-2}), \mathbf{E}_k(\mathcal{S}_{\ell-2}), \mathbf{E}_k^{-1}(\mathcal{S}_\ell)$$

with all possible keys $k \in \{0,1\}^\kappa$. These are all the queries $\mathbf{D}$ makes, it remains to justify that they are sufficient to expect that there is a constant number of values $x \in \{0,1\}^n$ that, in case the correct keys are guessed, can be traced through the whole cascade only with the information obtained above. Each such path then allows us to compare its endpoint with $\mathbf{S}(x)$ which will most probably only match if $\mathbf{S}$ is the cascade.

Let us analyze the probability that the set $\mathcal{I}$ is found on line 14 in the setting where $\mathbf{S} = \mathsf{Casc}^{\mathbf{E}}_{\ell,\bar{K}}$ and the examined key is the correct one, i.e., for $\bar{k}$ chosen on line 13 we have $\bar{k} = \bar{K}$. Consider the sets

$$\begin{aligned}
\mathcal{P}_0 &= \mathcal{S}_0 \\
\mathcal{P}_2 &= \mathbf{E}_{k_1}^{-1}(\mathbf{E}_{k_2}^{-1}(\mathcal{S}_2)) \\
&\vdots \\
\mathcal{P}_{\ell-2} &= \mathbf{E}_{k_1}^{-1}(\cdots \mathbf{E}_{k_{\ell-3}}^{-1}(\mathbf{E}_{k_{\ell-2}}^{-1}(\mathcal{S}_{\ell-2}))\cdots),
\end{aligned}$$

i.e., $\mathcal{P}_{2i}$ for $i \geq 1$ is the subset of the plaintext space $\{0,1\}^n$ that gets mapped to $\mathcal{S}_{2i}$ after applying the first $2i$ steps of the cascade with the correct keys. Since the sets $\mathcal{S}_{2i}$ for $i \geq 1$ were chosen independently at random, we can invoke Lemma 2 to obtain that for $\mathcal{P} = \bigcap_{i=0}^{\ell/2-1} \mathcal{P}_{2i}$ we have

$$\mathsf{E}(|\mathcal{P}|) = \frac{\prod_{i=0}^{\ell/2-1} |\mathcal{P}_{2i}|}{2^{n(\frac{\ell}{2}-1)}} = \frac{\prod_{i=0}^{\ell/2-1} |\mathcal{S}_{2i}|}{2^{n(\frac{\ell}{2}-1)}} = 2t$$

and similarly $\mathsf{Var}(|\mathcal{P}|) \leq 2t$. Using Chebyshev inequality, this gives us $\mathsf{P}(|\mathcal{P}| < t) \leq 2/t$. If this does not occur (i.e., if $|\mathcal{P}| \geq t$) then any t-element subset of $\mathcal{P}$ clearly satisfies all requirements imposed on the set $\mathcal{I}$ on lines 14 and 15 (note that any such subset can be chosen, we assume that $\mathbf{D}$ has a fixed way of doing so). Since the desired $\mathcal{I}$ exists, $\mathbf{D}$ will output 1 in this case. Overall, this gives us that $\mathbf{D}(\mathbf{E}, \mathsf{Casc}^{\mathbf{E}}_{\ell.\bar{K}})$ outputs 1 with probability at least $1 - 2/t$.

On the other hand, if $\mathbf{S} = \mathbf{P}$ then for each $\bar{k}$ the condition on line 15 can only be satisfied with probability at most $2^{-t(n-1)}$, hence by union bound $\mathbf{D}(\mathbf{E}, \mathbf{P})$ outputs 1 with probability at most $2^{\ell\kappa - t(n-1)}$, which concludes the proof for the case of even ℓ.

For odd ℓ we just start by choosing $\mathcal{S}_0, \mathcal{S}_1, \mathcal{S}_3, \ldots, \mathcal{S}_{\ell-2} \subseteq \{0,1\}^n$ with $|\mathcal{S}_0| = 2t \cdot 2^{\frac{\ell-1}{\ell+1}n}$ and each of the remaining sets having size $2^{\frac{\ell-1}{\ell+1}n}$. The rest of the attack and its analysis is analogous and therefore omitted. □

Interestingly, for $\ell = 2$ our attack corresponds to the well-known meet-in-the-middle attack against double encryption [12] and for $\ell = 3$ it corresponds to one of the attacks given in [23].

Note that there is a trade-off between the number of construction queries and block-cipher queries required for the attack presented in Theorem 1. The attack can be modified in a straightforward way to use a lower number $2tm$ of construction queries and $2^{\kappa+n-\frac{2\log m}{\ell-2}}$ block-cipher queries. Moreover, the construction queries can be chosen arbitrarily, making it a known-plaintext attack.

4 XOR-Cascades

We now turn to investigate the so-called XOR-cascades that, loosely speaking, consist of multiple encryption steps interleaved with key-whitening steps using the XOR operation.

This design paradigm still offers several degrees of freedom: the addition or omission of the key-whitening step at the beginning and at the end; as well as repetition or dependence of keys across the encryption and whitening steps. We resolve the first choice by including the first XOR operation and omitting the last one, see Fig. 4 and the formal definition below. In the choice of key-scheduling we consider the variant that derives all keys used in the encryption steps from a single one in a fixed deterministic way such that they are distinct. This is safe thanks to the properties of the ideal-cipher model that we are working in that postulates the independence of the permutations realized for each key by

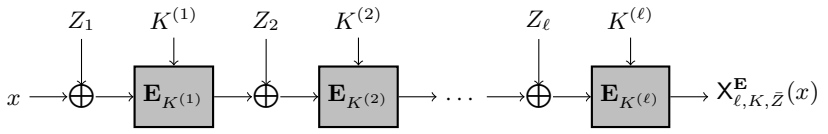


Fig. 4. The XOR-cascade construction realized by $\mathsf{X}^{\mathbf{E}}_{\ell,K,\bar{Z}}$

the block cipher; any practical instantiation would however require a form of security under related-key attack [9]. In order to avoid such an assumption, one could also consider independent keys for each of the encryption steps, arriving at the same security statement for a construction requiring more key material. Finally, we assume the whitening keys to be random and independent. A formal definition of the ℓ-XOR-cascade construction follows.

Let us fix a deterministic way to derive ℓ distinct κ-bit keys $(k^{(1)}, \ldots, k^{(\ell)})$ from a given κ-bit key k in such a way that each mapping $k \mapsto k^{(i)}$ is a bijection. For example, if we assume $\ell \leq \kappa$ then we can simply set $k^{(i)} := k \oplus 0^{i-1}10^{\kappa-i}$, i.e., $k^{(i)}$ will differ from k in the i-th bit. The definition extends naturally to random variables $K^{(1)}, \ldots, K^{(\ell)}$ derived from a uniformly random key K.

In the following discussion, let us model the XOR-cascade of length ℓ by a (deterministic stateless) construction $\mathsf{X}^{(\cdot)}_\ell \colon \{0,1\}^\kappa \times (\{0,1\}^n)^{\ell+1} \times \{+,-\} \to \{0,1\}^n$ which expects to access a subsystem $\mathbf{E} \colon \{0,1\}^\kappa \times \{0,1\}^n \times \{+,-\} \to \{0,1\}^n$ realizing a block cipher. The combined system $\mathsf{X}^{\mathbf{E}}_\ell$ then answers each forward query $(k, z_1, \ldots, z_\ell, x, +)$ by $\mathbf{E}_{k^{(\ell)}}(\cdots \mathbf{E}_{k^{(2)}}(\mathbf{E}_{k^{(1)}}(x \oplus z_1) \oplus z_2) \cdots \oplus z_\ell)$ and each backward query $(k, z_1, \ldots, z_\ell, y, -)$ by $\mathbf{E}^{-1}_{k^{(1)}}(\cdots \mathbf{E}^{-1}_{k^{(\ell-1)}}(\mathbf{E}^{-1}_{k^{(\ell)}}(y) \oplus z_\ell) \oplus z_{\ell-1} \cdots) \oplus z_1$. Again, we let $\mathsf{X}^{\mathbf{E}}_{\ell,K,\bar{Z}}$ be the system that first chooses uniformly random (secret) keys $(K, \bar{Z}) \in \{0,1\}^\kappa \times (\{0,1\}^n)^\ell$ where $\bar{Z} = (Z_1, \ldots, Z_\ell)$ and then gives access to the permutation $\mathsf{X}^{\mathbf{E}}_\ell(K, \bar{Z}, \cdot)$ in both directions (i.e., takes inputs from $\{0,1\}^n \times \{+,-\}$). The evaluation of a forward query by $\mathsf{X}^{\mathbf{E}}_{\ell,K,\bar{Z}}$ is depicted in Fig. 4.

Before presenting our results, we introduce the notion of *key-alternating ciphers*. This concept, studied for example in [15,11,29,20,8,21], is surprisingly close to the notion of XOR-cascades, however introduced with a very different motivation. It refers to a construction of a block cipher by alternating two types of steps: an XOR of a secret key and an application of a publicly known permutation (see Fig. 5 and the formal definition below). A prominent example of a block cipher having this structure is the current standard AES [4]. This approach to block-cipher construction is then typically studied in the random-permutation model where one assumes that the permutation steps consist of applications of uniformly random and independent, publicly accessible permutations. Below we model the key-alternating ciphers under this assumption. Note that in this setting it is natural to consider constructions that both start and end with the XOR operation.

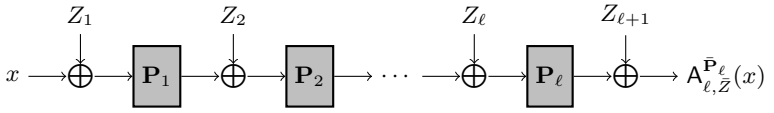


Fig. 5. The key-alternating cipher realized by $\mathsf{A}^{\bar{\mathbf{P}}_\ell}_{\ell,\bar{Z}}$

In the following, let us denote by $\mathsf{A}^{(\cdot)}_{\ell,\bar{Z}}$ the key-alternating cipher as it is formalized in the random permutation model (e.g. in [15,11,29,20]). More precisely, let $\mathsf{A}^{(\cdot)}_{\ell}\colon (\{0,1\}^n)^{\ell+2} \times \{+,-\} \to \{0,1\}^n$ be a construction which expects to access a subsystem $\hat{\mathbf{P}}_\ell$ giving bidirectional access to ℓ arbitrary permutations (denoted $P_1,\ldots,P_\ell$), using some fixed addressing mechanism for the queries. The combined system $\mathsf{A}^{\hat{\mathbf{P}}_\ell}_{\ell}$ then answers each forward query $(z_1,\ldots,z_{\ell+1},x,+)$ by the value $P_\ell\left(\cdots P_2\left(P_1\left(x\oplus z_1\right)\oplus z_2\right)\cdots\oplus z_\ell\right)\oplus z_{\ell+1}$ and each backward query $(z_1,\ldots,z_{\ell+1},y,-)$ by $P_1^{-1}(\cdots P_{\ell-1}^{-1}(P_\ell^{-1}(y\oplus z_{\ell+1})\oplus z_\ell)\oplus z_{\ell-1}\cdots)\oplus z_1$. Again, we let $\mathsf{A}^{\hat{\mathbf{P}}_\ell}_{\ell,\bar{Z}}$ be the system that first chooses uniformly random (secret) keys $\bar{Z}\in(\{0,1\}^n)^{\ell+1}$ where $\bar{Z}=(Z_1,\ldots,Z_{\ell+1})$ and then gives access to the permutation $\mathsf{A}^{\hat{\mathbf{P}}_\ell}_{\ell}(\bar{Z},\cdot)$ in both directions (taking inputs from $\{0,1\}^n\times\{+,-\}$). Finally, let $\bar{\mathbf{P}}_i$ denote a system that provides bidirectional access to i independent uniformly random permutations. The evaluation of a forward query by $\mathsf{A}^{\bar{\mathbf{P}}_\ell}_{\ell,\bar{Z}}$ is depicted in Fig. 5 and some known results on the security of key-alternating ciphers in the random-permutation model are summarized using our formalism in Appendix A.

We are now ready to present the reduction of the security of XOR-cascades in the ideal-cipher model to the problem of the security of one step shorter key-alternating ciphers in the random-permutation model. This reduction allows one to analyze the problem in a simpler setting without considering the block-cipher keys, as well as invoke existing results on key-alternating ciphers. The proof modularizes the approach used in [18] to analyze the security of XOR-cascade of length 2 and generalizes it to arbitrary lengths.

Theorem 2. *For $\ell\geq 2$, for the constructions $\mathsf{X}^{(\cdot)}_{\ell,K,\bar{Z}}$ and $\mathsf{A}^{(\cdot)}_{\ell-1,\bar{Z}}$ defined as above, and for every distinguisher $\mathbf{D}$ making q queries to $\mathbf{E}$,*

$$\Delta^{\mathbf{D}}\left(\left(\mathbf{E},\mathsf{X}^{\mathbf{E}}_{\ell,K,\bar{Z}}\right),(\mathbf{E},\mathbf{P})\right)\leq\min_h\left\{\frac{\ell q}{h2^\kappa}+\Delta_h\left(\left(\bar{\mathbf{P}}_{\ell-1},\mathsf{A}^{\bar{\mathbf{P}}_{\ell-1}}_{\ell-1,\bar{Z}}\right),\bar{\mathbf{P}}_\ell\right)\right\}\ .$$

In particular, $\mathbf{D}$ can make arbitrarily many queries to either of $\mathsf{X}^{\mathbf{E}}_{\ell,K,\bar{Z}}$ and $\mathbf{P}$.

Proof. In accordance with [10,16,18] we first reduce the original distinguishing problem to a simpler one, involving only block-cipher queries. Overall, the system $(\mathbf{E},\mathsf{X}^{\mathbf{E}}_{\ell,K,\bar{Z}})$ provides an interface to query $2^\kappa+1$ (dependent) permutations:

2^κ of them correspond to the block cipher $\mathbf{E}$ being used under all possible keys and the last permutation is provided by $\mathsf{X}^{\mathbf{E}}_{\ell,K,\bar{Z}}$, where the values K and $\bar{Z}$ are chosen at the beginning by the construction $\mathsf{X}_{\ell,K,\bar{Z}}$. (All these permutations can be queried both in forward and backward direction.) Since the last permutation is also uniformly distributed and $\mathrm{Perm}(n)$ forms a group under composition, the joint distribution of these permutations does not change if we first choose the last permutation uniformly at random, i.e., we replace it by $\mathbf{P}$, then pick random K and $\bar{Z}$ and finally choose the permutations of the block cipher independently and uniformly for all keys except $K^{(\ell)}$, for which we choose the permutation $x \mapsto \mathbf{P}(\mathbf{E}^{-1}_{K^{(1)}}(\cdots \mathbf{E}^{-1}_{K^{(\ell-2)}}(\mathbf{E}^{-1}_{K^{(\ell-1)}}(x \oplus Z_\ell) \oplus Z_{\ell-1})\cdots) \oplus Z_1)$. To formalize this transition, let $\mathsf{G}^{(\cdot)}$ be a construction that expects a single permutation as its subsystem (let us denote it P) and itself provides an interface to a block cipher (let us denote it G). Any query to G is answered in the following way: in advance, G chooses random keys $(K, \bar{Z})$ and then generates random independent permutations for G used with any key except $K^{(\ell)}$. For $K^{(\ell)}$, G instead realizes the permutation $x \mapsto P(G^{-1}_{K^{(1)}}(\cdots G^{-1}_{K^{(\ell-2)}}(G^{-1}_{K^{(\ell-1)}}(x \oplus Z_\ell) \oplus Z_{\ell-1})\cdots) \oplus Z_1)$, querying P for any necessary values. By the above argument we then have $\left(\mathbf{E}, \mathsf{X}^{\mathbf{E}}_{\ell,K,\bar{Z}}\right) = (\mathsf{G}^{\mathbf{P}}, \mathbf{P})$ and hence also

$$\Delta_q\left(\left(\mathbf{E}, \mathsf{X}^{\mathbf{E}}_{\ell,K,\bar{Z}}\right), (\mathbf{E}, \mathbf{P})\right) = \Delta_q\left(\left(\mathsf{G}^{\mathbf{P}}, \mathbf{P}\right), (\mathbf{E}, \mathbf{P})\right).$$

Now we can apply claim (iii) in Lemma 1 to obtain $\Delta_q\left(\left(\mathsf{G}^{\mathbf{P}}, \mathbf{P}\right), (\mathbf{E}, \mathbf{P})\right) \leq \Delta_q\left(\left(\mathsf{G}^S, S\right), (\mathbf{E}, S)\right)$ where S denotes the fixed permutation whose existence is guaranteed by this claim. Since S is fixed and hence can be seen as known to the distinguisher, it makes no sense to query it and therefore we only have to bound $\Delta_q\left(\mathsf{G}^S, \mathbf{E}\right)$ for an arbitrary permutation S. To simplify the notation, we shall denote the system G^S by $\mathbf{G}$.

Let us call a (forward or backward) query to $\mathbf{G}$ *relevant* if it involves any of the keys $K^{(1)}, \ldots, K^{(\ell)}$. Similarly, we can see the system $\mathbf{E}$ as also choosing some random key K (and hence also all $K^{(i)}$) that does not affect its behavior, it just serves to define relevant queries for $\mathbf{E}$ in an analogous way. We now define monotone conditions $\mathcal{A}^h$ and $\mathcal{B}^h$ on systems $\mathbf{E}$ and $\mathbf{G}$ respectively, such that each of these conditions remains satisfied as long as at most h of the queries asked so far were relevant. In $\mathbf{E}$ the probability of violating this condition can be upper-bounded easily since the keys $K^{(i)}$ do not affect the system's behavior and hence it suffices to consider non-adaptive strategies. The expected number of relevant queries among any given q queries asked by the distinguisher is $\ell q \cdot 2^{-\kappa}$ and from Markov inequality we obtain $\nu(\mathbf{E}, \overline{\mathcal{A}}^h_q) \leq \ell q/h2^\kappa$. Hence by claim (i) of Lemma 1 we have

$$\Delta_q(\mathbf{G}, \mathbf{E}) \leq \Delta_q(\mathbf{G}^\perp, \mathbf{E}^\perp) + \nu(\mathbf{E}, \overline{\mathcal{A}}^h_q) \leq \Delta_q(\mathbf{G}^\perp, \mathbf{E}^\perp) + \ell q/h2^\kappa$$

where $\mathbf{E}^\perp$ and $\mathbf{G}^\perp$ denote the systems $\mathbf{E}$ and $\mathbf{G}$ blocked by $\mathcal{A}^h$ and $\mathcal{B}^h$, respectively.

In order to upper-bound the term $\Delta_q(\mathbf{G}^\perp, \mathbf{E}^\perp)$, we notice that the systems $\mathbf{G}^\perp$ and $\mathbf{E}^\perp$ only differ in a small part. Moreover, this part corresponds to the systems considered in the security definition of key-alternating ciphers in the random-permutation model. More precisely, $\mathbf{G}^\perp = \mathsf{C}^{\mathbf{S}}$ and $\mathbf{E}^\perp = \mathsf{C}^{\mathbf{T}}$ where:

- $\mathbf{S}$ denotes a system that chooses ℓ random keys $\bar{Z} \in (\{0,1\}^n)^\ell$ and then provides access (by means of both forward and backward queries) to ℓ randomly chosen permutations $\pi_1, \ldots, \pi_\ell \in \mathrm{Perm}(n)$ such that they satisfy the equation

$$\pi_\ell^{-1}(\pi_{\ell-1}(\cdots \pi_2(\pi_1(\cdot \oplus Z_1) \oplus Z_2) \oplus Z_3 \cdots) \oplus Z_\ell) = id;$$

  i.e., $\pi_1, \ldots, \pi_{\ell-1}$ are chosen independently at random and π_ℓ is set to

$$x \mapsto \pi_{\ell-1}(\cdots \pi_2(\pi_1(x \oplus Z_1) \oplus Z_2) \oplus Z_3 \cdots) \oplus Z_\ell.$$

  Note that this corresponds to the system $\left(\bar{\mathbf{P}}_{\ell-1}, \mathsf{A}_{\ell-1,\bar{Z}}^{\bar{\mathbf{P}}_{\ell-1}}\right)$.
- $\mathbf{T}$ denotes a system that provides access (by means of both forward and backward queries) to ℓ uniformly random permutations $\pi_1, \ldots, \pi_\ell \in \mathrm{Perm}(n)$ that are independent. This in turn corresponds to the system $\bar{\mathbf{P}}_\ell$.
- $\mathsf{C}^{(\cdot)}$ denotes a randomized construction expecting a subsystem providing bidirectional access to ℓ permutations $\pi_1, \ldots, \pi_\ell$. The construction $\mathsf{C}^{(\cdot)}$ itself then provides access to a block cipher (let us denote it C) as follows: it first chooses a uniformly random key K and then sets $C_{K^{(i)}} := \pi_i$ for all $i \in \{1, \ldots, \ell-1\}$ and $C_{K^{(\ell)}}(\cdot) := S(\pi_\ell^{-1}(\cdot))$. ($\mathsf{C}$ only queries its subsystem once it is necessary in order to answer a relevant query to C). The permutations for all other keys are chosen independently at random and maintained by C. Moreover, C only allows h relevant queries, after that it returns $\perp$.

It is now straightforward to verify that we indeed have $\mathbf{G}^\perp = \mathsf{C}^{\mathbf{S}}$ and $\mathbf{E}^\perp = \mathsf{C}^{\mathbf{T}}$. Since $\mathsf{C}^{(\cdot)}$ issues at most h queries to its subsystem, we can invoke Lemma 1(ii) to obtain

$$\Delta_q(\mathbf{G}^\perp, \mathbf{E}^\perp) \leq \Delta_h(\mathbf{S}, \mathbf{T}) = \Delta_h\left(\left(\bar{\mathbf{P}}_{\ell-1}, \mathsf{A}_{\ell-1,\bar{Z}}^{\bar{\mathbf{P}}_{\ell-1}}\right), \bar{\mathbf{P}}_\ell\right).$$

The whole argument holds for any parameter h, hence we can minimize over it to conclude the proof of the theorem. □

Combining our Theorem 2 with the known results on the security of key-alternating ciphers in the random permutation model [11,29,20] given in Appendix A we obtain the following corollary.

Corollary 1. *Let $\mathsf{X}_{\ell,K,\bar{Z}}^{(\cdot)}$ denote the ℓ-XOR-cascade construction as above. Then we have:*

1. *3-XOR-cascade is secure up to roughly $2^{\kappa+\frac{2}{3}n}$ queries; more precisely, for $n \geq 20$ we have*

$$\Delta_q\left(\left(\mathbf{E}, \mathsf{X}_{3,K,\bar{Z}}^{\mathbf{E}}\right), (\mathbf{E}, \mathbf{P})\right) \leq 3 \cdot \left(\frac{q}{2^{\kappa+\frac{2}{3}n}}\right)^{\frac{1}{2}} + 9 \cdot \left(\frac{q}{2^{\kappa+\frac{2}{3}n}}\right)^{\frac{3}{2}} + 3 \cdot \frac{q}{2^{\kappa+\frac{2}{3}n}}.$$

2. *ℓ-XOR-cascade is secure up to roughly $2^{\kappa+\frac{3}{4}n}$ queries for $\ell \geq 4$; more precisely, for $n \geq 27$ we have*

$$\Delta_q\left(\left(\mathbf{E}, \mathsf{X}^{\mathbf{E}}_{\ell,K,\bar{Z}}\right), (\mathbf{E}, \mathbf{P})\right) \leq \ell \cdot \left(\frac{q}{2^{\kappa+\frac{3}{4}n}}\right)^{\frac{1}{2}} + 9 \cdot \frac{q}{2^{\kappa+\frac{3}{4}n}} + 4 \cdot \left(\frac{q}{2^{\kappa+\frac{3}{4}n}}\right)^{\frac{3}{2}} .$$

3. *ℓ-XOR-cascade is secure up to roughly $2^{\kappa+\frac{\ell-1}{\ell+1}n}$ queries for odd ℓ; more precisely, we have*

$$\Delta_q\left(\left(\mathbf{E}, \mathsf{X}^{\mathbf{E}}_{\ell,K,\bar{Z}}\right), (\mathbf{E}, \mathbf{P})\right) \leq (\ell+1) \cdot \left(\frac{q}{2^{\kappa+\frac{\ell-1}{\ell+1}n}}\right)^{\frac{1}{2}} + 2^{3+\frac{\ell-1}{4}} \cdot \left(\frac{q}{2^{\kappa+\frac{\ell-1}{\ell+1}n}}\right)^{\frac{\ell+1}{8}} .$$

For even ℓ one can prove the same security as for one step shorter odd-length XOR-cascade.

Proof (sketch). We combine the statement of Theorem 2 with the bounds on the security of the key-alternating cipher listed in Theorem 4, choosing the value h to be $q^{\frac{1}{2}}2^{\frac{n}{3}-\frac{\kappa}{2}}$, $q^{\frac{1}{2}}2^{\frac{3n}{8}-\frac{\kappa}{2}}$ and $q^{\frac{1}{2}}2^{\frac{(\ell-1)n}{2(\ell+1)}-\frac{\kappa}{2}}$ in the three cases above, respectively. The statements for constructions with more rounds follow from the fact that

$$\Delta_h\left(\left(\bar{\mathbf{P}}_\ell, \mathsf{A}^{\bar{\mathbf{P}}_\ell}_{\ell,\bar{Z}}\right), \bar{\mathbf{P}}_{\ell+1}\right) \leq \Delta_h\left(\left(\bar{\mathbf{P}}_{\ell-1}, \mathsf{A}^{\bar{\mathbf{P}}_{\ell-1}}_{\ell-1,\bar{Z}}\right), \bar{\mathbf{P}}_\ell\right)$$

which can be shown by a straightforward reduction. □

5 Sequential Constructions

To obtain an upper bound on the security achievable by the ℓ-XOR-cascade construction, in this section we consider keylength-extending constructions having a particular natural form which we call *sequential*.

A construction $\mathsf{C}\colon \{0,1\}^{\kappa'} \times \{0,1\}^n \times \{+,-\} \to \{0,1\}^n$ is sequential if, given an underlying block cipher $\mathbf{E}$, the mapping it realizes can be written as

$$\mathsf{C}^{\mathbf{E}}(k', x, +) = Q_{\ell,k'}\left(\mathbf{E}_{k_\ell}\left(Q_{\ell-1,k'}\left(\cdots \mathbf{E}_{k_2}\left(Q_{1,k'}\left(\mathbf{E}_{k_1}\left(Q_{0,k'}(x)\right)\right)\right)\cdots\right)\right)\right)$$

where all keys k_i are determined by k' and $Q_{i,k'}$ is a fixed permutation for all $(i,k') \in \{0,\ldots,\ell\} \times \{0,1\}^{\kappa'}$. Again, we let $\mathsf{C}^{\mathbf{E}}_{K'}$ be the system that first chooses a uniformly random (secret) key $K' \in \{0,1\}^{\kappa'}$ and then gives access to the permutation $\mathsf{C}^{\mathbf{E}}(K', \cdot)$ in both directions (i.e., takes inputs from the set $\{0,1\}^n \times \{+,-\}$).

The attack on a class of so-called injective 2-query constructions given in [18] can be generalized to sequential ℓ-query constructions for arbitrary ℓ, resulting in the statement below. Its proof is given in the full version of this paper. Note that this attack can also be seen as a lifting of an attack presented in [11] into the ideal block-cipher setting.

Theorem 3. *Let* $\mathsf{C}^{(\cdot)}\colon \{0,1\}^{\kappa'} \times \{0,1\}^n \times \{+,-\} \to \{0,1\}^n$ *be a sequential* ℓ*-query construction. For any parameter* $0 < t < 2^{n/\ell-1}$ *there exists a distinguisher* $\mathbf{D}$ *such that*

$$\Delta^{\mathbf{D}}((\mathbf{E}, \mathsf{C}^{\mathbf{E}}_{K'}), (\mathbf{E}, \mathbf{P})) \geq 1 - 2/t - 2^{\kappa'-t(n-1)},$$

where $\mathbf{D}$ *makes at most* $(2t+\ell)\cdot 2^{\kappa+\frac{\ell-1}{\ell}n}$ *block-cipher queries as well as* 2^n *forward construction queries.*

Again, a trade-off between the number of construction queries and block-cipher queries is possible: an analogous attack can be mounted with a lower number m of construction queries and at most $(2t+\ell)\cdot 2^{\kappa+n-\frac{\log m}{\ell}}$ block-cipher queries. Also here the construction queries can be arbitrary, resulting in a known-plaintext attack.

Acknowledgements. I would like to thank Stefano Tessaro for useful discussions and helpful comments to earlier versions of this work and also the anonymous reviewers for their feedback.

References

1. Data encryption standard. In: FIPS PUB 46. Federal Information Processing Standards Publication (1977)
2. ANSI X9.52: Triple Data Encryption Algorithm Modes of Operation (1998)
3. FIPS PUB 46-3: Data Encryption Standard (DES). National Institute of Standards and Technology (1999)
4. Advanced encryption standard. In: FIPS PUB 197. Federal Information Processing Standards Publication (2001)
5. Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher. National Institute of Standards and Technology. Special Publication 800-67 (2004)
6. EMV Integrated Circuit Card Specification for Payment Systems, Book 2: Security and Key Management, v.4.2 (June 2008)
7. Aiello, W., Bellare, M., Di Crescenzo, G., Venkatesan, R.: Security amplification by composition: The case of doubly-iterated, ideal ciphers. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 390–407. Springer, Heidelberg (1998)
8. Andreeva, E., Bogdanov, A., Dodis, Y., Mennink, B., Steinberger, J.P.: On the indifferentiability of key-alternating ciphers. Cryptology ePrint Archive, Report 2013/061 (2013), `http://eprint.iacr.org/`
9. Bellare, M., Kohno, T.: A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 491–506. Springer, Heidelberg (2003)
10. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer, Heidelberg (2006), Full version at `http://eprint.iacr.org/2004/331`

11. Bogdanov, A., Knudsen, L.R., Leander, G., Standaert, F.-X., Steinberger, J., Tischhauser, E.: Key-alternating ciphers in a provable setting: encryption using a small number of public permutations. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 45–62. Springer, Heidelberg (2012)
12. Diffie, W., Hellman, M.E.: Exhaustive Cryptanalysis of the NBS Data Encryption Standard. Computer 10(6), 74–84 (1977)
13. Dinur, I., Dunkelman, O., Keller, N., Shamir, A.: Efficient dissection of composite problems, with applications to cryptanalysis, knapsacks, and combinatorial search problems. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 719–740. Springer, Heidelberg (2012)
14. Even, S., Goldreich, O.: On the power of cascade ciphers. ACM Trans. Comput. Syst. 3(2), 108–116 (1985)
15. Even, S., Mansour, Y.: A construction of a cipher from a single pseudorandom permutation. Journal of Cryptology, 151–161 (1991)
16. Gaži, P., Maurer, U.: Cascade encryption revisited. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 37–51. Springer, Heidelberg (2009)
17. Gaži, P., Maurer, U.: Free-start distinguishing: Combining two types of indistinguishability amplification. In: Kurosawa, K. (ed.) ICITS 2010. LNCS, vol. 5973, pp. 28–44. Springer, Heidelberg (2010)
18. Gaži, P., Tessaro, S.: Efficient and optimally secure key-length extension for block ciphers via randomized cascading. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 63–80. Springer, Heidelberg (2012)
19. Kilian, J., Rogaway, P.: How to Protect DES Against Exhaustive Key Search (an Analysis of DESX). Journal of Cryptology 14, 17–35 (2001)
20. Lampe, R., Patarin, J., Seurin, Y.: An Asymptotically Tight Security Analysis of the Iterated Even-Mansour Cipher. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 278–295. Springer, Heidelberg (2012)
21. Lampe, R., Seurin, Y.: How to construct an ideal cipher from a small set of public permutations. Cryptology ePrint Archive, Report (2013), `http://eprint.iacr.org/`
22. Lee, J.: Towards Key-Length Extension with Optimal Security: Cascade Encryption and Xor-cascade Encryption. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 405–425. Springer, Heidelberg (2013)
23. Lucks, S.: Attacking triple encryption. In: Vaudenay, S. (ed.) FSE 1998. LNCS, vol. 1372, pp. 239–253. Springer, Heidelberg (1998)
24. Maurer, U.M.: Indistinguishability of Random Systems. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 110–132. Springer, Heidelberg (2002)
25. Maurer, U., Massey, J.L.: Cascade ciphers: The importance of being first. Journal of Cryptology 6(1), 55–61 (1993)
26. Maurer, U.M., Pietrzak, K.: Composition of random systems: When two weak make one strong. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 410–427. Springer, Heidelberg (2004)
27. Maurer, U.M., Pietrzak, K., Renner, R.S.: Indistinguishability amplification. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 130–149. Springer, Heidelberg (2007)
28. Maurer, U., Tessaro, S.: Computational indistinguishability amplification: Tight product theorems for system composition. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 355–373. Springer, Heidelberg (2009)

29. Steinberger, J.: Improved Security Bounds for Key-Alternating Ciphers via Hellinger Distance. Cryptology ePrint Archive, Report 2012/481 (2012), http://eprint.iacr.org/
30. Tessaro, S.: Security amplification for the cascade of arbitrarily weak PRPs: Tight bounds via the interactive Hardcore Lemma. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 37–54. Springer, Heidelberg (2011)
31. Vaudenay, S.: Decorrelation: a theory for block cipher security. Journal of Cryptology 16(4), 249–286 (2003)

A Security of Key-Alternating Ciphers

In this appendix we present several bounds recently proved for the security of key-alternating ciphers in the random-permutation model, recast into our formalism.

Theorem 4. *Let* $\mathsf{A}_{\ell,\bar{Z}}$ *denote the key-alternating cipher of length* ℓ *as described above.*

1. *[11] For any* $q < 2^n/100$ *we have*

$$\Delta_q((\bar{\mathbf{P}}_2, \mathsf{A}_{2,\bar{Z}}^{\bar{\mathbf{P}}_2}), \bar{\mathbf{P}}_3) \leq \frac{8.6q^3}{2^{2n}} + \frac{3q^2}{2^{\frac{4}{3}n}}.$$

2. *[29] For any* $\ell \geq 1$ *and* $q < 2^n/100$ *we have*

$$\Delta_q((\bar{\mathbf{P}}_\ell, \mathsf{A}_{\ell,\bar{Z}}^{\bar{\mathbf{P}}_\ell}), \bar{\mathbf{P}}_{\ell+1}) \leq 3\ell \cdot \frac{q^2}{2^{\frac{3}{2}n}} + (\ell+1) \cdot \frac{q^\ell}{2^{\frac{\ell^2}{\ell+1}n}}.$$

3. *[20] For any even* $\ell \geq 1$ *we have*

$$\Delta_q((\bar{\mathbf{P}}_\ell, \mathsf{A}_{\ell,\bar{Z}}^{\bar{\mathbf{P}}_\ell}), \bar{\mathbf{P}}_{\ell+1}) \leq 2^{\frac{\ell}{4}+3} \cdot \left(\frac{q^{\ell+2}}{2^{\ell n}}\right)^{\frac{1}{4}}.$$

Digital Signatures with Minimal Overhead from Indifferentiable Random Invertible Functions

Eike Kiltz[1,⋆], Krzysztof Pietrzak[2,⋆⋆], and Mario Szegedy[3]

[1] Horst-Görtz Institute for IT Security, Ruhr-Universität Bochum, Germany
eike.kiltz@rub.de
[2] Institute of Science and Technology, Austria
pietrzak@ist.ac.at
[3] Rutgers University, USA
szegedy@dragon.rutgers.edu

Abstract. In a digital signature scheme with message recovery, rather than transmitting the message m and its signature σ, a single enhanced signature τ is transmitted. The verifier is able to recover m from τ and at the same time verify its authenticity. The two most important parameters of such a scheme are its security and overhead $|\tau| - |m|$. A simple argument shows that for any scheme with "n bits security" $|\tau| - |m| \geq n$, i.e., the overhead is lower bounded by the security parameter n. Currently, the best known constructions in the random oracle model are far from this lower bound requiring an overhead of $n + \log q_h$, where q_h is the number of queries to the random oracle. In this paper we give a construction which basically matches the n bit lower bound. We propose a simple digital signature scheme with $n + o(\log q_h)$ bits overhead, where q_h denotes the number of random oracle queries.

Our construction works in two steps. First, we propose a signature scheme with message recovery having optimal overhead in a new ideal model, the random invertible function model. Second, we show that a four-round Feistel network with random oracles as round functions is tightly "public-indifferentiable" from a random invertible function. At the core of our indifferentiability proof is an almost tight upper bound for the expected number of edges of the densest "small" subgraph of a random Cayley graph, which may be of independent interest.

Keywords: digital signatures, indifferentiability, Feistel, Additive combinatorics, Cayley graph.

1 Introduction

When transmitting a message m over an unauthenticated public channel, one usually appends a string σ to the message that can be used to verify (relative

⋆ Funded by a Sofja Kovalevskaja Award of the Alexander von Humboldt Foundation and the German Federal Ministry for Education and Research.

⋆⋆ Supported by the European Research Council under the European Unions Seventh Framework Programme (FP7/2007-2013) / ERC Starting Grant (259668-PSPC).

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part I, LNCS 8042, pp. 571–588, 2013.

to a public key) the authenticity of the message. This string σ is called a *digital signature of* m. More generally, one transforms the message m into an *enhanced signature* τ such that (i) the original message m can be recovered from τ; (ii) the authenticity of m can be verified from τ. This is called a digital signature scheme with message recovery (MR) and is used to save on bandwidth, i.e., to minimize the *signature overhead* informally defined as $\mathsf{O} = |\tau| - |m|$ (signature length minus message length). Standard bodies for signature schemes (e.g. ISO/IEC 9796 and IEEE P1363a) contain several schemes with MR. In this paper we ask the natural question: *what is the minimal overhead required to achieve a desired security level?*

1.1 Bounds on the Overhead

A Trivial Lower Bound for Every Scheme. Following [3], we say that a signature scheme has "n-bit security" if all adversaries A attacking the scheme have success ratio $\mathsf{SR}(\mathsf{A})$ at most 2^{-n}, where $\mathsf{SR}(\mathsf{A}) := \mathsf{success}(\mathsf{A})/\mathsf{time}(\mathsf{A})$. A natural lower bound for the overhead of a signature scheme (with or without message recovery) for n-bit security is $\mathsf{O} \geq n$ bits. This is since for a signature scheme with O bits of overhead any random bit string τ constitutes a valid enhanced signature with probability $2^{-\mathsf{O}}$. Hence an adversary A guessing a single random authenticated message τ has success ratio $\mathsf{SR}(\mathsf{A}) = 2^{-\mathsf{O}}$ which implies $\mathsf{O} \geq n$.

Overhead of Schemes without MR. In standard digital signature schemes (without message recovery) such as RSA full domain hash [5], the probabilistic signature scheme PSS [5], or (pairing-based) BLS signatures [6] the overhead equals the size of a signature. Since classical signatures contain (at least) one group element (e.g., $\mathbb{Z}_N^*$ or an elliptic curve group) whose representation requires at least $2n$ bits (for n bits security, due to generic square-root attacks) we cannot hope to obtain an overhead smaller than $2n$ bits. The above lower bounds do not apply for schemes without such a group structure, in particular schemes based on lattices or codes, but for other reasons these schemes tend to have a very large overhead and/or prohibitively large public parameters.

Overhead of Schemes with MR in the RO model. Computing the overhead for a given signature scheme turns out to be a bit subtle and depends on the security reduction. We exemplify such a calculation for the RSA-based probabilistic signature scheme with message recovery PSS-MR$[n_0, n_1]$ [5], which can be seen as a two-round Feistel construction. PSS-MR$[n_0, n_1]$ has an overhead of $n_0 + n_1$ bits, where parameter n_0 controls the randomness and n_1 the amount of added redundancy used during signing. The minimal size of n_0 and n_1 providing a given security level can be computed from the security reduction. The security reduction from [5] in the random oracle model [4] transforms an adversary against PSS-MR$[n_0, n_1]$ making q_s (online) signing and q_h (offline) hash queries with success probability $\varepsilon_{\mathsf{PSS\text{-}MR}}$ into an adversary against RSA with success probability $\varepsilon_{\mathsf{RSA}}$ such that $\varepsilon_{\mathsf{PSS\text{-}MR}} = \varepsilon_{\mathsf{RSA}} + \varepsilon_{sim}$, where $\varepsilon_{sim} = (q_s + q_h)^2(2^{-n_0} + 2^{-n_1})$.

An easy computation shows that this implies $\mathsf{O}_{\mathsf{PSS\text{-}MR}} = n_0 + n_1 \geq 2n + 2\log_2(q_h)$ bits of overhead for n bits security.[1] An improved security reduction by Coron gives $\mathsf{O}_{\mathsf{PSS\text{-}MR}} \geq 2n + \log_2(q_h) + \log_2(q_s)$. Recently, an alternative security reduction for PSS-MR was proposed in [15] demonstrating a tight security reduction for $\mathsf{PSS\text{-}MR}[n_0 = 0, n_1]$ with zero-padding from the (stronger) phi-hiding assumption [7]. However, the required overhead is still $\mathsf{O}_{\mathsf{PSS\text{-}MR}} = n + \log_2(q_h)$ bits, stemming from an additive term $\varepsilon_{sim} = q_h^2/2^{n_1}$ in the security reduction.

The Random Invertible Permutation Model. Besides the popular random-oracle model, signature schemes have also been analyzed in other idealized models. In particular, [16,8] propose a digital signature scheme with message recovery, together with optimal security reduction in the ideal *random invertible permutation model*. Unfortunately, unlike for random oracles, there is no standard cryptographic object which could be used to directly instantiate random invertible permutations over a large domain.[2] In order to get a construction in the random oracle model, one can replace the random invertible permutation $\mathcal{P}$ with some construction $\mathsf{C}^{\mathcal{H}}$ (based on a random oracle $\mathcal{H}$) that is *indifferentiable* [19,10] from $\mathcal{P}$. In the context of signature schemes, already a weaker notion called "public-indifferentiability" [23,11,18] is sufficient. In [18] it is proven that a six-round Feistel network with random round functions is public-indifferentiable from a random invertible permutation. (For full indifferentiability more rounds are needed [14].) Unfortunately, the reduction from [18] is not tight in the oracle query complexity (i.e., the number of queries made by the simulator is quadratic in the number of the queries made by the distinguisher), and as a consequence the required overhead is $\log(q_h)$ bits larger than in the ideal permutation model.

Table 1 summarizes the signature overhead and gives concrete parameters for a typical security parameter of $n = 80$ bits and using 1024/2048-bit RSA. (Parameters for $n \in \{128, 192, 256\}$ can be computed accordingly.) We remark that the table is only valid for sufficiently large messages, i.e., if $|M| \geq 1024 - \mathsf{O}$. For smaller messages standard signatures such as BLS naturally outperform any RSA-based signature scheme with MR.

1.2 Our Contribution

Our main contribution is to revisit and affirmatively answer the question whether there exist signature schemes with minimal overhead in the random oracle model. In a first step we show that such a scheme exists in a new ideal model which we call *random invertible function* model, provided that the ideal functions' image

[1] For n-bit security of $\mathsf{PSS\text{-}MR}[n_0, n_1]$ we require $\mathsf{SR}(\mathsf{A}) \leq 2^{-n+1}$ which is implied by $\varepsilon_{\mathsf{RSA}}/\mathsf{time}(\mathsf{A}) \leq 2^{-n}$ and $\varepsilon_{sim}/\mathsf{time}(\mathsf{A}) \leq 2^{-n}$. With $\mathsf{time}(\mathsf{A}) \geq q_s + q_h$ we obtain $n_0 \geq n + \log_2(q_h)$ and $n_1 \geq n + \log_2(q_h)$ and consequently the overhead is $\mathsf{O} = n_0 + n_1 \geq 2n + 2\log_2(q_h)$.

[2] For fixed small domain, one might use a block-cipher with a fixed key. Though, the heuristic to replace a random permutation with a block-cipher like AES with fixed known keys is not as well analyzed as replacing a random oracle with a strong cryptographic hash function.

Table 1. Overhead of RSA-based signature schemes with message recovery in the random oracle model for n bits security assuming the adversary makes at most q_h hash and q_s signing queries. The table shows the overhead required for $n = 80$ (and only the trivial upper bound $q_h \leq 2^{80}$) and when we additionally assume that the number of random-oracle/signature queries are upper bounded by $q_h \leq 2^{60}$ and $q_s \leq 2^{40}$, respectively. As the $o(\log q_h)$ term in our bound depends on the domain, we give the bounds for 1024 and 2048 bits RSA.

Type	Required overhead O for n bits security		Security
	asymptotic	$n = 80$ $q_h \leq 2^{60}, q_s \leq 2^{40}$	reduction
2-round Feistel	$2n + 2(\log q_h)$	320	280	Bellare-Rogaway [5]
2-round Feistel	$2n + \log(q_h) + \log(q_s)$	320	240	Coron [9]
2-round Feistel	$n + \log(q_h)$	160	140	Kakvi-Kiltz [15]
6-round Feistel	$n + \log(q_h)$	160	140	[16,8]+[18]
4-round Feistel	$n + o(\log q_h)$	97	93	this work (1024-bit RSA)
4-round Feistel	$n + o(\log q_h)$	92	90	this work (2048-bit RSA)

is sufficiently sparse. Next, we show that a Feistel network with four rounds and random oracles as round functions is public-indifferentiable from a random invertible function *with an almost tight reduction*. Combining the two steps, we obtain a new signature scheme with message recovery with almost minimal overhead in the random oracle model.

Signature scheme with MR from random invertible functions. Given a trapdoor permutation $\mathsf{TDP} = (\mathsf{f}, \mathsf{f}^{-1})$ over $\{0,1\}^k$ and an injective function $\mathcal{F} : \{0,1\}^m \to \{0,1\}^k$ $(k > m)$ that can be queried in both directions, we can define a signature scheme with message recovery $\mathsf{SIG\text{-}MR}^{\mathcal{F}}$ as follows. The enhanced signature τ on a message m is defined as $\tau = \mathsf{f}^{-1}(\mathcal{F}(m))$. Signature recovery first evaluates the trapdoor permutation on τ and checks if the result has a valid pre-image or not, i.e., $\{m, \perp\} = \mathcal{F}^{-1}(\mathsf{f}(\tau))$. If the result is not $\perp$, it returns message m. The overhead of $\mathsf{SIG\text{-}MR}^{\mathcal{F}}$ is $\mathsf{O} = k - m$ bits. It is a straightforward generalization of [16,15], to prove that the resulting signature scheme $\mathsf{SIG\text{-}MR}^{\mathcal{F}}$ is tightly secure (losing an additive factor $q_{\mathcal{F}}/2^{k-m}$, where $q_{\mathcal{F}}$ is the number of queries to $\mathcal{F}$) if $\mathcal{F}$ is chosen at random. (The above scheme can only be proved secure assuming TDP is lossy [22]. Using a trick of [16] we can also prove a slightly modified scheme tightly secure assuming TDP is one-way.)

Instantiating Invertible Random Functions with Random Oracles. To instantiate the above scheme in the random oracle model, we must replace the random invertible function $\mathcal{F} : \{0,1\}^m \to \{0,1\}^k$ with a construction $\mathsf{C}^{\mathcal{H}}$ that is public-indifferentiable from $\mathcal{F}$.

It is easy to construct a random invertible function $\mathcal{F} : \{0,1\}^m \to \{0,1\}^k$ from a random invertible permutation $\mathcal{P} : \{0,1\}^k \to \{0,1\}^k$ (by setting $\mathcal{F}(x) = \mathcal{P}(x \| 0^{k-m})$) with a tight reduction. But as discussed above, we do not know how to instantiate $\mathcal{P}$ in the random oracle model without losing at least a quadratic factor in the oracle query complexity [18]. Furthermore, it is well known that a

five (or less) round Feistel network cannot be pub-indifferentiable from a random invertible permutation [18].

A formal definition of pub-indifferentiability is given in Definition 1. The important parameters are the error ε_{sim} and the number of queries q_{S} made by the simulator S, which are both functions in the number of queries q_{D} made by the distinguisher D. In order to get a reduction with optimal overhead, i.e., where the security (in bits) is not much smaller than the overhead $\mathsf{O} = k - m$, we need $q_{\mathsf{S}} \approx q_{\mathsf{D}}$ and $\varepsilon_{sim} \approx q_{\mathsf{D}}/2^{k-m}$.

Two Feistel rounds. As a simple warmup example we show that a two-round Feistel network (with random oracles as round functions) is pub-indifferentiable from $\mathcal{F}$ with

$$\varepsilon_{sim} = q_{\mathsf{D}}^2/2^{k-m} \qquad \text{and} \qquad q_{\mathsf{S}} = q_{\mathsf{D}}.$$

The resulting signature scheme (as explained above) requires an overhead of $\mathsf{O} = n + \log_2(q_h)$ to achieve n bits security. This essentially reproves the overhead of PSS-MR obtained in [15].

Four Feistel rounds. As the main technical result of this paper we give a construction $\mathsf{C}_{4F}^{\mathcal{H}}$ based on a four round Feistel network and prove it pub-indifferentiable from $\mathcal{F}$ with

$$\varepsilon_{sim} \leq q_{\mathsf{D}}^{1+o(1)}/2^{k-m} \qquad \text{and} \qquad q_{\mathsf{S}} = \tilde{O}(q_{\mathsf{D}}). \tag{1}$$

Hence the resulting signature scheme has an overhead of $\mathsf{O} = n + o(\log q_h)$ bits, cf. Table 1. The $o(1)$ term can be computed explicitly and for example leads to 97 bits overhead for $n = 80$ bits security if the domain of the TDP is at least 1024 bits (we get smaller overhead if the domain is larger or we put non-trivial bound on q_h). The $o(1)$ term goes to 0 as the ratio of the security we want to achieve, divided by the domain size of the TDP, decreases.

In the proof of (1), the variable $Q(\mu, q) = \max_{\mathcal{X},\mathcal{Z}} |\{(x, z) \mid x \in \mathcal{X}, z \in \mathcal{Z}, z - x \in \mathcal{B}\}|$ (where $\mathcal{B}, \mathcal{X}, \mathcal{Z}$ are q element subsets of $\mathbb{Z}_\mu$ and $\mathcal{B}$ is sampled uniformly at random) will play a central role. This variable has a natural interpretation in graph theoretical terms, it's the number of edges of the densest "small" subgraph of a random (bipartite) Cayley graph (here the Cayley graph has μ vertices on each side, is of degree q and the subgraph has q vertices on each side.) We prove by a compression argument (Corollary 1) an upper bound

$$\text{for each } 0 < a < 1/4 : Q(\mu, \mu^a) \leq \mu^{a+2a^2} \quad \text{(with probability extremely close to 1).} \tag{2}$$

We believe that this bound may be of independent interest. It complements a result of Alon et al. [2, Th. 4] which states that $Q(\mu, \mu^a) \approx \mu^{3a-1}$ for $2/3 < a \leq 1$, i.e. their bound applies to large subgraphs of size $\geq \mu^{2/3}$.

We show (Theorem 5) that the four round Feistel network $\mathsf{C}_{4F}^{\mathcal{H}}$ is pub-indifferentiable form a random invertible function with a simulator making $q_{\mathsf{S}} = \tilde{O}(q_{\mathsf{D}})$ queries and failing with probability $\varepsilon_{sim} = O(\mathsf{E}[Q(\mu, q_{\mathsf{D}})]/2^{k-m})$. Setting $q_{\mathsf{D}} = \mu^a$ in (2) this gives the claimed bound (1) on the pub-indifferentiability of $\mathsf{C}_{4F}^{\mathcal{H}}$.

We leave it is an interesting open problem whether our techniques can be used to prove better bounds for constructions of *permutations* from random oracles. As mentioned above, currently all such constructions suffer from a quadratic increase in the oracle query complexity. Another interesting question is, whether random invertible functions can be used to build chosen-ciphertext secure encryption with optimal overhead. Interestingly, the construction from [1] also uses a four round Feistel network, but the proven security suffers from a quadratic loss in running time.

2 Preliminaries

For $n \in \mathbb{N}$, we write 1^n for the string of n ones, and $[n]$ for $\{1, \ldots, n\}$. $|x|$ denotes the length of a bitstring x, while $|S|$ denotes the size of a set S. $s \leftarrow S$ denotes sampling an element s uniformly at random from the set S. For an algorithm A, we write $z \leftarrow \mathsf{A}(x, y, \ldots)$ to indicate that A is a (probabilistic) algorithm that outputs z on input $(x, y, \ldots)$. In the following we will introduce some basic cryptographic objects that (for simplicity) are defined over bit-strings (rather than arbitrary domains).

2.1 Ideal Primitives and Indifferentiability

Throughout, we use the letter $\mathcal{H}$ to denote a random oracle [4], $\mathcal{P}$ for a random invertible permutation and $\mathcal{F}$ for a random invertible function.

A random oracle $\mathcal{H} : \mathcal{D} \to \mathcal{R}$ with input domain $\mathcal{D} \subset \{0,1\}^*$ and range $\mathcal{R} \subset \{0,1\}^*$ is a function chosen uniformly at random from all functions $\mathcal{D} \to \mathcal{R}$. A random invertible function $\mathcal{F} : \mathcal{D} \to \mathcal{R}$ is a function chosen uniformly at random from all injective functions (i.e., all functions where $x \neq x' \Rightarrow \mathcal{F}(x) \neq \mathcal{F}(x')$). A random invertible permutation $\mathcal{P}$ is a random injective function where $\mathcal{D} \equiv \mathcal{R}$.

Unlike for $\mathcal{H}$, which can only be queried in forward direction, whenever we consider algorithms with oracle access to $\mathcal{F}$ (or $\mathcal{P}$), it is always understood that $\mathcal{F}$ can be queried also in inverse direction. Technically, we can think of $\mathcal{F}$ as being given by two oracles $\mathcal{F}$ and $\mathcal{F}^{-1}$, where $\mathcal{F}^{-1}(\mathcal{F}(x)) = x$ and $\mathcal{F}^{-1}(y) = \perp$ if y is not in the range of $\mathcal{F}$.

Below we define a pub-indifferentiable [11,23] construction of $\mathcal{F}$ from $\mathcal{H}$. The public indifferentiability notion differs from the standard indifferentiability notion [19,10] by the fact that in the public notion the simulator S gets to see all queries made by D.

Definition 1 (pub-indifferentiability). *A $(q_{\mathsf{D}}, q_{\mathsf{S}}, \varepsilon_{sim}, t_{sim})$-public indifferentiable construction of a random invertible function $\mathcal{F}$ from a random oracle $\mathcal{H}$ is a stateless oracle circuit C and a (stateful, probabilistic) simulator S such that for any distinguisher D making at most q_{D} oracle queries, S makes at most q_{S} oracle queries, runs in time at most t_{sim} and the following holds:*

$$|\Pr[\mathsf{D}^{\mathcal{F},\mathsf{S}^{\mathcal{F}}}(1^n) = 1] - \Pr[\mathsf{D}^{\mathsf{C}^{\mathcal{H}},\mathcal{H}}(1^n) = 1]| \leq \varepsilon_{sim},$$

here the second oracle $\mathsf{S}^{\mathcal{F}}$ gets to see also the queries made by $\mathsf{D}^{\mathcal{F},\mathsf{S}^{\mathcal{F}}}$ to the first oracle $\mathcal{F}$.

2.2 Digital Signatures with Message Recovery

A digital signature scheme with message recovery $\mathsf{SIG\text{-}MR} = (\mathsf{G}_{\mathsf{SIG\text{-}MR}}, \mathsf{Sign}, \mathsf{Recover})$ consists of three algorithms and two function families $m(n), k(n)$ describing message space $\{0,1\}^{m(n)}$ and signature space $\{0,1\}^{k(n)}$. Key generation $\mathsf{G}_{\mathsf{SIG\text{-}MR}}$ generates a keypair $(pk, sk) \leftarrow \mathsf{G}(1^n)$ for a secret signing key sk and a public verification key pk. The signing algorithm Sign, on input a message $M \in \{0,1\}^{m(n)}$ and the secret signing key, returns an enhanced signature $\tau \leftarrow \mathsf{Sign}_{sk}(M) \in \{0,1\}^{k(n)}$ of the message. The recovery algorithm $\mathsf{Recover}$ takes a verification key pk and an enhanced signature τ as input and returns $M \leftarrow \mathsf{Recover}_{pk}(\tau)$, where $M \in \{0,1\}^{m(n)} \cup \{\perp\}$. We require that $\Pr[\mathsf{Recover}_{pk}(\mathsf{Sign}_{sk}(M)) = M] = 1$.

The security of the signature scheme can be analyzed in a model where an idealized primitive exists, for example a random oracle or a random invertible function. In that case the adversary and the scheme get access to the idealized primitive $\mathcal{O}$ by making oracle calls.

Security. Let us recall the *existential unforgeability against chosen message attacks* (EUF-CMA) security game [12] relative to the ideal primitive $\mathcal{O}$, played between a challenger and a forger A.

1. The challenger runs $\mathsf{G}_{\mathsf{SIG\text{-}MR}}(1^n)$ to generate a keypair (pk, sk). Forger A receives pk as input.
2. Forger A may ask the challenger to sign a number of messages and evaluate the ideal object $\mathcal{O}$. To query the i-th signature, A submits a message $M_i \in \{0,1\}^{m(n)}$ to the challenger. The challenger returns an enhanced signature τ_i under sk for this message. For the j-th query to $\mathcal{O}$, A submits a query x_j to the challenger who returns the values $\mathcal{O}(x_j)$.
3. Forger A outputs an enhanced signature τ^*.

Let $M^* \leftarrow \mathsf{Recover}(pk, \tau^*)$ be the recovered message of A's forgery. The game outputs 1 (meaning forger A wins the game) if $M^* \neq \perp$ (i.e., τ^* is a valid enhanced signature) and $M^* \neq M_i$ for all i. The success probability of A is the probability that the game outputs 1.

Definition 2 (Security and Overhead of SIG-MR). *Let $\mathcal{O}$ be an ideal primitive and let $\mathsf{SIG\text{-}MR}^{\mathcal{O}}$ be a signature scheme with message recovery, where $\{0,1\}^{m(n)}$ is the message and $\{0,1\}^{k(n)}$ is the signature space. Let $t_{sig}, q_s, q_o, \varepsilon_{sig}$ be functions of a security parameter n.*

Security: *$\mathsf{SIG\text{-}MR}^{\mathcal{O}}$ is $(t_{sig}, q_s, q_o, \varepsilon_{sig})$-secure relative to $\mathcal{O}$, if all adversaries A running in time at most t_{sig} making at most q_s signing queries and q_o queries to $\mathcal{O}$ (this includes direct queries to $\mathcal{O}$, but also the queries to $\mathcal{O}$ done during evaluation of the signature queries), have success probability at most ε_{sig}. If $\mathcal{O}$ is a random oracle (random invertible function), then we say that $\mathsf{SIG\text{-}MR}^{\mathcal{O}}$ is secure in the random oracle (random invertible function) model.*

n-bit security: *We say* $\mathsf{SIG\text{-}MR}^{\mathcal{O}}$ *has n bits of security against q_s, q_o queries if it is $(t_{sig}, q_s, q_o, \varepsilon_{sig})$-secure for all $t_{sig}, \varepsilon_{sig}$ satisfying $\varepsilon_{sig}/t_{sig} \leq 2^{-n}$. We simply say it has n bits security if it has n bits security for any q_s, q_o (we can always assume the trivial upper bound $q_o \leq t_{sig} \leq 2^n$.*[3]*)*

Overhead: *The overhead is defined as $k(n) - m(n)$. $\mathsf{O}_{\mathsf{SIG\text{-}MR}^{\mathcal{O}}}(n, q_s, q_o)$ denotes the overhead required in the construction $\mathsf{SIG\text{-}MR}^{\mathcal{O}}$ to reach n bits security against q_s and q_o queries. $\mathsf{O}_{\mathsf{SIG\text{-}MR}^{\mathcal{O}}}(n)$ is short for $\mathsf{O}_{\mathsf{SIG\text{-}MR}^{\mathcal{O}}}(n, 2^n, 2^n)$ (i.e., when putting no upper bounds on q_o, q_s).*

In the following we will propose a scheme with finite message space. In the full version [17] we show how to do domain extension in order to get a scheme that can sign arbitrary longer messages with the same security and overhead.

Using a composition theorem [19], we can express the security of a signature scheme proven secure in the invertible function model when we replace the invertible random function $\mathcal{F}$ with an pub-indifferentiable constructions $\mathsf{C}^{\mathcal{H}}$ as follows.

Theorem 1. *If* $\mathsf{SIG\text{-}MR}^{\mathcal{F}}$ *is $(t_{sig}, q_s, q_h, \varepsilon_{sig})$-secure in the random invertible function model, and* C *is a $(q_{\mathsf{D}} = q_h, q_{\mathsf{S}}, \varepsilon_{sim}, t_{sim})$-pub-indifferentiable construction of $\mathcal{F}$ from $\mathcal{H}$ (cf. Def.1), then* $\mathsf{SIG\text{-}MR}^{\mathsf{C}^{\mathcal{H}}}$ *is $(t_{sig} - t_{sim}, q_s, q_{\mathsf{S}}, \varepsilon_{sig} + \varepsilon_{sim})$-secure in the random oracle model.*

2.3 Trapdoor Permutations

A trapdoor permutation $\mathsf{TDP} = (\mathsf{G}_{\mathsf{TDP}}, \mathsf{f}, \mathsf{f}^{-1})$ over domain $\mathcal{D}(n) = \{0,1\}^{k(n)}$ consists of three ppt algorithms. The key generation algorithm $\mathsf{G}_{\mathsf{TDP}}$ generates a keypair $(ek, td) \leftarrow \mathsf{G}_{\mathsf{TDP}}(1^n)$ of evaluation key and trapdoor. For every (ek, td) in the domain of $\mathsf{G}_{\mathsf{TDP}}(1^n)$, $\mathsf{f}(ek, \cdot)$ and $\mathsf{f}^{-1}(td, \cdot)$ compute permutations $\mathsf{f}_{ek}(\cdot), \mathsf{f}_{td}^{-1}(\cdot)$ on $\{0,1\}^{k(n)}$ s.t. for all $x \in \{0,1\}^{k(n)}$: $\mathsf{f}_{td}^{-1}(\mathsf{f}_{ek}(x)) = x$. We say TDP is homomorphic if $(\mathcal{D}(n), \circ)$ is a group and for all $x_1, x_2 \in \mathcal{D}(n)$, $\mathsf{f}_{ek}(x_1) \circ \mathsf{f}_{ek}(x_2) = \mathsf{f}_{ek}(x_1 \circ x_2)$.

We now recall the security properties of one-wayness and regular lossiness [15,22].

Definition 3 (Security of TDP). *Let $t = t(n)$ and $\varepsilon_{one-way} = \varepsilon_{one-way}(n)$ be functions of a security parameter n.* TDP *is $(\varepsilon_{one-way}, t)$-one-way if for all adversaries* A *running in time at most t, $\Pr[\mathsf{A}(ek, \mathsf{f}_{ek}(x)) = x] \leq \varepsilon_{one-way}$, where $(ek, td) \leftarrow \mathsf{G}_{\mathsf{TDP}}(1^n)$, $x \leftarrow \{0,1\}^{k(n)}$.*

Definition 4 (Lossy TDP). *Let $t_{lossy} = t_{lossy}(n)$, $\ell = \ell(n)$ and $\varepsilon_{lossy} = \varepsilon_{lossy}(n)$ be functions of a security parameter n. A trapdoor permutation* TDP *over domain $\{0,1\}^{k(n)}$ is regular $(\varepsilon_{lossy}, t_{lossy}, \ell)$-lossy if there exists a ppt algorithm G_{lossy} (the lossy key generator) that on input 1^n outputs ek' such that*

[3] As $\varepsilon \leq 1$, $\varepsilon_{sig}/t_{sig} \leq 2^{-n}$ for every $t_{sig} \geq 2^n$, so we only have to look at the case $t_{sig} \leq 2^n$.

1. *(indistinguishability of real and lossy keys) for all adversaries* A *running in time at most* t_{lossy}, $\Pr[\mathsf{A}(ek)=1]-\Pr[\mathsf{A}(ek')=1] \leq \varepsilon_{lossy}$, *where* $(ek, td) \leftarrow \mathsf{G}_{\mathsf{TDP}}(1^n)$ *and* $ek' \leftarrow \mathsf{G}_{lossy}(1^n)$;
2. *(lossiness)* $\mathsf{f}_{ek'}(\cdot)$ *is* ℓ*-to-*1, *i.e.* $\forall x \in \{0,1\}^{k(n)} : |\{z \ : \ f_{ek'}(z) = f_{ek'}(x)\}| = \ell$

For any $\ell \geq 1$, a lossy trapdoor permutation is collision-resistant when instantiated in lossy mode [22]. The most important example of a trapdoor permutation is RSA with domain $\mathbb{Z}_N^*$, defined as $\mathsf{f}_{N,e}(x) = x^e \bmod N$. It is homomorphic with respect to modular multiplication. It is one-way under the RSA assumption; for any $e < N^{1/4}$ it is furthermore regular e-lossy under the phi-hiding assumption [15], where e is the public RSA exponent. Another example of a (homomorphic and regular lossy) trapdoor function is Paillier [21].

3 Signatures with MR from Random Invertible Functions

Let $k = k(n)$ and $m = m(n)$ be functions with $k(n) \geq m(n)$. Let TDP be a trapdoor permutation over domain $\{0,1\}^k$ and $\mathcal{F} : \{0,1\}^m \to \{0,1\}^k$ be a random invertible function. We build a signature scheme with message recovery $\mathsf{SIG\text{-}MR}^{\mathcal{F}} = (\mathsf{G}_{\mathsf{SIG\text{-}MR}}, \mathsf{Sign}, \mathsf{Recover})$ with message space $\mathcal{M}(n) = \{0,1\}^m$ and signature space $\mathcal{S}(n) = \{0,1\}^k$. $\mathsf{G}_{\mathsf{SIG\text{-}MR}}(1^n)$ runs $(ek, td) \leftarrow \mathsf{G}_{\mathsf{TDP}}(1^n)$. It returns $pk = ek$ and $sk = td$.

Algorithm $\mathsf{Sign}_{sk}(M \in \{0,1\}^m)$	Algorithm $\mathsf{Recover}_{pk}(\tau \in \{0,1\}^k)$
$y := \mathcal{F}(M) \in \{0,1\}^k$	$y = \mathsf{f}_{ek}(\tau)$
Return $\tau = \mathsf{f}_{td}^{-1}(y) \in \{0,1\}^k$	If $\mathcal{F}^{-1}(y) = \perp$ then return $\perp$
	Else return $M = \mathcal{F}^{-1}(y)$

Note that $\mathsf{SIG\text{-}MR}$ has $n_1 = k - m$ bits of redundancy and correctness follows since TDP is a permutation.

The following theorem proves security provided TDP is regular lossy. Its proof is similar to the one of FDH in [15] and postponed to the full version [17].

Theorem 2. *Suppose* TDP *is regular* $(\ell, t_{lossy}, \varepsilon_{lossy})$*-lossy (i.e., lossy by* $\log_2(\ell)$ *bits) and* $\mathcal{F}$ *is a random invertible function from* $\{0,1\}^m$ *to* $\{0,1\}^k$. *Then* $\mathsf{SIG\text{-}MR}^{\mathcal{F}}$ *is* $(t_{sig}, q_s, q_f, \varepsilon_{sig})$ *secure with*

$$t_{sig} \approx t_{lossy}, \quad \varepsilon_{sig} = (2\ell - 1)/\ell \cdot \varepsilon_{lossy} + \frac{q_f}{2^{k-m}}.$$

In case TDP only satisfies the weaker security property of $(t, \varepsilon_{one-way})$-one-wayness, we only can obtain a non-tight security reduction [9] with respect to $\varepsilon_{one-way}$. As we will show now, a tight security reduction from one-wayness can be obtained by padding M with one random bit b, using a reduction technique by Katz and Wang [16]. We now define an alternative signature scheme $\mathsf{SIG\text{-}MR}_{\mathrm{ow}}^{\mathcal{F}}$ with message space $\mathcal{M}(n) = \{0,1\}^{m-1}$ which can be proved tightly secure from one-wayness of TDP.

Algorithm $\mathsf{Sign}_{sk}(M \in \{0,1\}^{m-1})$	Algorithm $\mathsf{Recover}_{pk}(\tau \in \{0,1\}^k)$
$b(M) \leftarrow \{0,1\}$	$y = \mathsf{f}_{ek}(z)$
$y := \mathcal{F}(b\|M) \in \{0,1\}^k$	If $\mathcal{F}^{-1}(y) = \perp$ then return $\perp$
Return $\tau = \mathsf{f}_{td}^{-1}(y) \in \{0,1\}^k$	Else compute $b\|M = \mathcal{F}^{-1}(y)$
	Return M

It is furthermore enforced that Sign always uses the same random bit $b = b(M)$ for message M. (E.g., by defining $b = \mathsf{PRF}_K(M)$.) Note that $\mathsf{SIG\text{-}MR}^{\mathcal{F}}_{\mathrm{ow}}$ has $k - m + 1$ bits redundancy.

The proof of the following theorem is postponed to the full version [17].

Theorem 3. *Suppose* TDP *is homomorphic and* $(t, \varepsilon_{one-way})$*-one-way and* $\mathcal{F}$ *is a random injective function from* $\{0,1\}^m$ *to* $\{0,1\}^k$*. Then the scheme* $\mathsf{SIG\text{-}MR}^{\mathcal{F}}_{ow}$ *is* $(t, q_s, q_f, 2\varepsilon_{one-way} + \frac{q_f}{2^{k-m}})$ *secure.*

4 Pub-Indifferentiable Constructions Based on Feistel Networks

4.1 The Two Round Feistel Network

Consider the two-round construction $\mathsf{C}^{\mathcal{H}}_{2f} : \mathbb{Z}_\mu \to \mathbb{Z}_\mu \times \mathbb{Z}_\rho$ Figure 1 (left) which is derived from an unbalanced two-round Feistel network ϕ_{2f} instantiated with random oracles $\mathcal{H}_1 : \mathbb{Z}_\mu \to \mathbb{Z}_\rho, \mathcal{H}_2 : \mathbb{Z}_\rho \to \mathbb{Z}_\mu$

$$\phi_{2f}(x, v) = (x + \mathcal{H}_2(\mathcal{H}_1(x) + v), \mathcal{H}_1(x) + v)$$
$$\phi^{-1}_{2f}(w, y) = (w - \mathcal{H}_2(y), y - \mathcal{H}_1(w - \mathcal{H}_2(y))$$

$$\text{as} \qquad \mathsf{C}^{\mathcal{H}}_{2f}(x) = \phi_{2f}(x, 0) \qquad \mathsf{C}^{\mathcal{H}\,-1}_{2f}(w, y) = \begin{cases} x \text{ if } \phi^{-1}_{2f}(w, y) = (x, 0) \\ \perp \text{ otherwise} \end{cases}$$

This will serve as an example of a simple indifferentiability proof and to prepare for our four round Feistel network in the next section.

Theorem 4 (pub-indifferentiability of C_{2f}, implicit in [5]). $\mathsf{C}^{\mathcal{H}}_{2f}$ *as illustrated in Figure 1 (left) is* $(q_{\mathsf{D}}, q_{\mathsf{S}}, \varepsilon_{sim}, t_{sim})$*-pub-indifferentiable from* $\mathcal{F}$ *(cf. Def. 1) where*

$$q_{\mathsf{S}} = q_{\mathsf{D}} \qquad t_{sim} = q_{\mathsf{D}} \cdot polylog(\mu) \qquad \varepsilon_{sim} = q^2_{\mathsf{D}}/\rho,$$

More precisely, we can set $t_{sim} = O(q_{\mathsf{D}} \log(q_{\mathsf{D}}) \log(\mu))$ *using that the cost per (find or insert) operation on a sorted list with* $\le q_{\mathsf{D}}$ *elements of size* $\log(\mu)$ *bits is* $O(\log(q_{\mathsf{D}}) \log(\mu))$*.*

The proof of Theorem 4 is postponed to the full version [17]. There we also formally show that a combination with Theorems 2/3 and Theorem 1 leads to the overhead of $\mathsf{O}(n, q_h, q_s) = n + \log(q_h)$ bits for the two schemes $\mathsf{SIG\text{-}MR}^{\mathsf{C}^{\mathcal{H}}_{2f}}[\mathsf{RSA}]$ and $\mathsf{SIG\text{-}MR}^{\mathsf{C}^{\mathcal{H}}_{2f}}_{\mathrm{ow}}[\mathsf{RSA}]$ in the random oracle model.

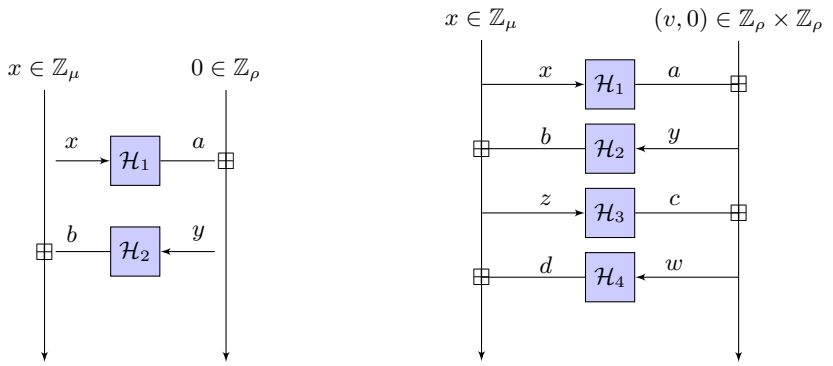


Fig. 1. (left) Two round Feistel network $\phi_{2f} : \mathbb{Z}_\mu \times \mathbb{Z}_\rho \to \mathbb{Z}_\mu \times \mathbb{Z}_\rho$, the construction $\mathsf{C}^{\mathcal{H}}_{2f} : \mathbb{Z}_\mu \to \mathbb{Z}_\mu \times \mathbb{Z}_\rho$ of a random invertible function $\mathcal{F}$ from a random oracle $\mathcal{H}$ is derived from ϕ_{2f} by setting the right part to 0, i.e. $\mathsf{C}^{\mathcal{H}}_{2f}(x) = \phi_{2f}(x, 0)$. $\boxplus$ denotes component-wise addition in the respective domains. **(right)** Four round Feistel network ϕ_{4F}, our main construction is derived from it as $\mathsf{C}^{\mathcal{H}}_{4F}(x, v) = \phi_{4F}(x, v, 0)$.

4.2 The Four Round Feistel Network

We prove the following theorem which bounds the pub-indifferentiability of our main construction $\mathsf{C}^{\mathcal{H}}_{4F}$ as illustrated in Figure 1 (right) in terms of the variable $Q(\mu, q)$ (which we mentioned in the introduction, and will discuss in detail in Section 5).

Theorem 5 (pub-indifferentiability of $\mathsf{C}^{\mathcal{H}}_{4F}$). *$\mathsf{C}^{\mathcal{H}}_{4F}$ as illustrated in Figure 1 (right) is $(q_{\mathsf{D}}, q_{\mathsf{S}}, \varepsilon_{sim}, t_{sim})$-pub-indifferentiable from $\mathcal{F}$ (cf. Def. 1) where*

$$q_{\mathsf{S}} \leq q_{\mathsf{D}} \log(\rho) \qquad t_{sim} = q_{\mathsf{S}} \cdot polylog(\mu)$$
$$\varepsilon_{sim} = \frac{2\mathsf{E}[Q(\mu, q_{\mathsf{D}})]}{\rho} + \frac{2q_{\mathsf{D}}^4}{\mu} + \frac{2q_{\mathsf{D}}^2}{\rho^2} \cdot \left(\frac{\log(\rho)}{\log(\rho/q_{\mathsf{D}})}\right)^2. \tag{3}$$

Given Theorem 5 we will now compute the concrete overhead of $\mathsf{SIG\text{-}MR}^{\mathsf{C}^{\mathcal{H}}_{4F}}[\mathsf{RSA}]$ and $\mathsf{SIG\text{-}MR}^{\mathsf{C}^{\mathcal{H}}_{4F}}_{\mathrm{ow}}[\mathsf{RSA}]$. Let $N = pq$ be the RSA modulus with $k = \log N$ and recall that $\log \mu = k - \log \rho$, where $\log \rho$ is the redundancy of the scheme. For all practically relevant values, the first term in ε_{sim} in (3) is the dominating one.[4] With the same argument as in the case of two rounds, by Theorems 2/3 and Theorem 1 the overhead for n-bit security can (up to a small additive constant) be computed as

$$\mathsf{O}(n, q_h, q_s) = n + \log \mathsf{E}[Q(\mu, q_h)] - \log q_h. \tag{4}$$

[4] Unless one proves an even stronger upper bound on $Q(\mu, q_{\mathsf{D}})$ than we do in this work, in which case the last term might become dominant for large q_{D}.

In order to bound $\mathsf{E}[Q(\mu, q_h)]$ we assume $n \leq \log \rho \leq 1.25n$ and hence $\log \mu = \log N - 2\log \rho \geq \log N - 2(1.25n)$. The following table summarizes the overhead $\mathsf{O}(n, q_s, q_h)$ for $n = 80$ bits security using (4) and the bounds on $\Pr[Q(\mu, q_h) \geq q_h 2^s]$ from Theorem 6 in Section 5. We use $\log N \in \{1024, 2048\}$ as bit-length of RSA and $\log q_h \in \{60, 80\}$ as upper bound on the random oracle queries.

$\log N$	$\log q_h$	$t = \log \mu$	s	$\Pr[Q(\mu, q_h) \geq q_h 2^s]$	$\mathsf{O}(n, q_h, q_s)$
1024	80	824	17	2^{-427} $(l = 8)$	≈ 97
1024	60	824	13	2^{-430} $(l = 10)$	≈ 93
2048	80	1848	12	2^{-230} $(l = 16)$	≈ 92
2048	60	1848	10	2^{-92} $(l = 18)$	≈ 90

4.3 Proof Intuition

For space reasons, the proof of Theorem 5 is only given in the full version [17] of this paper. In this section we give a high level intuition of the simulator, and in the next section give a proof of a combinatorial result which is at the heart of our proof.

To prove Theorem 5, we have to define a simulator $\mathsf{S}^{\mathcal{F}}$, which is given access to a random function $\mathcal{F} : \mathbb{Z}_\mu \times \mathbb{Z}_\rho \to \mathbb{Z}_\mu \times \mathbb{Z}_\rho \times \mathbb{Z}_\rho$, such that the pair of oracles $(\mathcal{F}, \mathsf{S}^{\mathcal{F}})$ behaves like $(\mathsf{C}_{4F}^{\mathcal{H}}, \mathcal{H})$.

Our simulator will internally define fake random oracles $\hat{\mathcal{H}}_i, i = 1, \ldots, 4$ by lazy sampling, and always try to make sure that they are consistent with $\mathcal{F}$ in the sense that on inputs x on which $\mathcal{F}$ has been queried, the $\hat{\mathcal{H}}_i$ are defined on all values required to evaluate $\mathsf{C}_{4F}^{\hat{\mathcal{H}}}(x)$ and moreover $\mathsf{C}_{4F}^{\hat{\mathcal{H}}}(x) = \mathcal{F}(x)$.

At some point, the simulator might not be able to define the $\hat{\mathcal{H}}$'s consistently any more due to collisions or more complicated linear relations amongst some of the inputs/outputs of $\mathcal{F}$ and the $\hat{\mathcal{H}}$'s. We can easily bound the probability of most such failure events by roughly q_{D}/ρ or less, except for one case, which we'll outline below.

Consider a q_{D} query adversary D who queries an unbalanced three round Feistel network (as illustrated in Figure 1 on the right side, but ignore the last round, and let the right half of the input be $0 \in \mathbb{Z}_\rho$ not $(v, 0) \in \mathbb{Z}_\rho \times \mathbb{Z}_\rho$). Assume the adversary queried the third oracle $\mathcal{H}_3$ on inputs $\mathcal{Z}$ and the second on inputs $\mathcal{Y}$ (receiving outputs $\mathcal{B}$). Next, D chooses some set $\mathcal{X}$ and queries the network on inputs $(x, 0)$. If for some $x \in \mathcal{X}$ we have $\mathcal{H}_1(x) \in \mathcal{Y}$ and $x + \mathcal{H}_2(\mathcal{H}_1(x)) \in \mathcal{Z}$, then the input to $\mathcal{H}_3$ on this query has been already fixed, and the simulator can't program it so it is consistent with $\mathcal{F}(x)$, we'll refer to this as a bad event below.[5] Any tuple $(x, y, z) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$ where $x + \mathcal{H}_2(y) = z$ can lead to such a

[5] The reason our actual construction needs one more round, is so we can also program the right half of the output of the network. Moreover the input to the right half contains not just the redundancy $0 \in \mathbb{Z}_\rho$, but another element $\mathbb{Z}_\rho$ which is part of the message. The reason we do this is that now the domain of the right half is large enough so we can upper bound by $q_{\mathsf{D}}^2/\rho^2 \leq q_{\mathsf{D}}/\rho$ terms which come up in the proof that depend on the collision probability of random elements over this domain.

failure with probability $\mathbb{Z}_\rho^{-1}$ (namely, if $\mathcal{H}_1(x) = y$). The number of such tuples (for an optimal choice of $\mathcal{X}, \mathcal{Z}$ for a given $\mathcal{B}$) is

$$Q(\mu, q, \mathcal{B}) = \max_{\mathcal{X},\mathcal{Z}\subset\mathbb{Z}_\mu, |\mathcal{X}|=|\mathcal{Z}|=q} |\{(x,z) \mid x \in \mathcal{X}, z \in \mathcal{Z}, z - x \in \mathcal{B}\}| \qquad (5)$$

We can thus bound the probability of this bad event by $Q(\mu, q_{\mathsf{D}}, \mathcal{B})/\rho$. Simple lower and upper bounds on $Q(\mu, q_{\mathsf{D}}, \mathcal{B})$ are[6]

$$\forall \mathcal{B} \subset \mathbb{Z}_\mu, |\mathcal{B}| = q \ : \ 2q - 1 \leq Q(\mu, q, \mathcal{B}) \leq q^2.$$

Unfortunately, there exist $\mathcal{B}$ for which the upper bound is almost achieved.[7] Fortunately, the set $\mathcal{B}$ is not adversarially chosen, but the output of a random oracle, which makes it a random subset of $\mathbb{Z}_\mu$. We thus consider the variable $Q(\mu, q_{\mathsf{D}})$ which denotes $Q(\mu, q_{\mathsf{D}}, \mathcal{B})$ for a randomly chosen $\mathcal{B}$. In order to get a good upper bound on the probability of the bad event, it thus suffices to give an upper bound on $Q(\mu, q_{\mathsf{D}})$ that holds with high probability over the choice of $\mathcal{B}$. In Section 5 we give such a bound showing that $Q(\mu, q_{\mathsf{D}})$ is $q_{\mathsf{D}}^{1+o(1)}$ with very high probability, here the $o(1)$ term goes to 0 as the ratio q_{D}/μ decreases.

5 A Bit of Additive Combinatorics

Additive combinatorics deals with questions of the sort that given an Abelian group A, find subsets $\mathcal{Z}, \mathcal{X}$ of given size that minimizes the size of

$$\mathcal{Z} - \mathcal{X} = \{z - x | z \in \mathcal{Z}, \ \ x \in \mathcal{X}\}$$

Often we also want to find out the structure of such optimal (or nearly optimal) $\mathcal{Z}, \mathcal{X}$ pairs. Such pairs are of course special, and we do not have too many of them. Analogous questions are also raised when the '$-$' is replaced with '$+$'.

Here we investigate a variant, where we also have a third set $\mathcal{B} \subseteq A$ with the same size as $\mathcal{Z}$ and $\mathcal{X}$ with the property that $z - x \in \mathcal{B}$ for an unusually large number (say, $|\mathcal{B}|^{3/2}$) of (x, z) pairs with $z \in \mathcal{Z}$ and $x \in \mathcal{X}$. We show that for an adequately small random $\mathcal{B}$ it is very unlikely that we can find *any* $\mathcal{Z}, \mathcal{X}$ such that $\mathcal{Z}, \mathcal{X}$ and $\mathcal{B}$ form a triplet as above. We may interpret our result as a property of the random Cayley graph generated by $\mathcal{B}$.

Remark 1. Although our setting is natural and undoubtedly useful for the application at hand, the problem we raise does not seem to have been studied before. An often-cited work of B. J. Green [13] computes the maximum clique

[6] To see the $2q - 1$ lower bound, add any x to $\mathcal{X}$ and let $\mathcal{Z}$ be $\{z \mid z - x \in \mathcal{B}\}$, this gives us already a set of size $|\mathcal{B}| = q$. We can get $q - 1$ more by adding any $q - 1$ elements x to $\mathcal{X}$ s.t. $x = z - b$ for some $b \in \mathcal{B}$, each increasing the set by at least 1. To see the q^2 upper bound, note that for any of the q distinct $x \in \mathcal{X}$, $z - x \in \mathcal{B}$ can hold for at most q z's as $|\mathcal{B}| = q$. This upper bound holds even if we allow $\mathcal{Z}$ (or $\mathcal{X}$) to be all of $\mathbb{Z}_\mu$.

[7] Consider the case $\mathcal{B} = \mathcal{X} = \mathcal{Z} = \{0, \ldots, q-1\}$, which shows $|Q(\mu, q, \mathcal{B})| \geq q(q+1)/2$.

size of (dense) random Cayley graphs of cyclic groups and of $\mathbb{Z}_2^n$. Other authors e.g. Christofides and N. Alon have also investigated random Cayley graphs, but with focus on Hamiltonicity, chromatic number, etc. The size of the generator set, unlike in our case, in most studies are either very small ($\text{poly}(\log |A|)$) or very large ($\Omega(|A|)$). Since spectra of random Cayley graphs have been studied, it is conceivable that there is a shorter analytic proof to our statement. We use simple combinatorics to prove our theorem.

We (non-crucially) set the Abelian group A to be the cyclic group $\mathbb{Z}_\mu$, where μ is a prime. Let $1 \leq q \leq \mu$ arbitrary, but we will think of it as a small constant power of μ, for instance $q = \mu^{0.1}$. For a set $\mathcal{B} \subset \mathbb{Z}_\mu, |\mathcal{B}| = q$ define

$$Q(\mu, q, \mathcal{B}) = \max_{\mathcal{X},\mathcal{Z} \subset \mathbb{Z}_\mu, |\mathcal{X}|=|\mathcal{Z}|=q} |\{(x, z) \mid z \in \mathcal{Z},\ x \in \mathcal{X},\ z - x \in \mathcal{B}\}| \qquad (6)$$

Expression (6) becomes a random variable $Q(\mu, q, .)$ as $\mathcal{B}$ ranges over all uniformly random $\mathcal{B} \subseteq \mathbb{Z}_\mu$ of size q. The minimum value of this random variable is at least q, because for any $\mathcal{B}$ one can choose $\mathcal{Z} = \mathcal{B}$ and $0 \in \mathcal{X}$. We show that if q is a small power of μ, the probability of the event that this random variable much exceeds q is small. To obtain practical expressions in the theorem and simpler formulas in the proof, we introduce $\mu = 2^t$ and $q = 2^r$.

Theorem 6. *For $0 < r < t/4$, and for every $s, l > 0$, $2^s \geq l^2$ it holds that*

$$\Pr[Q(2^t, 2^r) \geq 2^{r+s}] \leq 2^{-DB+t}$$

where $D = \lceil 2^{s-\frac{r}{l}}/(2l+2) \rceil$ and $B = t - l(r+1)$.

Corollary 1. *Let $q = \mu^a$, where $a \leq 1/4$. If q is large enough (while parameter a is fixed), then*

$$\Pr[Q(\mu, q) \geq q^{1+2a}] \leq 2^{-q^a/2}$$

We defer the proof of the corollary to after that of the theorem.

Proof. (of Theorem 6) Let $\mu = 2^t$ denote the size of the group, which we assume to be $\mathbb{Z}_\mu$, but this is not essential. We prove Theorem 6 by an information compression argument. What we show is that a set $\mathcal{B}$ satisfying $|\mathcal{B}| = 2^r$, $Q(2^t, 2^r, \mathcal{B}) \geq 2^{r+s}$ has a lot of constant size linear relations between its elements, which allows us to describe it with significantly less than $\log \binom{2^t}{2^r}$ bits.

In order to encode a $\mathcal{B} \subseteq \mathbb{Z}_\mu$ for which $|\mathcal{B}| = 2^r$, $Q(2^t, 2^r, \mathcal{B}) \geq 2^{r+s}$ efficiently, we show that any such $\mathcal{B}$ has a decomposition $\mathcal{B} = \mathcal{D} \cup \overline{\mathcal{D}}$, where $|\mathcal{D}| = D$ as in the theorem, $\overline{\mathcal{D}} = \mathcal{B} \setminus \mathcal{D}$, and there exist fixed $x, z \in \mathbb{Z}_\mu$ that the elements b of $\mathcal{D}$ can be ordered suitably and be expressed as

$$b = \epsilon(z - x) - \epsilon_1 b_1 - \ldots - \epsilon_{l-1} b_{l-1}, \qquad (7)$$

where $b_1, \ldots, b_{l-1}$ are either from $\overline{\mathcal{D}}$ or from elements of $\mathcal{D}$ that are expressed earlier. The numbers $\epsilon, \epsilon_1, \ldots, \epsilon_{l-1}$ are all in $\{-1, 1\}$. The saving per every item

in $\mathcal{D}$ is the difference measured in bits between its description length via (7) versus their default information cost per item. The latter is:

$$\frac{\log\binom{2^t}{2^r} - \log\binom{2^t}{2^r - D}}{D} \sim t - r$$

Since the sequence $\epsilon_1, b_1, \ldots, \epsilon_{l-1}, b_{l-1}$ together with ϵ can be described with $(l-1)(r+1)+1$ bits (each b_i is element of $\mathcal{B}$ which is already on our list, so has an r-bit description), our saving per item is

$$B = t - r - (l-1)(r+1) - 1 = t - l(r+1)$$

bits. Our total saving is then $DB - t$, since we also need t bits to describe $z - x$ (once for the entire $\mathcal{D}$). The upper bound on the probability of the event $Q(2^t, 2^r, \mathcal{B}) \geq 2^{r+s}$ is then 2^{-DB+t}.

We are left to construct the $(\mathcal{D}, \overline{\mathcal{D}})$ decomposition and to calculate D. Consider a $\mathcal{B}$ that satisfies $Q(2^t, 2^r, \mathcal{B}) \geq 2^{r+s}$. Then there are $\mathcal{X}, \mathcal{Z} \subseteq \mathbb{Z}_\mu$, $|\mathcal{X}| = |\mathcal{Z}| = 2^r$ such that $|\{(x,z) \mid x \in \mathcal{X}, z \in \mathcal{Z}, z - x \in \mathcal{B}\}| \geq 2^{r+s}$. We fix such an $\mathcal{X}, \mathcal{Z}$ pair. Let G be the bipartite graph with bipartition $(\mathcal{X}, \mathcal{Z})$ and edge set

$$e(G) = \{(x,z) \mid \ x \in \mathcal{X}, z \in \mathcal{Z}, z - x \in \mathcal{B}\}.$$

By our assumption $|e(G)| \geq 2^{r+s}$. If we iteratively remove the minimum degree vertex from G until all degrees of the resulting graph are at least $2^s/2$ (i.e. the average degree of G divided by two), it is easy to show that this process ends up with a non-empty graph G' with minimum degree at least $2^s/2$. Fix a vertex $x \in \mathcal{X} \cap V(G')$. Our proof hinges upon the following construction:

Definition 5. *Let P_i for $i = 1, 2, \ldots$ be the set of all those (not necessarily simple) paths π of length i in G' (the length is the number of edges) that satisfy:*

1. *π starts at x*
2. *No two edges edges of π have identical labels, where a label of an edge (v, w) ($v \in \mathcal{X}$, $w \in \mathcal{Z}$) is by definition $w - v$.*

Let π be a path in P_i and let $d = d(\pi)$ denote the degree of its end point z. All edges incident to z have distinct labels, so the number of those edges incident to z whose label do not coincide with any labels we already have in π is at least $d - i$. Thus π has $d - i \geq \frac{2^s}{2} - i$ continuations in P_{i+1}. Therefore, by induction, for $i \geq 1$:

$$|P_i| \geq \prod_{j=0}^{i-1} \left(\frac{2^s}{2} - j\right) > \frac{1}{e}\frac{2^{is}}{2^i}.$$

Consider the set P_l. Notice that if l is odd, then every path in P_l end in $\mathcal{Z}$, otherwise they all end in $\mathcal{X}$. Since the nodes of G' are from $\mathcal{X} \cup \mathcal{Z}$ and $|\mathcal{X}|, |\mathcal{Z}| = 2^r$, there must be a $z \in \mathcal{X}$ (if l is even) or $z \in \mathcal{Z}$ (if l is odd) such that at least $\frac{|P_l|}{2^r} \geq \frac{1}{e}\frac{2^{ls-r}}{2^l}$ paths from P_l end in z.

Let T be the set of the paths in P_l that end in this z. We will use the paths in T to find a lot of small linear relations among the elements of $\mathcal{B}$. For a path π let $\ell(\pi)$ denote the set of labels that occur on its edges, and define $\mathcal{D}_0 = \cup_{\pi \in T} \ell(\pi)$, which is just the collection of all labels that ever occur in those paths of P_l that end in z. Of course, $\mathcal{D}_0 \subseteq \mathcal{B}$, because all labels along the edges of G' are in $\mathcal{B}$. In order to estimate $|\mathcal{D}_0|$ we view a path $\pi \in P_l$ as an ordered sequence of labels. Each $\pi \in P_l$ uniquely corresponds to such a sequence of length l (although not necessarily every element of $\mathcal{D}_0^l$ is an element of P_l). Since from an alphabet of size $|\mathcal{D}_0|$ we can create at most $|\mathcal{D}_0|^l$ different sequences of length l, we have that

$$|\mathcal{D}_0| \geq |T|^{1/l} \geq \left(\frac{1}{e}\frac{2^{ls-r}}{2^l}\right)^{1/l} \geq 2^{s-r/l}/(2+2/l).$$

We are now ready to define the decomposition $\mathcal{B} = \mathcal{D} \cup \overline{\mathcal{D}}$ as promised in the beginning. The role of x and z in expression (7) will be played by the common starting- and end-point of all paths in T. For any path $\pi \in T$ we have that

$$z - x = b_1 - b_2 + b_3 - \ldots + b_l \quad \text{(if } l \text{ is odd, otherwise the last sign is a minus)}$$

It is a trivial matter to transform the above equation into (7), where b is one of the b_is (our choice which one). What remains is to show is that starting from a subset of T we can to generate all remaining elements by (7) such, that the number of generated elements is no less than the bound we require. A combinatorial lemma will help us in this.

Definition 6. *We say that a set $\{h_1, \ldots, h_{l-1}\}$ of nodes in an undirected hyper-graph $\mathcal{H}$ generates node h, if $\{h_1, \ldots, h_{l-1}, h\}$ is a hyper-edge. A generator set for $\mathcal{H}$ is a subset of nodes from which we can iteratively generate the entire vertex set of $\mathcal{H}$.*

Lemma 1. *Let $\mathcal{H}$ be an hyper-graph on m nodes such that every edge is a set of size at most l, and every node is contained in at least one hyper-edge. Then $\mathcal{H}$ has a generator of size at most $\frac{(l-1)m}{l}$.*

Proof. The proof is by induction on l. The claim is trivial for $l = 1$. Take a minimal generator set X for $\mathcal{H}$. If it does not satisfy our condition, then $|X| > \frac{(l-1)m}{l}$. Consider the hyper-graph $\mathcal{H}'$ we get from $\mathcal{H}$ by restricting all of its nodes and edges to X. Since a minimal generator set in $\mathcal{H}$ cannot properly contain any hyper-edge, every hyper-edge in $\mathcal{H}'$ has size at most $l-1$. Thus by induction $\mathcal{H}'$ has a generator set Y of size at least $\frac{(l-2)|X|}{l-1}$. But $Y \cup \overline{X}$ generates $\mathcal{H}$, and it has size at most $\frac{l-2}{l-1}|X| + m - |X| \leq \frac{(l-1)m}{l}$.

We now apply this lemma for the hyper-graph, which has vertex set $\mathcal{D}_0$ and edge set $\{\ell(\pi) \mid \pi \in T\}$. We get a generator set of size $(l-1)|\mathcal{D}_0|/l$. We put the elements of this generator set into $\overline{\mathcal{D}}$, as well as the elements of $\mathcal{B}$ that are not in $\mathcal{D}_0$. We can generate the remaining elements of $\mathcal{D}_0$ out of these via (7), and we let these form the set $\mathcal{D}$. The size of $\mathcal{D}$ is $|\mathcal{D}_0|/l = \frac{2^{s-r/l}}{(2l+2)}$.

Proof. (of Corollary 1) In Theorem 6 we set $a = r/t$, $s = 2r^2/t$, $l = \frac{3t}{4(r+1)}$. This gives $B = t/4$ and

$$D = \frac{\exp_2\left(2\frac{r^2}{t} - \frac{4r(r+1)}{3t}\right)}{2l+2} = q^{2\frac{r}{t} - \frac{4(r+1)}{3t}}/(2l+2) \geq q^{a/2}$$

if q is large enough (above $\exp_2(z)$ is by definition 2^z). Thus $2^{-DB+t} \leq 2^{-q^a/2}$ when q is sufficiently large.

Acknowledgements. We thank Mihir Bellare for suggesting the open problem to us. Furthermore, we are grateful to Yannick Seurin for pointing out a gap in the definition of the simulator in a previous version of this paper.

References

1. Abe, M., Kiltz, E., Okamoto, T.: Chosen ciphertext security with optimal ciphertext overhead. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 355–371. Springer, Heidelberg (2008)
2. Alon, N., Kaufman, T., Krivelevich, M., Ron, D.: Testing triangle-freeness in general graphs. SIAM J. Discrete Math. 22(2), 786–819 (2008)
3. Bellare, M., Ristenpart, T.: Simulation without the artificial abort: Simplified proof and improved concrete security for waters' IBE scheme. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 407–424. Springer, Heidelberg (2009)
4. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Ashby, V. (ed.) ACM CCS 1993, pp. 62–73. ACM Press (November 1993)
5. Bellare, M., Rogaway, P.: The exact security of digital signatures: How to sign with RSA and Rabin. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 399–416. Springer, Heidelberg (1996)
6. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)
7. Cachin, C., Micali, S., Stadler, M.A.: Computationally private information retrieval with polylogarithmic communication. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 402–414. Springer, Heidelberg (1999)
8. Chevallier-Mames, B., Phan, D.H., Pointcheval, D.: Optimal asymmetric encryption and signature paddings. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 254–268. Springer, Heidelberg (2005)
9. Coron, J.-S.: On the exact security of full domain hash. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 229–235. Springer, Heidelberg (2000)
10. Coron, J.-S., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-Damgård revisited: How to construct a hash function. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 430–448. Springer, Heidelberg (2005)
11. Dodis, Y., Ristenpart, T., Shrimpton, T.: Salvaging Merkle-Damgård for practical applications. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 371–388. Springer, Heidelberg (2009)

12. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM Journal on Computing 17(2), 281–308 (1988)
13. Green, B.: Counting sets with small sumset, and the clique number of random cayley graphs. Combinatorica, 307–326 (2005)
14. Holenstein, T., Künzler, R., Tessaro, S.: The equivalence of the random oracle model and the ideal cipher model, revisited. In: Fortnow, L., Vadhan, S.P. (eds.) ACM STOC, pp. 89–98. ACM Press (June 2011)
15. Kakvi, S.A., Kiltz, E.: Optimal security proofs for full domain hash, revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 537–553. Springer, Heidelberg (2012)
16. Katz, J., Wang, N.: Efficiency improvements for signature schemes with tight security reductions. In: Jajodia, S., Atluri, V., Jaeger, T. (eds.) ACM CCS 2003, pp. 155–164. ACM Press (October 2003)
17. Kiltz, E., Pietrzak, K., Szegedy, M.: Digital signatures with minimal overhead. Cryptology ePrint Archive, Report 2012/658 (2012), `http://eprint.iacr.org/`
18. Mandal, A., Patarin, J., Seurin, Y.: On the public indifferentiability and correlation intractability of the 6-round feistel construction. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 285–302. Springer, Heidelberg (2012)
19. Maurer, U.M., Renner, R.S., Holenstein, C.: Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 21–39. Springer, Heidelberg (2004)
20. Naor, A., Verstraëte, J.: A note on bipartite graphs without 2k-cycles. Comb. Probab. Comput. 14(5-6), 845–849 (2005)
21. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
22. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: Ladner, R.E., Dwork, C. (eds.) ACM STOC, pp. 187–196. ACM Press (May 2008)
23. Yoneyama, K., Miyagawa, S., Ohta, K.: Leaky random oracle. IEICE Transactions 92-A(8), 1795–1807 (2009)

Author Index