Sven Dietrich
Rachna Dhamija (Eds.)

Financial Cryptography and Data Security

11th International Conference, FC 2007, and
1st International Workshop on Usable Security, USEC 2007
Scarborough, Trinidad and Tobago, February 2007
Revised Selected Papers

Springer

Lecture Notes in Computer Science 4886

Sven Dietrich   Rachna Dhamija (Eds.)

Financial Cryptography and Data Security

11th International Conference, FC 2007, and
1st International Workshop on Usable Security, USEC 2007
Scarborough, Trinidad and Tobago, February 12-16, 2007
Revised Selected Papers

Springer

Volume Editors

Sven Dietrich
Stevens Institute of Technology, Computer Science Department
Castle Point on Hudson, Hoboken, NJ 07030, USA
E-mail: spock@cs.stevens.edu

Rachna Dhamija
Harvard University, Center for Research on Computation and Society, DEAS
Maxwell Dworkin 110, 33 Oxford Street, Cambridge, MA 02138, USA
E-mail: rachna@deas.harvard.edu

The original version of the book was revised:
The copyright line was incorrect. The Erratum
to the book is available at
DOI: 10.1007/978-3-540-77366-5_37

Preface

The 11th International Conference on Financial Cryptography and Data Security (FC 2007, http://fc07.ifca.ai), organized by the International Financial Cryptography Association (IFCA, http://www.ifca.ai/), was held in Tobago, February 12–15, 2007. The conference is a well-established and premier international forum for research, advanced development, education, exploration, and debate regarding security in the context of finance and commerce. We continue to cover all aspects of securing transactions and systems, which this year included a range of technical areas such as cryptography, payment systems, anonymity, privacy, authentication, and commercial and financial transactions. For the first time, there was an adjacent workshop on Usable Security, held after FC 2007 in the same location. The papers are included in the last part of this volume. The conference goal was to bring together top cryptographers, data-security specialists, and computer scientists with economists, bankers, implementers, and policy makers.

The goal was met this year: there were 85 submissions, out of which 17 research papers and 1 system presentation paper were accepted. In addition, the conference featured two distinguished speakers, Mike Bond and Dawn Jutla, and two panel sessions, one on RFID and one on virtual economies. As always, there was the rump session on Tuesday evening, colorful as usual.

Putting together the program was a challenging task: the Program Committee fought long and hard in online discussions in late fall 2006 over which papers to accept, assisted by the many external reviewers who brought in their respective expertise. Each paper was carefully evaluated by at least three referees. The work was made more difficult by the large number of high-quality papers received and the relatively small number which could be accepted. We would like to thank all submitters for the papers and their hard work, and hope that the comments received from the reviewers will allow them to progress with their work.

I would like to thank the General Chair, Rafael Hirschfeld, and the Sponsorship Chair, Burton Rosenberg, for all their hard work in getting this conference organized and sponsored in Tobago, its southernmost location thus far, and Jon Callas for moderating the rump session. Special thanks go to Joe McManus, Rudy Maceyko, and Jason McCormick at CERT for setting up, securing, and running the Web-based submission and reviewing system in a tight environment.

I hope all of the participants found this year's program as exciting as I did, with its continued interdisciplinary views on the subject and its strong focus on the financial side, and that the conference continues to provide an opportunity to participate in fruitful discussions on the issues and trends in the financial industry and cryptography.

June 2007   Sven Dietrich

Financial Cryptography and Data Security 2007

11th International Conference
February 12–15, 2007
Lowlands, Scarborough, Trinidad and Tobago

Program Chair

Sven Dietrich, Carnegie Mellon University, USA

General Chair

Rafael Hirschfeld, Unipay Technologies, Netherlands

Program Committee

Alessandro Acquisti	Carnegie Mellon University, USA
Jon Callas	PGP Corporation, USA
Yvo Desmedt	University College London, UK
Giovanni Di Crescenzo	Telcordia, USA
Roger Dingledine	The Tor Project, USA
Bernhard Esslinger	Deutsche Bank and University of Siegen, Germany
Philippe Golle	Palo Alto Research Center, USA
Klaus Kursawe	Philips Research Eindhoven, Netherlands
Arjen Lenstra	EPFL, Switzerland
Patrick McDaniel	Penn State University, USA
Tatsuaki Okamoto	NTT, Japan
Kazue Sako	NEC, Japan
Radu Sion	Stony Brook University, USA
Stuart Stubblebine	Stubblebine Consulting/Research and UC Davis, USA
Paul Syverson	Naval Research Laboratory, USA
Mike Szydlo	RSA, USA
Jonathan Trostle	JHU/APL, USA
Moti Yung	RSA Labs. (EMC) and Columbia University, USA
Yuliang Zheng	University of North Carolina at Charlotte, USA

External Reviewers

Farid Ahmed	Lisa Johansen
Patrick Amon	Stefan Katzenbeisser
Toshinori Araki	Tim Kerrins
Kevin Butler	Aggelos Kiayias

Bogdan Carbunar	Eike Kiltz
Scott Contini	Phil MacKenzie
David Dagon	Kengo Mori
William Enck	Rei Safavi-Naini
Allan Friedman	Gregory Neven
Jun Furukawa	Satoshi Obana
Craig Gentry	Pascal Paillier
Rachel Greenstadt	Luke St. Clair
Stuart Haber	Isamu Teranishi
Boniface Hicks	Patrick Traynor
Toshiyuki Isshiki	Shoko Yonezawa

Sponsorship Chair

Burton Rosenberg, University of Miami, USA

Sponsors

Bronze Sponsors: Google, Inc.
                 nCipher
                 PGP Corporation
                 EverBank

In-kind Sponsor: Bibit Global Payment Services

Table of Contents

Keynote Address

Payment Systems

Panel

Anonymity

Authentication

Anonymity and Privacy

Cryptography and Commercial Transactions

Financial Transactions and Web Services

Panel

Invited Talk

System Presentation

Cryptography

Usable Security Workshop

Full Papers

Short Papers

Leaving Room for the Bad Guys

Mike Bond

Cryptomathic Ltd
329 Cambridge Science Park
Milton Road
Cambridge CB4 0WG
United Kingdom
Mike.Bond@cryptomathic.com

Abstract. When designing a crypto protocol, or building a large security architecture, no competent designer ignores considering the bad guy, and anticipating his plans. But often we designers find ourselves striving to build totally secure systems and protocols, in effect writing the bad guys entirely out of the equation. In a large system, when you exclude the bad guys, they soon muscle their way in elsewhere, and maybe in a new and worse way over which you may have much less control. A crypto protocol with no known weaknesses may be a strong tool, but when it does break, it will break in an unpredictable way. This talk explores the hypothesis that it is safer and better for designers to give the bad guys their cut, but to keep it small, and keep in control. It may not just be our systems but also our protocol building blocks that should be designed to make room for the bad guy to take his cut. The talk is illustrated with examples of very successful systems with known weaknesses, drawn primarily from the European EMV payment system, and banking security in general. We also discuss a few too secure systems that end up failing in worse ways as a result.

Vulnerabilities in First-Generation
RFID-enabled Credit Cards⋆

Thomas S. Heydt-Benjamin[1], Daniel V. Bailey[2], Kevin Fu[1], Ari Juels[2],
and Tom O'Hare[3]

[1] University of Massachusetts, Amherst, MA, USA
{tshb, kevinfu}@cs.umass.edu
[2] RSA Laboratories, Bedford, MA, USA
{dbailey, ajuels}@rsa.com
[3] Innealta, Inc. Salem, MA, USA
tom@innealta.com

Abstract. RFID-enabled credit cards are widely deployed in the United
States and other countries, but no public study has thoroughly analyzed
the mechanisms that provide both security and privacy. Using samples
from a variety of RFID-enabled credit cards, our study observes that (1)
the cardholder's name and often credit card number and expiration are
leaked in plaintext to unauthenticated readers, (2) our homemade device
costing around $150 effectively clones one type of skimmed cards thus
providing a proof-of-concept implementation for the RF replay attack,
(3) information revealed by the RFID transmission cross contaminates
the security of RFID and non-RFID payment contexts, and (4) RFID-
enabled credit cards are susceptible in various degrees to a range of other
traditional RFID attacks such as skimming and relaying.

Keywords: RFID, credit cards, contactless, vulnerabilities.

1   Introduction

An increasing number of credit cards now contain a tiny wireless computer chip
and antenna based on RFID (Radio Frequency Identifier) and contactless smart-
card technology. RFID-enabled credit cards permit contactless payments that
are fast, easy, often more reliable than magstripe transactions, and require only
physical proximity (rather than contact) between the credit card and the reader.
An estimated 20 million RFID credit cards and 150,000 vendor readers [6] are
already deployed in the U.S. According to Visa USA [6], "This has been the
fastest acceptance of new payment technology in the history of the industry."

The conveniences of RFID credit cards also lead to new risks for security and
privacy. Traditional credit cards require visual access or direct physical contact for
retrieving information such as the cardholder's name and the credit-card number.
By contrast RFID credit cards make these and other sensitive pieces of data avail-
able using a small radio transponder that is energized and interrogated by a reader.

Experimental Results: Although RFID-enabled credit cards are widely reported to use sophisticated cryptography [3,15,18,21,29,32], our experiments found several surprising vulnerabilities in every system we examined. We collected two commercial readers from two independent manufacturers and approximately 20 RFID-enabled credit cards issued in the last year from three major payment associations and several issuing banks in the U.S. We were unable to locate public documentation on the proprietary commands used by RFID-enabled credit cards. Thus, we reverse engineered the protocols and constructed inexpensive devices that emulate both credit cards and readers. The experiments indicate that all the cards are susceptible to live relay attacks, all the cards are susceptible to disclosure of personal information, and many of the cards are susceptible to various types of replay attacks. In addition, we successfully completed a proof-of-concept cross-contamination attack.

Given the size and diversity of our sample set we believe that our results reflect the current state of deployed RFID credit cards; however, card issuers continue to innovate and will likely add new security features. Our findings are not necessarily exhaustive, and there may exist cards that use security mechanisms beyond what we have observed.

1.1   Background

Scale of Current Deployment: Several large chain stores in the U.S. have deployed many thousands of RFID readers for credit cards: CVS Pharmacies (all 5,300 locations), McDonald's (12,000 of 13,700 locations), the Regal Entertainment Group of movie theaters, and several other large vendors [26,30]. Reports estimate that 20 to 55 million RFID-enabled credit cards are in circulation, which is 5% to 14% of all credit cards [4,6,26]. In addition to traditional payment contexts, RFID-enabled credit cards are becoming accepted in other contexts such as public transportation [20]. The New York City subway [28] recently started a trial of 30 stations accepting an estimated 100,000 RFID-enabled credit cards [7]. A participant in this trial uses her credit card as a transit ticket as well as a credit card in place of the traditional magstripe-based dedicated subway tickets.

Integration of RF Technology Into Existing Credit-Card Infrastructure: In a typical deployment, an RFID-enabled credit card reader is attached to a traditional cash register. Each reader continually polls for cards by broadcasting a radio signal, to which RFID enabled credit cards can respond. The RFID payment cards that we examined seem to have been designed specifically for easy integration into the existing payment-authorization infrastructure. For instance, even though no magnetic stripes are read during an RF transaction, the RFID credit card readers that we examined reformat received RFID data into "Track 1 Data" and "Track 2 Data" before passing it along to point-of-sale terminals. In other words, data is presented to the charge-processing network in the same format regardless of whether the credit-card reader received the information from an RF transaction, or a traditional swipe of a magnetic strip.

Our work focuses on the first step in a long chain of system interactions: card presentation. When considering the potential impact of the vulnerabilities we have observed in RFID card presentation, one must take into account the expertise credit card issuers have gained in detecting fraudulent transactions by tracking patterns of behavior [11]. While detecting fraud is an effective defense against many types of financial risk, it does not *prevent* invasion of privacy. Our study considers vulnerabilities to privacy that today's anti-fraud methods do not prevent.

Communications Protocol Used by RFID Credit Cards: All of the credit cards we tested use a communications protocol specified by the International Organization for Standardization in a series of documents titled ISO 14443-1 through 14443-4 [22]. Our experiments indicate that the cards use the B version of this protocol, with an additional proprietary communications layer carried over ISO layer 4.

2   Related Work

RFID-enabled credit cards share many of the challenges and approaches for security and privacy as other RFID-based authentication and identification systems.

RFID Authentication and Cloning: Many types of RFID tags merely emit static identifiers, making them easy to clone. These tags are sometimes used in inappropriate contexts such as building access control. Westhues has demonstrated a simple, inexpensive device that can skim many types of cards at a distance—even through walls—and then simulate them [35]. If unclonability is a security assumption, then this is a security break.

More sophisticated tags do not emit static data, but use cryptography to emit different data during different transactions. For example the Texas Instruments Digital Signal Transponder (DST) is present in the ExxonMobil Speedpass, and is also part of a common theft deterrent system for automobiles. These systems have been shown to be vulnerable because of faulty cryptography [5]. To contrast with the RFID credit cards we have examined: the DST uses cryptography to increase the difficulty of cloning, but does not carry personally identifying information, e.g., the name of its owner.

Read Ranges: Industry claims around the security of RFID devices often hinge on their short read ranges. Some cautionary notes are in order, however: RFID tags do not have a single, definitive read range [23]. While the *nominal* read range of an RFID tag may be quite short, a non-standard reader or large antenna can increase the range at which an attacker can skim an RFID tag. The credit cards we examined are ISO 14443-B cards with a nominal range of 4 to 5 centimeters. Skimming ranges of over 20cm have been demonstrated for cards of this type [17] and ranges of up to 50cm are hypothesized in the literature [25].

Furthermore, while skimming requires that a reader power the targeted tag, an attacker performing passive eavesdropping on a session between a legitimate reader and RFID tag can potentially harvest tag data at a considerably longer range. Claims have surfaced of tests where e-passports, which rely on the same ISO standard as credit cards, were read at a distance of 30 feet [35][1] and detected at a distance of 20 meters [13].

Our study makes no claims about the read ranges of RFID-enabled credit cards beyond the observation that characterization of these ranges is not straightforward and constitutes an important open research question.

3   Methodology and Experiments

The following sections highlight our methodology for testing security of RFID-enabled credit cards against eavesdropping, skimming, and replay. A more detailed version is available in our technical report [19].

Eavesdropping Experiments: In our eavesdropping experiments we observed transactions between readers and cards with an oscilloscope attached to an antenna. Examination of data thus obtained demonstrated the efficacy of this simple attack, since the full cardholder name and card expiration date were present in cleartext in all transactions. A majority of cards examined transmitted credit card number in cleartext, while a minority broadcast a separate (but static) credit card number apparently reserved for wireless transactions. Section 4 provides further details.

Skimming Experiments: In our most simple skimming experiment we took a commercial RFID credit card reader and presented it with each of our experimental cards, obtaining in each case ISO 7813 (magstripe style) data. Since this is the exact data that is normally transmitted by a POS terminal to a charge processing network, this most naive of skimming attacks is sufficient for perpetration of certain kinds of financial fraud.

We programmed an RFID reader not intended for credit card use to emulate an RFID credit card reader. Eavesdropping on transactions between our credit card reader emulator and real RFID credit cards demonstrated that all of the RFID credit cards we tested responded to our emulator exactly as they respond to a commercial RFID credit card reader. This strongly suggests that cards do not use any secure mechanism to authenticate an authorized RFID reader before releasing sensitive information.

Replay Experiments: Our credit card emulator is a microprocessor controlled device with a simple radio permitting broadcast of arbitrary bytes over the ISO 14443-B transport layer.

---

[1] While the referenced report is short on details, it seems likely that the tests involved passive eavesdropping of some kind, rather than direct skimming.

Fig. 1. Our assembled credit card emulator

We programmed our credit card emulator to expect the RFID credit card reader commands that we captured during eavesdropping experiments, and to transmit replies captured from real RFID credit cards during a skimming attack performed with the reader emulator described above. In our experiments commercial readers were unable to distinguish between our emulated card and the real card upon which it was based.

Since the output from the card emulator is identical to that of the real card from which it was skimmed, a simple replay attack using this device would succeed. As noted above, many pieces of data go into an overall transaction approval decision including sophisticated risk-based fraud detection mechanisms on the back end. For this reason, a valuable future research direction would include field tests in which a credit card emulator is used to perform a purchase in a retail location rather than a laboratory.

4   Analysis and Results

To protect the identity of our cards, we label the cards A, B, C, and D based on semantic equivalence classes determined by observing behavior between cards and readers. Table 1 summarizes some of the vulnerabilities of three classes of cards.

Table 1. A summary of susceptibility to various attacks for the three semantic types of cards (A, B, C) from three payment associations (1, 2, 3). *Because the cards have no shielding or notion of time, all the cards are susceptible to relay. **This attack is proven in the field, but is limited to certain merchants. ***This card admits unrestricted replay for the readers we tested, while the others induce a race condition.

Card Type	Payment Association	Privacy Invasion?	Relay* Attack?	Cross-Contamination?	Replay Attack?
A	1	Yes	Yes	Limited**	Yes***
B	2	Yes	Yes	Limited	Limited
C	3	Yes	Yes	No	Limited

4.1   Observations of RFID-enabled Credit Card Protocols

This section explores some of the RFID credit card protocols that are in current deployment. The analysis is based on the ISO 7813 (magstripe format) data output by the serial port of RFID credit card readers when presented with different types of credit cards. Where pertinent, our analysis compares this serial output with the raw RF data from the same transactions as captured by our eavesdropping apparatus.

In keeping with a philosophy of ethical attacks research, we have redacted several pieces of information from the following subsections in part because of a desire to prevent criminal misuse of our findings. Cardholder name and card number have been concealed. Additionally we have obscured the number of digits in the card number in order to obscure which observations correlate with the products of specific payment associations and issuing banks.

Card A Protocol: When presented (RF transaction) with any sample of a card of type A, our reader outputs serial data identical to the data contained on the magstripe of the same credit card. When presented with the same card, the output is always the same: in the serial output there is no evidence of a counter, one-time-password, or any other mechanism for prevention of replay attacks.

Bxxxxxx6531xxxxxx^DOE/JANE^09061010000000000000000000000000000858000000
xxxxxx6531xxxxxx=09061010000085800000

Fig. 2. Serial output from a commercial reader after an RF transaction with a card of type A

Card B Protocol: The sample card B output in Figure 3 demonstrates the presence of a counter, determined to be such because of monotonic incrementation with successive transactions. Additionally we observe three digits that change with each transaction in no pattern that we have identified. Because of the relatively high entropy of these three digits, we consider it likely that they are the output of some cryptographic algorithm that takes the transaction counter as an input. If this is the case, then the algorithm must also take a card-specific value like a cryptographic key as an input since we observe that different cards with the same counter value produce different codes. We speculate that these data may serve as a stand-in for the traditional CVC.

Bxxxxxx1079xxxxxx^DOE/JANE^09011011000000000000100000000000
xxxxxx1079xxxxxx=09011011000*016*00221
Bxxxxxx1079xxxxxx^DOE/JANE^09011011000000000000100000000000
xxxxxx1079xxxxxx=09011011000*074*00231

Fig. 3. Sample of reader serial output after RF transaction with a card from issuer B. In this sample we see a three digit code (shown in bold italic font), and a four digit counter (shown underlined).

Card C Protocol: Card C's protocol differs from Card B's in a few crucial details:

1. its unique transaction codes are eight digits instead of three
2. its transaction counter, now located in the Cardholder Name field, displays only three digits instead of four
3. rather than sending the embossed card number over the air, it uses a fixed pseudonym

Bxxxxxx2892xxxxxx^DOE/JANE   <u>017</u>^10011010***10691958***
xxxxxx2892xxxxxx=10011010***10691958***01700
Bxxxxxx2892xxxxxx^DOE/JANE   <u>018</u>^10011010***40146036***
xxxxxx2892xxxxxx=10011010***40146036***<u>018</u>00

Fig. 4. Sample output from a card of type C. Transaction codes are shown in bold italic font, transaction counter is shown underlined.

4.2   Analysis of RFID-enabled Credit Card Protocols

The following sections analyze the susceptibility of the card types to replay, relay, cross-contamination, and privacy/tracking attacks. Our analysis considers only the protection mechanisms of the cards and readers, not the security of the charge processing network (e.g., fraud detection algorithms).

Replay Attacks: Replay attacks come in several flavors depending on what data are communicated from the credit card all the way to the back end charge processing network.

1. Unrestricted replay: A card that always reports the same data need be scanned only once. After that, the attacker can replay the captured data at will, and the processing network cannot detect any difference between a replay and successive transactions with a real card. Since we observed the serial output from real POS readers to always be static with respect to cards of type A, we conclude that cards of this type are susceptible to this attack.
2. Replay with race condition: A card that uses a transaction counter and rolling code poses more of a challenge if the back end processing network stores and checks counter values. In such a case, once transaction n has been accepted by the network, transactions numbered less than n should be declined if presented. However, if an adversary skims a transaction from a card, then replays that transaction to the network before the legitimate user has a chance to use their card, then the charge-processing network should accept the adversary's transactions, and actually decline the legitimate ones. Although the attacker is faced with a counter synchronization problem, such challenges are far easier to defeat than the cryptographic problems on which we prefer to base our security whenever possible.

3. Counter rollover: If a transaction counter is the only changing input to a code, then the number of possible codes is limited by the maximum possible transaction counter value. There are then two cases; in one the counter is permitted to roll over, repeating from the beginning, thus also repeating the codes from the beginning. In the other case the card refuses to engage in additional transactions after the counter is exhausted.

In the first case, an adversary that enjoys sufficient time in proximity to a card can build a database of all possible counter values and their corresponding codes, and therefore can mimic all possible behavior of the target card. Cards of type B are susceptible to this attack.

In the second case, denial-of-service can be perpetrated against the card if the attacker has sufficient time in proximity to exhaust the counter by repeated skimming. Our experiments determined that cards of type C exhibit this behavior.

Relay Attack: Even in the case of a hypothetical card we have not examined that combines a challenge-response protocol with a transaction counter, the relay attack [16] may still succeed. In an example relay attack, the adversary consists of a *mole* and a *proxy* that perform a purchase at an innocent user's expense. The mole possesses a clandestine credit card *reader emulator* with a (non-RFID) radio link to the proxy's clandestine credit *card emulator*. The mole sits down or stands next to the user, and the mole's device rapidly discovers the user's credit card. The proxy receiving this relayed signal approaches the POS terminal and initiates a purchase. The proxy presents his credit card emulator to the POS terminal. The emulator receives commands from the POS terminal and relays them to the mole's device, which transmits the commands to the user's credit card. The responses from the user's card are likewise relayed through the mole's device and are broadcast from the proxy's emulator to the POS terminal. The purchase should succeed, and the cost will be charged to the user. Observe that even with application-layer challenge-response or transaction-counter protocols, this attack will still succeed, as protocol messages will simply be relayed between the card and reader.

Cross-contamination Attack: To analyze the feasibility of a cross-contamination attack, we took a credit card of type A, placed it in a sealed envelope, and performed a "Johnny Carson attack" by reading the card through the envelope using our custom programmed TI s4100 reader.

We combined the data thus obtained with address and telephone information looked up in the telephone directory given the cardholder name transmitted through the envelope (for postal mail, the attacker already knows the cardholder address!). Using only this information we placed an online purchase for electronic parts from one of our major research-parts suppliers. Our purchase was successful, and we conclude that the cross-contamination attack is effective for cards of type A and merchants that do not require a CVC.

Privacy Invasion and Tracking: Our eavesdropping transcripts show that personally-identifying information is broadcast in cleartext by every RFID-enabled credit card we have examined.

This must be considered a privacy vulnerability in that automated, full identification of a person carrying an RFID credit card is easily demonstrated in the lab, and should be feasible in the field. This vulnerability is exacerbated by an adversary who could use the full identity disclosure of the RFID credit card to build up a database of associated pseudonyms based on other RFID tags with longer read range that a user may commonly carry.

In addition, the transaction counter found in some of the cards could be exploited by a vendor: by storing the transaction counter, a retailer could tell how often the card was used to purchase goods from others. Heavy card-users might be targeted for specific advertising, for instance.

5   Countermeasures

In addition to fraud detection to limit financial risk, several other countermeasures could significantly reduce risk of fraud and invasion of privacy.

Shielding and Blocking: One countermeasure to some cases of skimming and relay attacks is to ensure that credit cards are unreadable when not in use. A Faraday cage is a physical cover that assumes the form of a metal sheet or mesh that is opaque to certain radio waves. Consumers can today purchase Faraday cages in the form of wallets and slip-cases to shield their RFID-enabled cards against unwanted scanning [10]. Note that this countermeasure offers no protection when the card is in use, since a card must be removed from a shielded wallet before an RF purchase can be made. However, credit card companies ought to at least ship cards through the mail enclosed in a Faraday cage to obviate the dangers of the Johnny Carson attack.

A slightly more sophisticated approach to preventing attack against dormant RFID devices is to disrupt ambient RFID communication. Blocker tags [24] and the RFID Guardian [31] are two examples of devices that can selectively disrupt RFID communications to offer tag owners improved access control.

Signaling Cardholder Intent: As an alternative approach to protection, the credit cards themselves could be modified to activate only after indication of user intent. A simple push-button [33] would serve this purpose, but more sophisticated sensors might serve the same purpose, such as light sensors that render cards inactive in the dark, heat sensors that detect the proximity of the human hand, motion sensors that detect a telltale "tap-and-go" trajectory, etc. Ultimately, credit-card functionality will see incorporation into higher-powered consumer devices, such as NFC-ready mobile phones, and will benefit from the security protections of these host devices, such as biometric sensors and increased computational capacity [8].

Better Cryptography: Contactless smart cards capable of robust cryptography have long been available. These techniques have already been applied to payment cards in the EMV standards, detailed in Section 6. If personally identifiable data can only be decrypted by authorized readers, then the danger of many of the privacy-invasion attacks discussed in the paper are obviated. Anecdotal accounts suggest payment associations are moving to improve the on-chip cryptographic features of these cards, including challenge-response protocols to further frustrate replay attacks.

6   Discussion

As time goes on and technology costs decrease, we can expect issuers to provide more effective cryptographic protocols. Well-established methods to thwart these attacks already exist and issuers may in fact already be implementing these defenses. But even today, in most cases a financially motivated attacker has easier avenues to exploit than RF based attacks in order to perpetrate financial fraud. For instance, simple cloning of cards is often not sufficient to commit fraud. There are many back-end fraud-detection measures in place to help thwart fraudulent use of card information. Nevertheless privacy vulnerabilities should be addressed wherever they are found; privacy invasion may lead to financial fraud, but preventing financial fraud is not the only reason to protect privacy.

Comparison with Other Types of Fraud: It is hard to directly compare the security of traditional magstripe cards and RFID-enabled cards. RFID-enabled cards are only more secure than their traditional counterparts against *certain kinds* of attacks. For example some traditional card reading mechanisms, such as taking a physical carbon copy of the face of the card, leave a physical image of the card in the hands of a possibly adversarial merchant or clerk. In fact, the use of a magstripe generally means handing one's card to a clerk who may have nefarious intent. By contrast, an RFID transaction leaves behind no physical carbon copy; in fact the card never leaves the cardholder's hands. Certainly, the effort required to obtain an RF copy of the transaction is greater in this case.

Additionally some RFID-enabled cards include a unique code for each transaction replacing the static data in a magstripe. This mechanism protects against some kinds of attacks, but creates opportunities for new types of attacks that cannot be easily addressed by traditional fraud control (such as cardholder tracking attacks).

Perhaps the most important difference between RFID-enabled cards and traditional cards is the difference in cardholder control. Whereas a traditional magstripe reveals one's name and card number only when the artifact is physically handed to a merchant, an RFID enabled card is in some sense "always on." The card can be scanned and privacy can be compromised remotely and without the knowledge or consent of the cardholder.

Comparison with Other Electronic Cards: The relationship between the cards we examined and the EMV series [12] of standards is unclear. Certainly in Europe, EMV techniques like the UK's "Chip and PIN" are seeing wide deployment and analysis [1,2,34]. But based on our observations, the protocols used by the U.S. contactless cards do not appear in the EMV standards.

It is not clear to us why the U.S. payment associations have chosen to develop new protocols, with significant vulnerabilities, rather than use the more secure protocols that are already deployed in Europe. We can surmise that this choice was motivated by the prevalence of online readers in the U.S. (some of the expense of supporting the EMV standards has to do with support for offline operation) and a focus on contactless operation (whereas most of Europe's cards are contact based).

Policy and Regulation: Several state legislatures have recently considered bills on RFID. For instance, Gov. Schwarzenegger recently vetoed California's SB 768, which would have required interim protections for RFID cards, especially cards carrying personally identifiable information, and a process for figuring out long-term protections [14,27]. The information made available by the cards, including name and card number are called personally identifiable information (PII) in the parlance of that bill [27]. If signed into law, ID cards issued by the state government carrying PII would have been required to implement mutual authentication and encryption to release the data. While credit cards are not state ID cards, as time goes on we can expect more RFID-related legislation like SB 768 to be introduced. Indeed, U.S. Senator Charles Schumer recently announced his intent to increase federal regulation of RFID-enabled credit cards [9].

Beyond regulation, it is an important open problem how best to offer incentives to all custodians of personal data to take adequate precautions. The core of the financial industry is risk management. However, we have yet to find a satisfying definition of privacy for the equation of risk management. How do we quantify user privacy when different users place different values on privacy? In hard figures, how does this value affect the bottom line of businesses that are custodians of personal-data?

7   Conclusion

Despite the millions of RFID-enabled payment cards already in circulation, and the large investment required for their manufacture, personalization, and distribution, all the cards we examined are susceptible to privacy invasion and relay attacks. Some cards may be skimmed once and replayed at will, while others pose a modest additional synchronization burden to the attacker. After reverse engineering the secret protocols between RFID-enabled credit cards and readers, we were able to build a device capable of mounting several advanced replay attacks under laboratory conditions. While absolute security and privacy in a contactless-card form factor may be impossible to achieve, we hope that next-generation RFID-enabled payment systems will protect against the vulnerabilities that our study identifies.

Acknowledgments

References

1. Adida, B., Bond, M., Clulow, J., Lin, A., Murdoch, S., Anderson, R., Rivest, R.: Phish and chips: Traditional and new recipes for attacking EMV. Technical report, University of Cambridge Computer Laboratory (2006), http://www.cl.cam.ac.uk/~mkb23/research/Phish-and-Chips.pdf
2. Anonymous: Chip and spin (2006), http://www.chipandspin.co.uk/problems.html
3. Associated Press: Wave the card for instant credit. Wired News (2003), http://tinyurl.com/yc45ll
4. Averkamp, J.: ITS Michigan: Wireless technology and telecommunications (2006), http://www.itsmichigan.org/ppt/AM2005/Joe.ppt
5. Bono, S., Green, M., Stubblefield, A., Juels, A., Rubin, A., Szydlo, M.: Security analysis of a cryptographically-enabled RFID device. In: 14th USENIX Security Symposium (2005)
6. Bray, H.: Credit cards with radio tags speed purchases but track customers, too. Boston Globe (August 14, 2006), http://tinyurl.com/lmjt4
7. CardTechnology: Paypass subway trial starts in New York (2006), http://tinyurl.com/uya3k
8. Carey, D.: NFC turns phone into a wallet. EE Times (2006), http://tinyurl.com/yyxk28
9. Chan, S.: Metro briefing — New York: Manhattan: Warning about credit risks. The New York Times (2006), http://www.nytimes.com/2006/12/04/nyregion/04mbrfs-credit.html
10. DIFRWear: Faraday-Caged Apparel. (2006), www.difrwear.com
11. Dougherty, G.: Real-time fraud detection. MIT Applied Security Reading Group (2000), http://pdos.csail.mit.edu/asrg/02-28-2000.html and http://pdos.csail.mit.edu/asrg/02-28-2000.doc
12. EMVCo: EMV Integrated Circuit Card Specifications for Payment Systems (2004), http://tinyurl.com/oo663
13. EPIC: Mock point of entry test findings, p. 48 (2005), http://www.epic.org/privacy/us-visit/foia/mockpoe_res.pdf
14. Ferguson, R.: Schwarzenegger quashes RFID bill. eWeek DATE (2006), http://tinyurl.com/y29z6s
15. Greenemeier, L.: Visa expands contactless card efforts. Information Week (2006), http://tinyurl.com/ykzo4t
16. Hancke, G.P.: A practical relay attack on ISO 14443 proximity cards. Technical report, University of Cambridge Computer Laboratory (2005), http://www.cl.cam.ac.uk/~gh275/relay.pdf

17. Hancke, G.P.: Practical attacks on proximity identification systems (short paper). In: Proceedings of IEEE Symposium on Security and Privacy, pp. 328–333 (2006), http://www.cl.cam.ac.uk/~gh275/SPPractical.pdf

18. Harper, J.: RFID wiggles its way into credit cards? (2005), http://lists.jammed.com/politech/2005/05/0038.html

19. Heydt-Benjamin, T.S., Bailey, D.V., Fu, K., Juels, A., O'Hare, T.: Vulnerabilities in first-generation RFID-enabled credit cards. Technical report, University of Massachusetts Amherst, CS TR-2006-055 (2006)

20. Heydt-Benjamin, T.S., Chae, H.J., Defend, B., Fu, K.: Privacy for public transportation. In: Danezis, G., Golle, P. (eds.) PET 2006. LNCS, vol. 4258, Springer, Heidelberg (2006)

21. HowStuffWorks, Inc.: How blink works (2006), http://money.howstuffworks.com/blink1.htm

22. ISO: ISO/EIC 14443, proximity cards (PICCs). Technical report, ISO (2006), http://wg8.de/sd1.html

23. Juels, A.: RFID security and privacy: A research survey. IEEE Journal on Selected Areas in Communication 24(2) (2006)

24. Juels, A., Rivest, R.L., Szydlo, M.: The blocker tag: selective blocking of RFID tags for consumer privacy. In: CCS 2003. Proceedings of the 10th ACM conference on Computer and Communications Security, pp. 103–111 (2003)

25. Kfir, Z., Wool, A.: Picking virtual pockets using relay attacks on contactless smart-card systems. In: IEEE/CreateNet SecureComm., IEEE, Los Alamitos (2005), http://eprint.iacr.org/2005/052

26. Koper, S.: Contactless acceptance made easy for business payment systems. In: BPS 2006 Summer Conference, Las Vegas, NV (2006), http://tinyurl.com/sjte6

27. Molnar, D.: Personal communication (2006)

28. New York City Transit Authority: NYC MetroCard Fares. In: WWW (2006), http://tinyurl.com/y5egfd

29. O'Connor, M.C.: Chase offers contactless cards in a blink. RFID Journal (2005), http://tinyurl.com/yzy9u5

30. O'Connor, M.C.: At McDonald's, ExpressPay fits the bill. RFID Journal (2006), http://tinyurl.com/yc58sa

31. Rieback, M., Gaydadjiev, G., Crispo, B., Hofman, R., Tanenbaum, A.: A platform for RFID security and privacy administration. In: Proc. USENIX/SAGE Large Installation System Administration conference, Washington, DC, USA, pp. 89–102 (2006), http://www.rfidguardian.org/papers/lisa.06.pdf

32. Schuman, E.: How safe are the new contactless payment systems? (June 20, 2005), http://tinyurl.com/y9a525

33. Selker, E.: Manually-operated switch for enabling and disabling an RFID card. Technical report, MIT, Patent #20030132301 (2003)

34. UK Chip and Pin: Chip and pin (2006), www.chipandpin.com

35. Westhues, J.: Hacking the prox card. In: Garfinkel, S., Rosenberg, B. (eds.) RFID: Applications, Security, and Privacy, pp. 291–300. Addison-Wesley, Reading (2005)

36. Yoshida, J.: Tests reveal e-passport security flaw. EE Times (August 30, 2004), http://tinyurl.com/surgr

Conditional E-Cash

Larry Shi[2], Bogdan Carbunar[2], and Radu Sion[1]

[1] Network Security and Applied Cryptography Lab
Computer Science, Stony Brook University
sion@cs.stonybrook.edu
[2] Pervasive Platforms and Architectures
Motorola Labs
{larry.shi,carbunar}@motorola.com

Abstract. We introduce a novel *conditional* e-cash protocol allowing future anonymous cashing of bank-issued e-money only upon the satisfaction of an agreed-upon public condition. Payers are able to remunerate payees for services that depend on future, yet to be determined outcomes of events. Once payment complete, any double-spending attempt by the payer will reveal its identity; no double-spending by the payee is possible. Payers can not be linked to payees or to ongoing or past transactions. The flow of cash within the system is thus both correct and anonymous. We discuss several applications of conditional e-cash including online trading of financial securities, prediction markets, and betting systems.

1 Introduction

Electronic cash (e-cash) instruments allow digital payment for goods and services. Desirable properties of such protocols include: the ability to effect anonymous payments, the detection and prevention of malicious behavior (e.g., double spending), as well as the transactional consistency of the participants' financial state. A multitude of e-cash protocols have been proposed in the recent past. The main desiderata in such efforts has often been achieving digitally, levels of similarity and ease of use comparable to physical cash.

There are scenarios however, where basic e-cash properties are not sufficient. Here we consider the case of payments conditional on unknown future outcomes. In such settings, payers require the ability to anonymously remunerate payees for items that depend on future, yet to be determined outcomes of events. Prominent examples include trading of financial market instruments such as futures and securities [7,8,23], and other online protocols involving deferred conditional payments such as betting.

Correctness assurances are essential. Payees need to be confident that payment *will* occur with certainty for favorable future event outcomes. Payers should be able to cash back un-cashed issued conditional payments for events with unfavorable outcomes. Overall monetary consistency needs to be preserved.

We note that trivial designs for such mechanisms can be envisioned, e.g., involving the e-cash issuing institution (i.e., bank) as a trusted arbitrator. Such

assumptions, however, are rarely desirable. Requiring knowledge about the semantics of each and every considered future event at the bank is not scalable for even moderate transaction throughputs, considered events, and number of parties[1]. Moreover, an important concern in such scenarios is the privacy of participants. It is important to protect the privacy of interactions between payer and payee entities. Revealing identities should only be possible as a counter-incentive for faulty behavior (e.g., double spending) and specifically not during a correct run of the protocol.

Thus, one of the main challenges of a sound design is assuring participants' privacy while guaranteeing the conditional nature of payments. Payers and payees will naturally know each other, either by knowing each other's identity or at least by having access to a pre-authenticated channel through which to transfer public keys. No other party however should be able to associate them with each other and the conditional payments. While many existing e-cash protocols provide for participant anonymity, they cannot be directly deployed for payments of a conditional nature.

In this paper we introduce a new *conditional* e-cash protocol featuring the following properties. A payer can ask her bank to issue an anonymous payment token that can be cashed by any potential payee, *once* and if and only if a trusted publisher[2] will publish a specific secret (which only the publisher can do) in the future. In effect, payers are now able to remunerate payees (e.g., merchants) anonymously, for services that depend on future, yet to be determined outcomes of events. Once payment complete, any double-spending attempt by the payer will reveal its identity. Moreover, no double-spending by the payee is possible. Payers can not be linked to payees or to ongoing or past transactions. The flow of cash within the system is thus both correct and anonymous.

We explore a series of applications for conditional payments, including the online trading of securities, prediction markets, and online betting protocols.

The paper is organized as follows. We discuss the operational and adversarial models in Section 2. We introduce and analyze the payment protocol in Section 4 and explore several applications such as anonymous online betting in Section 5. We discuss related work in Section 3 and conclude in Section 6.

2   Model

A payer remunerates a payee by providing a payment token that can be activated and cashed at a specific bank, but only when a secret is published by a trusted publisher upon the completion of a certain agreed-upon event with a "favorable" outcome (e.g., stock price below given threshold, horse won race). Events with two possible outcomes will be considered ("favorable" – payment should be honored, and "unfavorable"). No other party but the publisher can generate the secret (under computational intractability assumptions). Without

---

[1] Additionally, arguably, very few banks would enter such an arbitration business.
[2] The publisher can be considered a "manager" of events – e.g., a stock market administrator, a race organizer.

sacrificing generality, we will consider a single such event/secret combination, but possibly many payees and payers exchanging conditional payments for one event. The protocol guarantees the following:

> **P1.** The bank is not able to associate previously issued conditional payments (to payers) with identities of principals (payers or payees) cashing them later.
>
> **P2.** Double spending by both the payers and the payees is prevented. Moreover, if a payer re-uses the payment token for a different payee, its identity is revealed to the bank.
>
> **P3.** The payer is able to cash back the payment token in the case of an unfavorable outcome.
>
> **P4.** Once the payee accepts the conditional payment from the payer, she will be able to cash it in with high probability in the case of a favorable outcome, when the publisher publishes the associated enabling secret. In this case, if the payer attempts to spend the payment token the payer's identity will be revealed to the bank (this is discussed in P2).
>
> **P5.** The publisher cannot infer any information about the existence of payer-payee-bank interactions solely through the protocol.
>
> **P6.** The bank cannot infer any event-specific details.
>
> **P7.** Neither the payer nor the payee should be able to prove to outside parties that they interacted in a conditional payment protocol (deniability).

2.1   Operational Model

Let A be the payer, C the payee, B the bank and T the trusted publisher. Factoring large composite numbers is hard. There exists a PKI infrastructure based on RSA. For any party X, we denote by $id(X)$ its identity, N_X its public RSA modulus, e_X its public key and d_X its private key. Network anonymizers [17] exist and can be deployed by both A and C to communicate to B. Let Mix be a notation for such an anonymizer. Whenever possible point-to-point communication will be encrypted semantically secure [3], including links passing through an anonymizer towards the bank. These will be encrypted with no forward security by using a session key generated by the anonymous party (e.g., C, when communicating with B). The meaning of all messages in the system is explicited as part of the message; we will not detail this in the protocols. The bank B manages client accounts and assists clients by generating or cashing traditional and conditional e-cash payments.

Let b denote the public "name" of the considered future event. Let t be the corresponding secret published by T in the case of a favorable (for payment) outcome. Without loss of generality we will consider b to be a large prime number, and $t = b^{-1} \ mod \ \phi(N_T)$, where $\phi()$ is Euler's totient (this is discussed further

---

[3] With keys being generated using authenticated DH or equivalent.

in Section 4.3). If the event's outcome is not favorable, T is trusted to immediately discard any information that could enable other parties to reconstruct t or portions thereof. We stress it is important for T to not collude with the payee to reveal the payer's identity by publishing t and allowing C to cause a payer double-spending condition. The publishing process of T could be as simple as maintaining an authenticated website. For scalability, outside of the publishing process, no interaction between T and other participants is required by the protocol.

2.2   Adversary

As discussed above we are concerned with a computationally bounded adversary. Because the message exchanges are encrypted, and the protocol only uses anonymizers when no authentication is required, we will consider here mainly the insider threat. Both the bank and the publisher should not be able to infer any additional information about ongoing or past conditional payment transactions. Specifically, without their direct cooperation, A and C should not be identifiable as conditional payment partners. Additionally, no subset of participants should be able to collude and violate any of the properties above.

2.3   Crypto Tools

For completeness, we will briefly discuss blind signature protocols.

Let a party A engage in a blind signature protocol with B (B is the signing party). At the end of a correct run of the protocol, A will be in the possession of a "well-formed" (e.g., "$10") message signed by B, such that B does not know the message contents but is (sufficiently) confident of its "well-formed"-ness. It can be considered that B's signature semantics in fact speak only about the fact that the message is "well formed". Thus, the "blind" signature should not be interpreted to mean anything else. We now overview an instance, namely the **cut-and-choose** protocol [12,13,14,15].

Let $S_B(M)$ denote B's signature on message M. A generates n "well-formed" messages $\{M_1, \dots, M_n\}$, such that any of them signed by B (i.e., any of $\{S_B(M_1), \dots, S_B(M_n)\}$) would satisfy A as an end-result. A "blinds" all n messages with different blinding factors and sends them to B. A blinded message cannot be read unless the corresponding blinding factor is known. B requests $n-1$ randomly chosen blinding factors from A. It un-blinds the corresponding messages and verifies that they are "well formed". B is now convinced that with probability $1 - 1/n$, the remaining message is also well formed. By making n arbitrarily high, this confidence can also be made sufficiently high. B then signs the remaining blinded message M_j and sends it back to A, who simply un-blinds it. The blinding mechanism is designed such that a message first blinded by A, then signed by B, can be transformed into its simple signed (un-blinded) corresponding message $S_B(M_j)$ by A, knowing the blinding factor. We say that B blindly signed M_j for A.

For illustration purposes, we consider B's signature to be simple RSA exponentiation with private key d_B. The blinding mechanism of a message M can

then be $M \times s^{e_B}$. The corresponding un-blinding process is simply division by the blinding factor s. We note that blind signature protocols can be run through anonymizers (with simple precautions).

3   Related Work

Prediction Markets. Prediction markets generate assets whose value is conditioned by specific events. Example markets include the Iowa Electronic Markets (IEM) [2], Intrade [1], and TradeSports [6]. IEM is an educational prediction market of University of Iowa, based on real money, where payoffs are based on real-world events such as political or economic outcomes. Intrade and Trade-Sports allow their members to speculate for real money on the outcome of a multitude of future events, ranging from politics to sports and pop culture.

Companies such as Hewlett-Packard, Eli Lilly, Microsoft and Google use internal prediction markets, where employees trade futures contracts on sales and profits, success of products or supplier behavior [21,24]. The Iowa Health Prediction Market [3] attempts to forecast the future activity of a wide variety of infectious diseases and related phenomena, by using the unique and fresh knowledge of health-care workers. University of Miami released a Hurricane Futures Market in an attempt to better understand the information that people rely on when forecasting hurricanes.

Conditional payments will enable novel applications for prediction makers and companies with an interest in future outcomes of events. Prediction makers can receive rewards for accurate predictions, while allowing companies to purchase safety for important decisions.

Time Release Encryption. Dodis and Yum [18] introduce a novel problem called the *time capsule signature*. It allows for the construction of a signature that becomes valid at a time in the future when a trusted third party publishes a trapdoor associated with the current time. The time capsule signature allows the recipient of the signature to immediately verify its validity. Moreover, the third party has no interaction with the generator or recipient of the signature. It may seem possible to use the time capsule signature to solve the conditional payment problem. The payer could ask the bank to generate a time capsule signature on a blinded e-cash such that the capsule can be removed only if a certain event occurs. Besides the technical difficulty of the payer un-blinding the time capsule, this solution would require the bank's knowledge of the event, its publishing procedure and ultimately the identity of the publishing institution. However, for privacy reasons, the conditional payment problem requires the decoupling of the publishing institution from all other participants. In particular, the bank's operation should be oblivious of the nature of the event determining the condition.

Blake and Chan [9] propose a protocol for transferring time-encrypted messages between users. A message becomes valid only after a trusted server publishes a signed piece of information on a specific time value. Their solution requires no interaction between the trusted server and the users and also preserves

the user's privacy from the server. Cathalo et al. [11] propose a more efficient solution for this problem, that also improves the user's anonymity. However, none of these schemes allows the receiver of a timed release message to verify its validity before release time, making them unsuitable for conditional e-cash transfers.

E-cash. The use of blind signatures and of the cut-and-choose protocol for providing untraceable electronic cash payments was proposed in [12,13,14,15]. Franklin and Yung [20] proposed the use of a trusted entity (trustee) that collaborates with the bank at withdrawal and deposit to provide a computation efficient on-line cash system. Trustees (either on-line or off-line) were proposed to provide variable degrees of anonymity for e-cash [22,16,10,19]. Stadler et al. [22] introduced the notion of coin tracing and introduced several tracing mechanisms, requiring the trustee to be on-line at withdrawal. Camenisch et al. [10], Frankel et al. [19] and Davida et al. [16] proposed payer and coin tracing mechanisms using off-line trustees. In our work however, the payer and payee anonymity is essential and requires the bank to be unable to link the payer and payee even when colluding with one of them.

Simon [26] proposes a simple e-cash protocol in a network where anonymous communication is possible. The payer generates the e-cash by having the bank sign $f(x)$ where x is the payer's secret and f is a one-way function. The e-cash can be transferred by revealing x to the payee. The payee can then either cash the money with the bank or further transfer it by providing the bank with x and asking it to sign $f(y)$ for which it knows y. If the communication between the payee and the bank is anonymous, the payee remains anonymous and can transfer the money further. The bank can link the start and end points of a transfer chain, however, for long chains this information may be meaningless. Moreover, the end point of a transfer chain may repeat this protocol with itself, to artificially increase the length of the chain. Even though we also require the use of anonymizers, the solution of [26] does not provide support for conditional transfers. Even if conditional transfers would be provided, the payer could easily spend the e-cash transferred to the payee before the condition is satisfied – as the e-cash does not encode any information about the payer for anonymity reasons.

4   Conditional Anonymous Payments

The solution is composed of a set of logical sub-components: the generation of conditional payments, the validated transfer of the payments from the payer to the payee, and their spending by the payee in the case of a successful event outcome, or the cashing of the un-spent payments by the payer otherwise. All the above will also be designed to prevent double spending by both the payer and the payee. In the following we detail each of these components.

4.1   Payment Generation (PG)

Let n_1 and n_2 be security parameters. To generate the conditional payment, the payer A will contact the bank B as follows (A holds an account with B).

A generates $2n_1$ random numbers $X_1, .., X_{n_1}$ and $R_1, .., R_{n_1}$. Using a standard secret splitting algorithm [25], A constructs n_2 shares for each of the values $X_i \oplus id(A)$, for $i = 1..n_1$. We denote the j-th share corresponding to $X_i \oplus id(A)$ by $share_{ij}$ for $j = 1..n_2$. For any $X_i \oplus id(A)$, all its n_2 shares are required for its correct reconstruction.

A then constructs n_1 blocks, each of $n_2 + 1$ fields. The i-th block consists of

$$m_{iL} = [id(A), X_i, R_i, v, "left"], \quad m_{ijR} = [share_{ij}, R_i, v, "right"],$$

where v represents the value and currency of the payment (i.e. \$1). "$left$" and "$right$" are text messages used to differentiate between the m_{iL} value and the m_{ijR} shares.

Next, A asks the bank to blindly sign one of the n_1 message pairs using the cut-and-chose protocol discussed in Section 2.3. In this specific case however, the bank signature consists of a signature on both m_{iL}, and m_{ijR} as well as on each and every $share_{i}j$ in m_{ijR}. The bank will do so after verifying "well formed"-ness of $n_1 - 1$ random pairs as well as their associated shares. Specifically, the bank will verify

- that each set of n_2 shares in the $n_1 - 1$ "right" messages m_{ijR} can be used to reconstruct the corresponding $X_i \oplus id(A)$ values.
- that XOR-ing these reconstructed values $X_i \oplus id(A)$ with the second fields of m_{iL} yield indeed $id(A)$.
- that the third field of m_{iL} is equal to the second field of m_{ijR}. This value, R_i associates the messages later on.
- the correctness of the enclosed currency value (v).

If any check fails, B aborts the protocol. Otherwise, A's account is debited in an amount of v and A is able to retrieve (after un-blinding) the following payment document (signed by the bank B):

$$M_L = m_{lL}^{d_B} \bmod N_B, \quad M_{jR} = m_{ljR}^{d_B} \bmod N_B,$$

where $j = 1..n_2$ and $l \in [1, n_1]$ was randomly chosen by B.

Intuitively, M_L can be later used by A to cash any un-spent payment in the case of an un-successful event outcome (see Section 4.4), while the n_2 bank signed e-cash shares, M_{jR}, can be used by A for payments to potential payees such as C (see Section 4.3).

4.2 Preventing Double Spending (PDS)

Before we proceed with describing the actual transfer of these shares to payees, we will first discuss a simple token attribution mechanism designed as one of the tools we will use to prevent the payer from double spending. Specifically, A will be prevented from transferring the payment to more than one payee. Moreover, at the completion of this step, at most two participants, one being A, will be able to cash the payment.

To achieve this, B will issue two unique "use tokens" for each signed payment (identified so anonymously by its unique R_l value). Each of these tokens will be issued on-demand, in an online interactive protocol, through an anonymizer. Specifically, before interacting with C but after retrieving the signed payment document $\{M_L, M_{jR}\}$ from B, A will use the anonymizer Mix to send B the currency amount v and the R_l value occurring both in M_L and M_{jR}, $j = 1..n_2$. B will respond with a fresh random token $token_L$. B will also store an association between R_l and this token $R_l : \{token_L\}$ for future reference. We call the payment "activated" once this happens. If B has already seen R_l it ignores the message.

Before transferring the actual payment document, A sends R_l and v to C. C then forwards R_l anonymously to B who proceeds as follows:

- if B does not find any record of R_l it notifies C and then simply ignores the message as the payment has not been activated yet.
- otherwise, if R_l is associated with a *single* token $token_L$, B generates a new random token $token_R$, associates it also with R_l ($R_l : \{token_L, token_R\}$), and sends it back to C (through the Mix). It is important to note that only C and B know $token_R$. C will use $token_R$ later to cash the payment upon a successful event outcome, as will be discussed later.
- if B already stores *two* tokens associated with R_l, it notifies C, who in turn then aborts the protocol, knowing that A attempts to double spend.

4.3   Conditional Transfer (CT)

The PDS protocol above allows C to assert the fact that the payment that will follow from A has been activated and has not yet been spent. In this section we discuss achieving the "conditional" properties of the protocol. We introduce here a randomized probabilistic solution.

The main idea is for A to generate a quantity that can both (i) convince C to accept this payment because it is indeed valid cash-able money signed by B, (ii) allow its cashing only if t is published by T. A uses event b and T's modulus N_T (see Section 2) to blind each $M_{jR} = m_{ljR}^{d_B} \bmod N_B$, $j = 1..n_2$, separately, by computing

$$S_j = M_{jR}^b \bmod N_T.$$

A and C then engage in a cut-and-chose protocol (see Section 2.3) through which C becomes convinced that with $1 - 1/n_2$ probability, all of the S_j values are indeed well formed and signed by B, as follows.

A sends all such S_j values to C, along with the R_l value and currency amount v. C selects a random one of them (e.g., S_u) and asks A to prove that all the remaining ones are indeed valid M_{jR} messages. To do so, A sends C all $M_{jR} = m_{ljR}^{d_B} \bmod N_B$ values for all $j \in [1, n_2] \setminus \{u\}$ and C can verify that indeed $S_j = M_{jR}^b \bmod N_T$ for these values.

At this point, C will verify the "well-formed"-ness of all revealed M_{jR} values. After removing B's signature from M_{jR}, C verifies that the fourth field of m_{ljR}

equals the constant string "right" and that the second and third fields equal the R_l and v values previously sent by A for the present transaction. This verification prevents A from re-using shares from different protocol instances. C also verifies that there are no duplicates among the first fields ($share_{lj}$) of the $n_2 - 1$ m_{ljR} values recovered. As a reminder, all n_2 shares are required for the reconstruction of the corresponding $X_i \oplus id(A)$ value later on. If any of these checks fails, C aborts the protocol.

Later, for a successful event outcome, T will publish

$$t = b^{-1} \ mod \ \phi(N_T)$$

Since b is prime (see Section 2), it has an inverse mod $\phi(N_T)$. Only T can compute this inverse, knowing the factorization of N_T. Using t, C can retrieve the missing M_{uR} value as

$$M_{uR} = S_u^t \ mod \ N_T$$

By removing B's signature from M_{uR}, C yield the last unknown share, $share_{lu}$, to construct the secret $X_l \oplus id(A)$. We next discuss the payment cashing procedure.

4.4   Spending the Money (SM)

In the case of a favorable event outcome, C should be able to interact with B and get her account credited appropriately. To achieve this, we propose a three-stage protocol. In the first stage C contacts B anonymously and provides proof of credit. In the second stage, C and B engage in a blind signature protocol (see Section 2.3) in which B blindly signs an un-traceable piece of currency of equivalent value to the credit that was proven in the first stage. In the final stage, the payee C directly contacts the bank B through an authenticated channel and exchanges this piece of currency for credit to her account. For an unfavorable event outcome, to cash an un-spent payment, A proceeds identically.

We note that, technically, the three-stage anonymous protocol is apparently superfluous here for purposes of providing anonymity, as this has already been ensured by previous anonymization and the lack of any information about A's identity in the proof of credit. Nevertheless, we chose to discuss it here for ease of presentation. Its purpose will become apparent later when we discuss specific applications of conditional payments such as online betting.

We now detail the above. C uses the anonymizer Mix to send to B the message

$$token_R, M_{jR}, j = 1..n_2,$$

containing the n_2 shares recovered from A and T. Similar to C (see Section 4.3), B immediately verifies the validity of each share M_{jR}. If at least one share does not verify, B aborts the protocol. Otherwise, it uses the shares to recover $X_l \oplus id(A)$. B then verifies that $token_R$ is the *second* token associated with the R_l value contained in all M_{jR} shares. If the check fails, B aborts the protocol.

Next, B investigates potential double spending. If the M_{jR} shares have been previously spent, it simply drops the message, knowing that C double spends.

If the left part of the payment, M_L has been spent (by A), B can immediately recover A's identity by computing the XOR of the first field of the corresponding m_{lL}, X_l with $X_l \oplus id(A)$.

At this point, B has proof to believe that C is entitled to a credit equal to the v value stored in the third field of M_{jR}. Now C and B can anonymously engage in a blind signature protocol in which B blindly signs an un-traceable temporary piece of uniquely identifiable currency of equivalent value to this credit.

Finally, the payee C directly contacts the bank B through an authenticated channel and exchanges this piece of currency for credit to her account. B will first verify if this currency has been already spent, credit C's account, and store the unique identifier of the currency for future double spending detection.

4.5   Analysis

In this section we informally discuss the security properties of the above protocol.

Double Spending (P2). The payer could try to double spend during the PDS step by registering with the bank the same e-cash under different R_l values and transferring each value to a different payee. This is prevented during the CT step, by having the payee verify that the R_l value encoded in the e-cash matches the R_l value received during the PDS step.

Alternately, during the SM step, the payer could try to spend her e-cash (using M_L) even in the case of a favorable outcome published by T. However, once the payee performs her SM step, the payer's identity will be immediately revealed. The payer could also try to spend the e-cash she sends to the payee, before the payee has a chance to do it. For this, the payer would have to obtain the $token_R$ value associated with the unique R_l of the e-cash, shared by the payee and the bank. If the payer retrieves $token_R$ from the bank before the payee, the payee will be unable to get it and will abort the protocol.

The payee cannot double spend, since both her shares (M_{jR}) and the unique identifier generated at the end of the SM step (see Section 4.4) are recorded by the bank. The payer and the payee could try to collaborate in order to double spend e-cash without having their identities revealed. This is prevented by the fact that the e-cash generated during the PG step ensures w.h.p. $(1 - 1/n_1)$ the fact that spending both M_L and the M_{jR} shares reveals the payer's identity. Moreover, both M_L and the M_{jR} shares can only be spent once.

Guaranteed Payment or Rollback (P3,P4).   During the cut-and-choose sequence of the CT step, the payee receives $n_2 - 1$ shares of its choice of the payee's e-cash. If event b occurs and the corresponding t value is published by T, the payee can recover the missing share and spend the e-cash. If event b does not occur, the payer is certain that the payee is unable to recover the e-cash. The payer can then safely cash back its payment, without fear of double spending. At this point T is trusted to never reveal the factoring of the current N_T value. We stressed before the existence of a collusion vulnerability: T can collude with the payee to reveal the payer's identity by publishing t and allowing C to cause a payer double-spending condition.

Un-linkability and Deniability (P1,P5,P7). The payer obtains the payment signed by the bank, containing a R_l value that is unknown to the bank. Moreover, the payee cannot prove payment origin to other parties as no non-repudiable identification tokens are revealed in any steps outside of double spending. This prevents the bank from colluding with payees to trace payments to their payer.

The payer could collaborate with the bank and attempt to reveal the identity of the payee. To achieve this, the payer could spend her e-cash (M_L) or the payee's e-cash (the M_{jR} shares) in order to signal the bank the moment when her e-cash will be spent by the payee. However, before spending the e-cash in person, the payee performs two additional stages, both through an anonymizer (see Section 4.4). The second additional stage generates the anonymous e-cash the payee will spend then in person.

Since the publisher does not directly interact with any participants, except possibly for publishing event outcomes, property P5 is trivially satisfied.

5   Applications

In this section we briefly overview just a few of the application scenarios requiring conditional e-cash payments: financial securities, prediction markets, and anonymous online betting.

5.1   Securities Trading

A particularly relevant application scenario for conditional payments can be found in trade systems involving (atomic) securities. Securities are financial instruments that deliver future value as a function of event outcomes. A simple illustrative instance is the following contract:

> "The Smart Financial Group will pay the bearer of this certificate $50 at the end of the current financial year, if and only if the DOW Jones will increase by 5%."

Financial institutions can now sell such securities online with full privacy and assurances of payment for their clients.

5.2   Prediction Markets

Yet another application for conditional payments is in prediction markets [2,4, 1,6,5]. For example, manufacturers may use futures markets to direct investments. Additionally, a sense of confidence can be gained if conditional monetary transactions are involved. A prediction maker can express its confidence in a prediction by associating a payment to the manufacturer that is to take place if the outcome of the prediction is unfavorable. In return, the manufacturer agrees to reward the prediction maker if the outcome of the prediction is favorable.

For instance, the Smart Motors Company (SM) may propose the following trade to any willing prediction maker:

> "If crude oil is traded at under \$60 a barrel until the end of 2007,
> the Smart Motors Company will pay \$6. If the price goes above
> \$60, SM will be paid \$10. No money changes hands now."

SM and a prediction maker may sign as many of such contracts as they desire.

Manufacturers and prediction makers signing such contracts online are now able to preserve their interactions private, even from the financial institution handling the money. This is important in cases where manufacturers want to hide certain decisions from the competition and where prediction makers may posses insider information.

5.3   Online Betting

Interestingly, the conditional payment mechanisms discussed here can be deployed in the design of anonymous online betting protocols. We briefly outline how.

Without loss of generality, we will consider A as being the betting party and C the "bookie" (the party taking bets). Then, a simple online betting protocol can be constructed as a symmetrical conditional payment scenario. For example, A will provide a conditional (on a certain race outcome) \$1 to C, while C will reciprocate with \$10 conditional on the negated outcome. The race organizer T will publish different t values t_{win} and t_{lose} for a win or a loss respectively.

Even though the payments sent are conditional, either C or A may choose not to reciprocate if the other party sends its payment first. One simple (yet more costly) solution to address this issue is to break each payment into multiple smaller payments. For instance, for a 2:1 bet for \$100, A may initiate a 10 step protocol, by sending C a \$10 conditional payment. A then waits to receive a \$20 conditional payment from C before sending the next payment. While imposing a larger communication overhead, this ensures that no participant may loose more than 1/10th of the expected value. We also designed a few lower-overhead solutions (of increased exposition complexity) we will not discuss here.

Full Anonymity. The above solution provides a simple betting protocol geared towards achieving anonymity of both C and A with respect to B or T. Often however, online betting protocols would benefit from one additional property, namely full anonymity:

> **P8.** The payer and payee should not be required to know each
> other's identities nor should they be able to infer these identities
> from the betting protocol.

This is particularly important in hostile environments with concerns of collusion (of either C or A) with outside parties with incentives to reveal participation in the protocol of either the better or the bookie.

To achieve full anonymity we will require the interaction between A and C to be performed either through a special anonymous IP rendez-vous point, similar to the ones in Tor [17] or through IRC channels as follows.

C anonymously advertises its public key as well as the service it provides. *C* also registers its public key along with several introduction points in a lookup service (built to be censorship resilient [27]).

A finds the advertisements and then uses the lookup service to retrieve the introduction points of the bookie. It then chooses an anonymous rendez-vous point as the place where the transaction is to take place and registers its coordinates (encrypted with the public key of the bookie) on one or several of the introduction points. If the bookie decides to accept the better, it retrieves the bet anonymously from the rendez-vous point while it reciprocates with its own conditional payment or engages in a more complex multi-step *simultaneous* payment protocol as above.

A simpler idea is to use IRC channels and messages steganographed into posted media files to also achieve plausible deniability of participation claims in the case of compromised rendez-vous points.

6   Conclusions

In this paper we introduced a novel conditional payment protocol that allows future anonymous cashing of bank-issued e-money only upon the satisfaction of an agreed-upon public condition. We discussed a set of application scenarios including online trading of financial securities, prediction markets, and betting systems.

In future work we believe it is important to allow payees to further transfer their payment tokens to third parties. This is of interest for example in financial securities/options trading where securities and options are subject to multiple sell-buy cycles before maturation. It would also be interesting to pursue events with more complex, non-binary outcomes, e.g., a boolean formula of multiple variables. Additionally, we are currently working on lower overhead methods to enable conditional transfers. We have designed a few solutions based on bilinear maps that seem particularly promising.

References

1. Intrade: A trade exchange network company, http://www.intrade.com/
2. Iowa electronic markets, http://www.biz.uiowa.edu/iem/
3. Iowa health prediction market, http://fluweb.biz.uiowa.edu/fluhome/index.html
4. Newsfutures, http://us.newsfutures.com/home/home.html
5. Strategy page, http://www.strategypage.com/prediction_market/default.asp
6. Tradesports, http://www.tradesports.com/
7. Arrow, K.J., Debreu, G.: The existence of an equilibrium for a competitive economy. Econometrica 22 (1954)
8. Balasko, Y.: Foundations of the Theory of General Equilibrium (1986)
9. Blake, I.F., Chan, A.C.F.: Scalable, server-passive, user-anonymous timed release cryptography. In: ICDCS 2005. Proceedings of the 25th IEEE International Conference on Distributed Computing Systems, Washington, DC, USA, pp. 504–513. IEEE Computer Society, Los Alamitos (2005)

10. Camenisch, J., Maurer, U.M., Stadler, M.: Digital payment systems with passive anonymity-revoking trustees. In: Martella, G., Kurth, H., Montolivo, E., Bertino, E. (eds.) ESORICS 1996. LNCS, vol. 1146, pp. 33–43. Springer, Heidelberg (1996)
11. Cathalo, J., Libert, B., Quisquater, J.-J.: Efficient and non-interactive timed-release encryption. In: Qing, S., Mao, W., Lopez, J., Wang, G. (eds.) ICICS 2005. LNCS, vol. 3783, pp. 291–303. Springer, Heidelberg (2005)
12. Chaum, D.: Blind signatures for untraceable payments. In: Advances in Cryptology— Proceedings of Crypto 1982, pp. 199–203. Plenum Press (1982)
13. Chaum, D.: Security without identification: transaction systems to make big brother obsolete. Communications of the ACM 28(10), 1030–1044 (1985)
14. Chaum, D.: Privacy protected payments: Unconditional payer and/or payee untraceability. In: Proceedings of SmartCard 2000 (1988)
15. Chaum, D., Fiat, A., Naor, M.: Untraceable electronic cash. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 319–327. Springer, Heidelberg (1990)
16. Davida, G.I., Frankel, Y., Tsiounis, Y., Yung, M.: Anonymity control in e-cash systems. In: FC 1997. LNCS, vol. 1318, pp. 1–16. Springer, Heidelberg (1997)
17. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium (2004)
18. Dodis, Y., Yum, D.H.: Time capsule signature. In: Patrick, A.S., Yung, M. (eds.) FC 2005. LNCS, vol. 3570, pp. 57–71. Springer, Heidelberg (2005)
19. Frankel, Y., Tsiounis, Y., Yung, M.: "indirect discourse proof": Achieving efficient fair off-line e-cash. In: Kim, K.-c., Matsumoto, T. (eds.) ASIACRYPT 1996. LNCS, vol. 1163, pp. 286–300. Springer, Heidelberg (1996)
20. Franklin, M.K., Yung, M.: Secure and efficient off-line digital money (extended abstract). In: Lingas, A., Carlsson, S., Karlsson, R. (eds.) ICALP 1993. LNCS, vol. 700, pp. 265–276. Springer, Heidelberg (1993)
21. Kiviat, B.: The end of management? Time - Inside Business, http://www.time.com/time/insidebiz/printout/0,8816,1101040712-660965,00.html
22. Stadler, M., Piveteau, J.-M, Camenisch, J.: Fair blind signatures. In: Guillou, L.C., Quisquater, J.-J. (eds.) EUROCRYPT 1995. LNCS, vol. 921, pp. 209–219. Springer, Heidelberg (1995)
23. Samuelson, P.A.: Foundations of Economic Analysis. Harvard University Press (1947)
24. Saporito, B.: Place your bets! Time, http://www.time.com/time/magazine/article/0,9171,1118373,00.html
25. Schneier, B.: Applied Cryptography: Protocols, Algorithms and Source Code in C. Wiley & Sons, Chichester (1996)
26. Simon, D.R.: Anonymous communication and anonymous cash. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 61–73. Springer, Heidelberg (1996)
27. Waldman, M., Rubin, A.D., Cranor, L.F.: Publius: A robust, tamper-evident, censorship-resistant, web publishing system. In: Proc. 9th USENIX Security Symposium, pp. 59–72 (August 2000)

A Privacy-Protecting Multi-Coupon Scheme with Stronger Protection Against Splitting

Liqun Chen[1], Alberto N. Escalante B.[2], Hans Löhr[2],
Mark Manulis[2], and Ahmad-Reza Sadeghi[2]

[1] HP Laboratories
liqun.chen@hp.com
[2] Horst Görtz Institute, Ruhr-University of Bochum, Germany
{eban,hloehr,sadeghi}@crypto.rub.de, mark.manulis@rub.de

Abstract. A multi-coupon (MC) represents a collection of k coupons that a user can redeem to a vendor in exchange for some goods or services. Nguyen (FC 2006), deepening the ideas of Chen et al. (FC 2005), introduced an unforgeable privacy-protecting MC system with constant complexity for issuing and redemption of MCs, that discourages sharing of coupons through a property called *weak unsplittability*, where sharing of a single coupon implies sharing of the whole multi-coupon (all-or-nothing sharing). Both schemes still lack some features required by many applications in practice, and also stronger forms of unsplittability are desirable. In this paper, we propose a new security model for MC systems with stronger definitions, followed by a concrete realization where single coupons within a MC may represent different goods or services, have independent validity periods, and must be redeemed sequentially ensuring a stronger version of unsplittability compared to all-or-nothing sharing. The complexity of the proposed scheme is linear in k for the generation of multi-coupons and constant for each redeemed single coupon.

Keywords: Coupon, privacy, unsplittability, unlinkability, loyalty.

1 Introduction

Paper-based coupon schemes are successfully used by enterprises for various marketing purposes like providing discounts, increasing sales within a period of time (via coupons with some specified validity period), setting up prepayment models, attracting new customers, and establishing long-term relationships (loyalty) with them. From an abstract point of view, a *coupon* is some information that gives a customer the right to claim a good or service from a vendor.

The procedure in which a vendor provides a customer with a new coupon is called *issue*. The procedure in which the customer pays using the obtained coupon is called *redeem*. Here, the vendor verifies that the coupon is valid and authentic, and provides the customer with the specified good or service. Coupons can be used only once. In the following, we denote by *object* the good or service implied by a coupon. Any item that can be bought may become an object in

practice, e.g. cloths, songs, books, videos, medicines, tickets, and even immaterial services: discounts, access to computer resources or facilities, etc.

In contrast to widely used paper-based coupon schemes, *electronic coupons* (*e-coupons*) have gained acceptance relatively slowly [14], and are still waiting for their breakthrough. One of the reasons for this development is insufficient security of available schemes. A *multi-coupon* (*MC*) [7,11] denotes a collection of e-coupons that is handled as a single unit.

In this paper we consider a *multi-coupon scheme* (*MCS*) that protects the privacy of the customers, and encourages loyalty of clients by providing *unsplittability* [7], i.e., two users cannot redeem coupons from the same *MC* separately and independently. Consider prepaid-goods, where a vendor, hoping for a long-term client relation, sells many goods at once at a cheaper price compared to that of separately sold goods. In this case sharing would allow a group of users to buy a single *MC*, and obtain goods at a subsided price, but without giving loyalty in return. We focus on a basic *MC* framework where the only involved parties are many customers (users) and a single vendor. Note that other frameworks are imaginable, e.g., with several cooperating vendors.

From the security point of view, threats in *MC* systems are different from those in paper-based coupon systems. First, it is very easy to create a perfect (digital) copy of an electronic *MC*, whereas copying a paper-based booklet requires much higher effort. Second, when dealing with a *MCS* we must also consider attacks in which different users collude and attempt to cheat the vendor. Moreover, in the digital world privacy and anonymity of customers becomes more important since the vendor may try to infer and store additional information about them including purchase habits, gender, age, etc. This would harm privacy and allow client profiling and price discrimination [13], e.g., different customers are offered the same goods by the same vendor, but at different prices.

1.1   Desired Security Properties

We focus on *unforgeability*, *unlinkability*, and *unsplittability* because, as pointed out in [6,7,11], these are the essential properties of a *MCS*.

Unforgeability. There is an intrinsic monetary value associated to any coupon, explicitly or implicitly. Therefore, vendors want their multi-coupons to be *unforgeable*, in the sense that no coalition of users should be able to redeem more coupons than it has been rightfully allowed.

Unlinkability. It must be infeasible for a vendor to link a redeem procedure for a customer to the corresponding issue procedure, or to link two different redeem procedures with the same customer. This implies anonymity of customers.

Unsplittability. *Weak unsplittability* (WU) [7], also known as *all-or-nothing sharing*, intuitively, requires that whenever a user intends to share a single coupon with a second user, she has to provide her with *all* the secret information related to the involved *MC*. This, however, would make possible the complete redemption of the *MC* by the second user. Thus, in case that both users do not trust each other, WU discourages sharing.

A stronger version, called *(ordinary) unsplittability*, requires that it is infeasible for an adversary to produce more *autonomous* redemption algorithms than the number of multi-coupons he has rightfully obtained, where by autonomous we mean that such algorithms do not share any information gained during the redemption. In other words, if a user gives a single coupon to another user, then that second user has to send back some information to the first user after redeeming; otherwise the first user cannot spend further coupons from that multi-coupon. Hence, sharing is more cumbersome with this stronger version of unsplittability than with weak unsplittability because it requires a trust relationship and additional interaction between the users.

Contribution and Organization. We start in Section 2 with the description of related work on multi-coupon schemes, and give a brief overview of our construction in Section 3. In Section 4, we define the syntax and correctness of a *MCS*, and propose a more precise security model for *MCS*s that includes a stronger form of unsplittability without relying on all-or-nothing sharing. Thereafter, we propose in Section 5 a construction of a privacy-protecting *MCS* which satisfies our stronger requirements and provides additional features for practical applications, e.g., different objects for individual coupons within one multi-coupon and validity periods thereof. Redeem complexity (both computation and communication) is constant w.r.t. the size k of the multi-coupon (i.e., the number of coupons it contains), and complexity of the protocol for issuing multi-coupons is linear in k, which is the best we can get when each coupon has individual attributes. Additionally, we prove the security of our scheme w.r.t. the proposed security model. Finally, we provide in Section 6 some insights into possible improvements and future work.

2   Related Work

Syverson et al. [17] introduced the concept of unsplittability in the context of unlinkable serial transactions to discourage sharing, and suggested an extension of their scheme to implement coupon books. Later, Chen et al. [7] described the properties that a privacy-protecting multi-coupon system must provide, justified the use of unsplittability over other means to discourage sharing (e.g., hiding credit card numbers in the multi-coupons), and proposed an unforgeable, unlinkable, and weakly unsplittable scheme. However, their construction is less practical because of an expensive proof of knowledge used in the redemption, whose complexity is linear in k (i.e., the number of coupons in the multi-coupon).

More recently, Nguyen [11] addressed some disadvantages of [7], and defined a security model for *MCS*s, followed by an efficient construction based on a verifiable pseudorandom function and bilinear groups. Its issue and redeem complexity is constant w.r.t. k, it offers the same security properties as in [7], and adds a new feature to *revoke* multi-coupons. It is arguable whether revocation is indeed necessary for a *MCS*, since in real life it is unusual that a vendor revokes issued coupon booklets, and this operation might be costly.

One drawback of both above mentioned schemes is that every issued multi-coupon must contain the same number of coupons, i.e., k is a system parameter fixed for all multi-coupons. This limitation, as pointed out in [11], can be overcome in both schemes by extending the issue protocol. However, this extension is impractical, i.e., for [11] a term $k - m'$ is added to the complexity of the issue protocol, where m' $(0 < m' < k)$ is the number of issued single coupons. Another drawback of these schemes is that there is no concept of coupon's object (or coupon's type [6]). Hence, all coupons are valid for the same purpose.

As previously explained in [7,11], most related schemes (e.g., e-cash, digital credentials) cannot be employed as privacy-protecting unsplittable *MCS*s because they have different usage patterns [15,1], are inefficient in this setup [12], or lack at least one of the required properties [3], in particular unsplittability. Some e-cash systems can be used as unlinkable or at least anonymous *MCS*s (e.g. [4,6]). However, they are (unintentionally) at most weakly unsplittable.

3   Short Overview of Our Construction

In our scheme, each single *redeemable* coupon $(id, ob, sq, \sigma, \sigma')$ is specified by a coupon identifier id, a coupon sequence number sq, a coupon's object ob (i.e., the good or service represented by the coupon[1]), a signature σ on the tuple (id, ob, sq), and a signature σ' on sq. A coupon is not redeemable if it lacks σ'.

A multi-coupon M of size k is a list of k single coupons with consecutive sequence numbers, where at least the first coupon must be redeemable. In the issue protocol, the user obtains a multi-coupon where the coupon identifiers are kept private by the user, and all other attributes are known to both user and vendor. After the issue procedure, only the first coupon is redeemable, but every coupon has a valid signature σ_i, for $0 \leq i < k$. During the redemption of the i-th single coupon with sequence number sq_i, the user obtains a signature σ'_{i+1} on the sequence number $sq_i + 1$, and hence the next coupon in the list becomes redeemable. In order to redeem a coupon, the user must prove that the coupon has never been used before (by disclosing id), and that it is indeed redeemable (by proving that σ is a valid signature on id, ob and sq, and that σ' is a valid signature on sq).

Informally, the vendor's knowledge about elements of a single coupon depends on the actual procedure, i.e., id is hidden during the issue protocol, and disclosed to the vendor during redemption; sq, σ, σ' are known to the vendor during issuing, but hidden during the redeem protocol; ob is known to the vendor during both the issue and the redeem protocols.

Our scheme utilizes a digital signature scheme with efficient protocols that allows to obtain a signature on a (partially) blinded tuple (i.e., some elements of the tuple are disclosed, while others are only committed to), and to prove the knowledge of a signature on a (partially) blinded tuple without disclosing any useful information, other than the fact that the signature is valid.

---

[1] The vendor must publish an official coding of coupon's objects as integers.

4 Security Framework for Multi-coupon Schemes

Notation. For a finite set $\mathcal{S}$, $s \in_R \mathcal{S}$ denotes the assignment to the variable s of an element uniformly sampled from $\mathcal{S}$. Let A be a probabilistic algorithm. By $\text{out}_A \leftarrow A(\text{in}_A)$ we denote that the variable out_A is assigned the output of A's execution on input in_A. We denote by $(A(\text{in}_A), B(\text{in}_B))$ a pair of interactive algorithms with private inputs in_A and in_B, respectively, and write $(\text{out}_A, \text{out}_B) \leftarrow (A(\text{in}_A), B(\text{in}_B))$ to denote the assignment of A's and B's private outputs after their interaction to the variables out_A and out_B, respectively.

4.1 General Multi-coupon Schemes

We consider a basic framework where the participants are a single vendor $\mathcal{V}$ and a collection of users $\mathcal{U}_i$. The following definition is general in that it does not account for specific coupon features such as revocation, coupon objects, or validity periods. We will refer to any particular user simply by $\mathcal{U}$.

Definition 1 (Multi-Coupon Scheme). *A* multi-coupon scheme *(MCS) consists of a set of protocols:* {$\texttt{Setup}$, $\texttt{Issue}$, *and* $\texttt{Redeem}$}*, which are specified by the following algorithms.*

$\texttt{Setup}$ algorithm. $(PK, SK) \leftarrow \texttt{Setup}(1^\kappa)$ is the initialization algorithm executed by the vendor once to generate one instance of the multi-coupon scheme. It takes as input the security parameter κ, and outputs a public key PK (which from now on we assume to include the security parameter κ coded in unary, and a system parameter $k_{\max}$ representing the maximum allowed number of coupons per MC), and a secret key SK (which might include PK).

$\texttt{Issue}$ protocol. In order to obtain a MC with k coupons, $\mathcal{U}$ performs the following protocol with $\mathcal{V}$: $((res_u, M), res_v) \leftarrow (\texttt{Issue}_u(k, PK), \texttt{Issue}_v(k, SK))$, where, from now on, the subindices u and v denote user and vendor algorithms, respectively. The output flags res_u, $res_v \in \{acc, rej\}$ indicate success or failure according to the user or vendor, resp. $\texttt{Issue}_u$ outputs the flag res_u and a multi-coupon M, whereas $\texttt{Issue}_v$ only outputs the flag res_v.

$\texttt{Redeem}$ protocol. After $\mathcal{U}$ has obtained the multi-coupon M she redeems it to $\mathcal{V}$ by performing the protocol $((res_u, M'), (res_v, \varsigma')) \leftarrow (\texttt{Redeem}_u(M, PK), \texttt{Redeem}_v(\varsigma, SK))$. $\texttt{Redeem}_u$ outputs an updated multi-coupon M', and a flag res_u just like in issue, and $\texttt{Redeem}_v$ outputs a new vendor's internal state ς', which is initially set to the empty string, and a flag res_v.

The correctness requirement states that an honest user who obtains a MC from a fresh honest vendor must be able to redeem all the coupons it contains.

Definition 2 (Correctness). *A multi-coupon scheme is* correct *if the following experiment returns true with overwhelming probability (for any $k \in [1, k_{\max}]$), where resIs and resRe are the output flags of the issue and redeem algorithms, respectively, and ς_i is the vendor's state, which is updated after each redemption.*

$(PK, SK) \leftarrow \texttt{Setup}(1^\kappa); \varsigma_1 \leftarrow \varepsilon;$

$((resIs_u, M_1), resIs_v) \leftarrow (\texttt{Issue}_u(k, PK), \texttt{Issue}_v(k, SK));$

for $i = 1$ to k do:

$\quad ((resRe_u^i, M_{i+1}), (resRe_v^i, \varsigma_{i+1})) \leftarrow (\texttt{Redeem}_u(M_i, PK), \texttt{Redeem}_v(\varsigma_i, SK));$

if $(resIs_u, resIs_v, resRe_u^1, resRe_v^1, \ldots, resRe_u^k, resRe_v^k) = (acc, \ldots, acc)$

$\quad$ return *true*; else return *false*;

4.2 Adversarial Model and Security Requirements

In this section we present a solid security framework that covers a wide range of adversarial actions. We begin by defining the queries available to the adversary, and then we define the security requirements.

An adversary is a p.p.t. algorithm $\mathcal{A}$, which can play the role of, either, a vendor and a group of users, or only of a group of users. $\mathcal{A}$ can interact with the other participants through a set of queries, which cannot be interleaved.[2] Wlog we let the adversary be specified by a sequence of algorithms (e.g. $\mathcal{A} := (A_1, A_2, A_3)$). Honest parties are assumed to communicate over secure channels.

Depending on the degree of independence from the adversary, we consider two types of users: *scheduled* and *corrupted users*. Users belonging to the set of scheduled users ($\mathcal{SU}$) execute honest algorithms if requested by the adversary, but remain honest otherwise. The adversary has full control over the corrupted users, grouped in the set $\mathcal{CU}$, and is provided with their previous protocol views. Additionally, the adversary might act as a group of malicious users.

Similar to [9], we allow the adversary to interact with the system through a set of queries handled by an *interface*, which partially simulates the *MCS*, executes protocols with the adversary, and records certain user's or vendor's activities. The queries available to an adversary differ depending on whether he is playing the vendor's role or only a user coalition. We distinguish between two types of interfaces. The first interface (I_1) is employed to model a *MCS* facing a collusion of users, and is used to define unforgeability, and unsplittability. The second interface (I_2) models a *MCS* controlled by a malicious vendor, and is only employed to define unlinkability.

Interface 1 (I_1). In this case the adversary plays a collusion of users, and the interface plays the vendor and the honest users. I_1 maintains the vendor's state ς, and some counters, which are updated in each query: χ_M: number of non-empty multi-coupons rightfully provided to the adversary, χ_C^x: number of available (used and unused) coupons given to x, and χ_R^x: number of coupons redeemed by x, where x denotes one of the participants, and can be either A to denote the adversary, or some arbitrary string $\mathcal{U}$ to denote a particular user. Now we present the queries and the actions performed by the interface.

$I_1.\texttt{GetPK}.$ Returns the vendor's PK to the adversary.

$I_1.\texttt{Issue}_v(k).$ If $k < 1$ or $k > k_{\max}$ the interface aborts (halts and returns *rej*), otherwise it simulates the $\texttt{Issue}_v$ algorithm playing the vendor, and interacts

---

[2] This reflects the properties of existing schemes, and simplifies the construction.

with the adversary, who plays the user. The counters are updated as follows: χ_M++, χ_C^A+=k (where ++ and +=k denote increment by 1 and k, resp.).

I_1.$\texttt{Issue}_u(\mathcal{U}, k)$. If $k < 1$, $k > k_{\max}$, $\mathcal{U} \in \mathcal{SU}$, or $\mathcal{U} \in \mathcal{CU}$, then the interface aborts, otherwise it simulates a protocol run between an honest user $\mathcal{U}$ and the vendor. $\mathcal{U}$ is an arbitrary value specified by the adversary to the interface, which allows the adversary to refer to precisely the same user later on. The user's view of the protocol is stored in a *transcript*, and the variables are updated: $\mathcal{SU} \leftarrow \mathcal{SU} \cup \{\mathcal{U}\}$, $\chi_C^\mathcal{U} \leftarrow k$, $\chi_R^\mathcal{U} \leftarrow 0$. In our security model every existing user has exactly one multi-coupon: a real world honest user with m multi-coupons can be simulated by m users, each one having a single multi-coupon.

I_1.$\texttt{Redeem}_v$. The interface performs the $\texttt{Redeem}_v$ algorithm, enabling the adversary to redeem one of his coupons. If the interaction is successful ($res_v = acc$) the counters are updated as follows: χ_R^A++, $\chi_M \leftarrow \min(\chi_M, \chi_C^A - \chi_R^A)$. (An adversary with at most $\chi_C^A - \chi_R^A$ unused coupons is not allowed to have more than $\chi_C^A - \chi_R^A$ non-empty multi-coupons.) These counters are important for the unforgeability and unsplittability requirements.

I_1.$\texttt{Redeem}_u(\mathcal{U})$. The interface simulates a $\texttt{Redeem}$ protocol run between the honest user $\mathcal{U}$ and the vendor (both algorithms $\texttt{Redeem}_u$ and $\texttt{Redeem}_v$ are simulated). If $res_v = acc$ the interface stores the user's view in the *transcript*, and sets $\chi_R^\mathcal{U}$++. The only information returned to the adversary is res_v.

I_1.$\texttt{Corrupt}(\mathcal{U})$. The interface first verifies that $\mathcal{U} \in \mathcal{SU}$, otherwise it aborts. Then it sets $\mathcal{SU} \leftarrow \mathcal{SU} \backslash \{\mathcal{U}\}$, $\mathcal{CU} \leftarrow \mathcal{CU} \cup \{\mathcal{U}\}$, and finally it gives to the adversary the user's previous protocol views, which are extracted from the *transcript*. The counters are updated: χ_C^A+=$\chi_C^\mathcal{U}$, χ_R^A+=$\chi_R^\mathcal{U}$, and if $\chi_C^\mathcal{U} > \chi_R^\mathcal{U}$, then χ_M+=1.

Interface 2 (I_2). This interface is capable of simulating a collection of honest users scheduled by the adversary, who plays the vendor. Again we use $\mathcal{SU}$ and $\mathcal{CU}$ to denote sets of scheduled and corrupted users resp., and the counters χ_C^x and χ_R^x with the same meaning as in I_1. The following queries are provided:

I_2.$\texttt{GetPK-SK}$. The interface gives the pair (PK, SK) to the adversary.

I_2.$\texttt{Issue}_u(\mathcal{U}, k)$. If $k \in [1, k_{\max}]$ and $\mathcal{U} \notin \mathcal{SU} \cup \mathcal{CU}$ the interface executes the $\texttt{Issue}_u$ algorithm (otherwise it aborts). Then, interface sets $\mathcal{SU} \leftarrow \mathcal{SU} \cup \{\mathcal{U}\}$, $\chi_C^\mathcal{U} \leftarrow k$, $\chi_R^\mathcal{U} \leftarrow 0$, and appends its protocol view to the *transcript*.

I_2.$\texttt{Redeem}_u(\mathcal{U})$. If $\mathcal{U} \notin \mathcal{SU}$ or $\chi_C^\mathcal{U} = \chi_R^\mathcal{U}$, then the interface aborts (the second condition prevents the interface from trying to overuse a multi-coupon). Then it executes the $\texttt{Redeem}_u$ algorithm simulating the honest user $\mathcal{U}$. The vendor stores the user's view in the *transcript*, and sets $\chi_R^\mathcal{U}$++.

I_2.$\texttt{Corrupt}(\mathcal{U})$. This query is handled exactly as in I_1.

4.3   Unforgeability

Informally, unforgeability means that no group of users (controlled by $\mathcal{A}$), with χ_C^A coupons in total (comprised in, say, m multi-coupons), should be able to redeem $\chi_R^A > \chi_C^A$ coupons. More formally, this property is defined as follows.

$ForgeGame(\mathcal{A}, \kappa)$:
   $(PK, SK) \leftarrow \texttt{Setup}(1^\kappa)$;
   $\sigma \leftarrow A_1^{I_1}(1^\kappa)$;
   if $(\chi_C^A \neq \chi_R^A)$ then return *unbroken*;
   $(res_A, res_v) \leftarrow (A_2(\sigma), I_1.\texttt{Redeem}_v)$;
   if $(res_v = acc)$ then return *broken*;
     else return *unbroken*;

$SplitGame(\mathcal{A}, \kappa)$:
   $(PK, SK) \leftarrow \texttt{Setup}(1^\kappa)$;
   $(\sigma_0, \dots, \sigma_{\chi_M}) \leftarrow A_1^{I_1}(1^\kappa)$
   for $i = 0$ to χ_M do:
     $(res_A^i, res_v^i) \leftarrow (A_2(\sigma_i), I_1.\texttt{Redeem}_v)$;
   if $(res_v^0 = acc \wedge \dots \wedge res_v^{\chi_M} = acc)$ then
     return *broken*; else return *unbroken*;

Fig. 1. Forgeability and Splittability Games

Definition 3 (Unforgeable *MCS*). *A multi-coupon scheme is* unforgeable *if there is no p.p.t adversary $\mathcal{A} := (A_1, A_2)$ that can win the forgeability game in Fig. 1 (ForgeGame$(\mathcal{A}, \kappa) = broken$) with non-negligible probability (in κ).*

An adversary $\mathcal{A}$ first interacts with the interface I_1 (i.e., queries $\texttt{GetPK}$, $\texttt{Issue}_v(\cdot)$, $\texttt{Issue}_u(\cdot, \cdot)$, $\texttt{Redeem}_v$, $\texttt{Redeem}_u(\cdot)$, and $\texttt{Corrupt}(\cdot)$). $\mathcal{A}$ wins if he is able to redeem an additional coupon after having redeemed the *same* number of coupons he has rightfully obtained. Note that any adversary $\mathcal{A}'$ who achieves $\chi_R^A > \chi_C^A$, can be transformed into an adversary $\mathcal{A}$, who wins the *ForgeGame* at the expense of at most a polynomial factor in the success probability.

4.4   Unsplittability

Informally, a *MCS* is *unsplittable* if it is infeasible for an adversary $\mathcal{A}$ rightfully holding at most χ_M non-empty *MC*s to generate $\chi_M + 1$ *shares* $\sigma_0, \dots, \sigma_{\chi_M}$, which can be used each to autonomously redeem at least one coupon. This must hold, even though $\mathcal{A}$ might have $\chi_C^A - \chi_R^A \geq \chi_M$ unused coupons.

Definition 4 (Unsplittability). *A multi-coupon scheme is* unsplittable *if there is no p.p.t. adversary $\mathcal{A} := (A_1, A_2)$ capable of winning the splittability game in Fig. 1 (SplitGame$(\mathcal{A}, \kappa) = broken$) with non-negligible probability (in κ).*

In the splittability game the adversary first interacts with the interface I_1, and outputs $\chi_M + 1$ indexed states (shares) $\sigma_0, \dots, \sigma_{\chi_M}$. Then he sequentially executes $\chi_M + 1$ redemption algorithms $A_2(\sigma_i)$, for $0 \leq i \leq \chi_M$. The adversary wins if each one of the $\chi_M + 1$ redemption algorithms succeeds.

We remark that, inside the "for loop" in Fig. 1, $A_2(\sigma_i)$ does not depend on the information obtained in the execution of $A_2(\sigma_j)$ with $i \neq j$. The adversary's only input is a state σ_i (for some i); this ensures the autonomous redemption. In contrast, the interface I_1 implicitly updates the vendor's state.

4.5   Unlinkability

Informally speaking, unlinkability means that an adversary playing the role of the vendor cannot recognize (significantly better than by a random guess) which honest user redeems a coupon when such a user is randomly selected from a pair of users of his choice (equivalently with *MC*s instead of users).

In [6] a simple definition of unlinkability is proposed. However, the adversary cannot further interact with the users after the challenge took place.

The number of unused coupons left in the selected pair of MCs can be easily used by the adversary to link the protocols. This problem is (almost) solved in [11] by hiding the number of unused coupons of the pair of challenged MCs from the adversary. However, this is done (in part) by requiring that none of the challenged MCs is ever emptied, hence the adversary is unrealistically prevented from using the last coupons within the challenged MCs.

Definition 5 (Unlinkability). *A multi-coupon scheme is* unlinkable *if there is no p.p.t. adversary* $\mathcal{A} := (A_1, A_2, A_3)$ *with non-negligible linkability advantage, which is defined as:* $Adv^{link}(\mathcal{A}, \kappa) = Pr[LinkGame(\mathcal{A}, \kappa) = broken] - 1/2.$

For the linkability game, the adversary $\mathcal{A}$ first interacts with the interface I_2 (queries GetPK-SK, $\text{Issue}_u(\cdot, \cdot)$, $\text{Redeem}_u(\cdot)$, and $\text{Corrupt}(\cdot)$), and outputs the user identities $\mathcal{U}_0$ and $\mathcal{U}_1$, of two scheduled users that have at least one unused coupon left (i.e. $\chi_C^x > \chi_R^x$, for $x \in \{\mathcal{U}_0, \mathcal{U}_1\}$). Then, b is randomly selected from $\{0, 1\}$, and the redemption algorithm $\text{Redeem}_u(\mathcal{U}_b)$ is executed with $\mathcal{A}$. Afterwards, $\mathcal{A}$ is given a set of queries $I_2(m_0, m_1, \mathcal{U}_0, \mathcal{U}_1)$, similar to those of I_2, except that the users $\mathcal{U}_0$ and $\mathcal{U}_1$ cannot be corrupted, and at most m_0 $\text{Redeem}_u(\cdot)$ queries can be made for the user $\mathcal{U}_0$ and m_1 queries for $\mathcal{U}_1$, where m_0 (resp. m_1) is the number of unredeemed available coupons minus one held by user $\mathcal{U}_0$ (resp. $\mathcal{U}_1$) before A_2 redeems. This hides the number of unused coupons from $\mathcal{A}$, thus avoiding the problem mentioned above. Finally, $\mathcal{A}$ outputs d. If $d = b$ the adversary won the game, otherwise he lost.

$LinkGame(A_1, A_2, A_3, \kappa)$:
$\quad (PK, SK) \leftarrow \text{Setup}(1^\kappa);$
$\quad (\mathcal{U}_0, \mathcal{U}_1, \varsigma) \leftarrow A_1^{I_2}(1^\kappa);$
$\quad \text{if not } (\mathcal{U}_0 \in \mathcal{SU} \land \mathcal{U}_1 \in \mathcal{SU} \land \chi_C^{\mathcal{U}_0} > \chi_R^{\mathcal{U}_0} \land \chi_C^{\mathcal{U}_1} > \chi_R^{\mathcal{U}_1})$
$\quad\quad \text{then return } unbroken;$
$\quad b \leftarrow \{0, 1\}; \; m_0 \leftarrow \chi_C^{\mathcal{U}_0} - \chi_R^{\mathcal{U}_0} - 1; \; m_1 \leftarrow \chi_C^{\mathcal{U}_1} - \chi_R^{\mathcal{U}_1} - 1;$
$\quad (res_{\mathcal{U}_b}, \varsigma) \leftarrow (I_2.\text{Redeem}_u(\mathcal{U}_b), A_2(\varsigma));$
$\quad d \leftarrow A_3^{I_2(m_0, m_1, \mathcal{U}_0, \mathcal{U}_1)}(\varsigma);$
$\quad \text{if } (res_{\mathcal{U}_b} = acc \land d = b) \text{ then return } broken; \text{ else return } unbroken;$

Theorem 1. *Unsplittability is strictly stronger than unforgeability.*

Proof (Sketch). ($\Rightarrow$) The condition $\chi_C^A = \chi_R^A$ in the forgeability game implies $\chi_M \leq 0$. Therefore, an adversary $\mathcal{A}$ against *ForgeGame* is also an adversary against *SplitGame* with at least the same success probability. E.g., if $\chi_M = 0$, then $\mathcal{A}$ "splits zero multi-coupons into one". For the other direction ($\nRightarrow$) consider the schemes proposed in [7,11] which are unforgeable but not unsplittable.

5   Our Multi-coupon Scheme

We propose the first unsplittable MCS where each coupon has an individual object, and coupons belonging to the same MC must be redeemed in certain

linear order, which is fixed during the issue procedure. The scheme can be easily extended with validity periods and arbitrary attributes for each coupon. In contrast to previous proposals [7], the number of coupons contained in a multi-coupon is not fixed, but is upper-bounded by $k_{\max}$. Therefore, no inefficient step is required for issuing a fraction of the maximum number of coupons [11]. This is useful, for instance, to implement a personalized electronic discount booklet, where variable discounts are offered in certain order.

5.1   Notation and Building Blocks

Commitment Scheme (CS). We use the integer tuple CS from [10], based on the scheme in [8], with a tuple $(g_1, \ldots, g_k, n)$ of k bases $g_i \in \mathrm{QR}_n$ (quadratic residues modulo n), for $1 \leq i \leq k$, and a special RSA modulus n as a public key. A commitment to $(x_1, \ldots, x_{k-1})$ has the form $C_x = g_1^{x_1} \cdots g_k^{x_k}$, where x_k is a value randomly chosen from an appropriate interval.

Proofs of Knowledge (PoK). We use a number of honest-verifier statistical zero-knowledge PoK. By $\mathtt{PoK}\{(\tilde{x}_1, \ldots, \tilde{x}_n) : R(\tilde{x}_1, \ldots, \tilde{x}_n)\}$ we denote an interactive PoK, where a prover proves to a verifier that she knows a witness $(\tilde{x}_1, \ldots, \tilde{x}_n)$ (which we always denote with tilded variables) such that the relation R holds, and the verifier does not gain any useful information beyond this assumption.

Proof of Equality of Representations. $\mathcal{P}$ proves that she is able to open two commitments C_1 and C_2 (for two possibly different instances of the commitment scheme), such that certain components of the openings are equal. For example, we write $\mathtt{PoKEqRep}\{(\tilde{x}, \tilde{r}_x, \tilde{y}, \tilde{r}_y) : C_1 = g_1^{\tilde{x}} g_2^{\tilde{r}_x} \wedge C_2 = \hat{g}_1^{\tilde{y}} \hat{g}_2^{\tilde{r}_y} \wedge \tilde{x} = \tilde{y}\}$.

Camenisch Lysyanskaya signature scheme (CLS). The CLS [5] is a simple signature scheme with efficient protocols based on the *strong RSA assumption*. The following description is done in the context of our scheme.

$CLS.\mathtt{Setup}(1^\kappa)$. The signer $\mathcal{S}$ generates a special RSA modulus $n = pq$, such that n has size $\ell_n := 2\kappa$, where κ is a security parameter. Then he chooses numbers $a, b \in_R \mathrm{QR}_n$ called bases, and a constant $c \in_R \mathrm{QR}_n$. The public key CLS_{PK} is (a, b, c, n), and the secret key CLS_{SK} is the prime number p.

$CLS.\mathtt{Sign}(x, CLS_{SK})$. To sign a message $x \in [0, 2^{\ell_m})$, the signer chooses a random prime e of size exactly $\ell_e := \ell_m + 2$, a random number s of size at most $\ell_s := \ell_n + \ell_m + \ell$, where ℓ is another security parameter, $\mathcal{S}$ computes $v \leftarrow (a^x b^s c)^{e^{-1}} \pmod{n}$, and outputs (e, s, v).

$CLS.\mathtt{Verify}(x, \sigma, CLS_{PK})$. For $(e, s, v) := \sigma$, the algorithm tests whether $v^e \equiv a^x b^s c \pmod{n}$, $x \in [0, 2^{\ell_m})$, $s \in [0, 2^{\ell_s})$, e is exactly ℓ_e bits long, and outputs *true* or *false* accordingly.

The signature allows the following useful protocols:

Signature on a committed value and PoK of this signature [5]. Signature generation is a protocol from [5, Fig. 1] between a user $\mathcal{U}$ and a signer $\mathcal{S}$, who knows the secret key CLS_{SK}. (Let $CLS_{PK} := (a, b, c, n)$ be the corresponding public key.) The common input to $\mathcal{U}$ and $\mathcal{S}$ is a commitment C_x, for which $\mathcal{U}$ supposedly knows an opening $(x, r_x) : C_x = a^x b^{r_x}$. At the end of the protocol $\mathcal{U}$

obtains a signature $\sigma := (e, s, v)$ on x, while x is statistically hidden from $\mathcal{S}$. We denote this protocol as: $\sigma \leftarrow \texttt{SigOnCommit}\{\mathcal{U}(x, r_x), \mathcal{S}(CLS_{SK})\}(C_x)$.

Further, for a commitment C'_x, $\mathcal{U}$ can prove the knowledge of (x, r'_x, e, s, v) [5, Figure 2], such that (x, r'_x) is an opening of C'_x, and (e, s, v) is a valid signature on x, where x and σ are hidden by the zero-knowledge property of the protocol. An auxiliary commitment scheme (g, h, n) is required, where n is the modulus used in the CLS_{PK}. We denote this protocol as: $\texttt{PoKSigOnCommit}\{(\tilde{x}, \tilde{r}'_x, \tilde{\sigma}) : C'_x = a^x b^{r'_x} \wedge CLS.\texttt{Verify}(\tilde{x}, \tilde{\sigma}, CLS_{PK})\}$.

This signature scheme can be extended to sign message tuples $(x_1, \ldots, x_k)$ by introducing k bases a_i [5]. Also, the pair of protocols above can be extended to support multiple messages, and selective message disclosure. For instance, we denote by $\texttt{SigOnCommit}\{\mathcal{U}(\tilde{x}_1, \tilde{r}_{x_1}); \mathcal{S}(CLS3_{SK})\}(C_{x_1}, x_2, x_3)$ a protocol to generate a signature on a 3-tuple (x_1, x_2, x_3), where the message x_1, is (supposedly) blinded by a commitment C_{x_1}, and two messages x_2 and x_3 are disclosed in clear. Similarly, by $\texttt{PoKSigOnCommit}\{(\tilde{x}_3, \tilde{r}_{x_3}, \tilde{\sigma}) : C_{x_3} = a_3^{\tilde{x}_3} b^{\tilde{r}_{x_3}} \wedge CLS3.\texttt{Verify}(x_1, x_2, \tilde{x}_3, \tilde{\sigma}, CLS3_{PK})\}$ we denote the corresponding PoK that $\mathcal{U}$ knows a signature σ on a tuple (x_1, x_2, x_3), where x_1 and x_2 are disclosed to the verifier, but x_3 is kept blinded.

5.2 Construction

The components of our construction are two instances of the CL signature scheme: CLS, for messages in $[0, 2^{\ell_m})$, and $CLS3$, for messages in $[0, 2^{\ell_m})^3$.

Setup(1^κ). The vendor $\mathcal{V}$ generates an instance of the $CLS3$ signature scheme: $(CLS3_{PK}, CLS3_{SK}) := ((a_1, a_2, a_3, b, c, n), p) \leftarrow CLS3.\texttt{Setup}(1^\kappa)$, and the CLS signature scheme: $(CLS_{PK}, CLS_{SK}) := ((\hat{a}, \hat{b}, \hat{c}, \hat{n}), \hat{p}) \leftarrow CLS.\texttt{Setup}(1^\kappa)$. It is assumed that $CLS3$ and CLS have the same parameters $\ell_n, \ell_m, \ell_e, \ell,$ and ℓ_s. Additionally, $\mathcal{V}$ generates two instances of the CS by computing $g, h \in_R QR_n$, and $\hat{g}, \hat{h} \in_R QR_{\hat{n}}$. These commitment schemes are only used in the $\texttt{PoKSigOnCommit}$ protocol. Finally, $\mathcal{V}$ initializes a counter on sequence numbers: $\chi_{sq} \leftarrow 1$, stores $SK := (CLS3_{SK}, CLS_{SK})$, publishes $PK := (CLS3_{PK}, g, h, CLS_{PK}, \hat{g}, \hat{h})$, and creates an empty database DB of coupon identifiers.

Issue. In this protocol (Figure 2) the user $\mathcal{U}$ interacts with the vendor $\mathcal{V}$ to obtain k coupons with objects ob_i, for $0 \le i < k$. First, $\mathcal{V}$ chooses a new sequence number $sq_0 \leftarrow \chi_{sq}$, updates the counter $\chi_{sq} \leftarrow \chi_{sq} + k + 1$, computes $\sigma'_0 \leftarrow CLS.\texttt{Sign}(sq_0, CLS_{SK})$, and sends both sq_0 and σ'_0 to $\mathcal{U}$. Then, $\mathcal{U}$ randomly chooses k coupon identifiers id_i, for $0 \le i < k$, and commits to them by computing the commitments C_{id_i}, which are sent to $\mathcal{V}$. Afterwards, for each i, $0 \le i < k$, $\mathcal{U}$ executes the $\texttt{SigOnCommit}$ protocol to obtain a $CLS3$ signature σ_i on $(id_i, ob_i, sq_0 + i)$, where id_i is kept blinded in C_{id_i}, and $ob_i, sq_0 + i$ are known by the $\mathcal{V}$. Notice that only the first coupon is redeemable.

Redeem. In the redeem protocol (Figure 3) $\mathcal{U}$ selects her next unused redeemable coupon $(id_i, ob_i, sq_i, \sigma_i, \sigma'_i)$ from her MC, commits to sq_i via $C_{sq_i} \leftarrow a_3^{sq_i} b^{r_{sq_i}}$, $C'_{sq_i} \leftarrow \hat{a}^{sq_i} \hat{b}^{r'_{sq_i}}$ using the appropriate moduli n and $\hat{n}$ of the two signature schemes, and sends id_i, ob_i, C_{sq_i}, and C'_{sq_i} to $\mathcal{V}$. The vendor checks that id_i

Common input: public keys $CLS_{PK} = (\hat{a}, \hat{b}, \hat{c}, \hat{n})$, $CLS3_{PK} = (a_1, a_2, a_3, b, c, n)$,
  number of single coupons k, object identifiers ob_i, $i = 0, \ldots, k-1$
User's input: $-$
Vendor's input: private keys $CLS_{SK} = \hat{p}$, $CLS3_{SK} = p$

User $\mathcal{U}$	Vendor $\mathcal{V}$

$sq_0 \leftarrow \chi_{sq};\ \chi_{sq} \leftarrow \chi_{sq} + k + 1;$
$\sigma_0' \leftarrow CLS.\texttt{Sign}(sq_0, CLS_{SK});$

$\xleftarrow{\quad sq_0,\ \sigma_0' \quad}$

$CLS.\texttt{Verify}(sq_0, \sigma_0, CLS_{PK}) \overset{?}{=} true$
for each $i = 0, \ldots, k-1$ do
  $id_i \in_R (0, 2^{\ell_m});$
  $r_{id_i} \in_R (0, 2^{\ell_n});$
  $C_{id_i} \leftarrow a_1^{id_i} b^{r_{id_i}};$
end do;

$\xrightarrow{\quad C_{id_0}, \ldots, C_{id_{k-1}} \quad}$

for each $i = 0, \ldots, k-1$ do
  $\sigma_i \quad \leftarrow \quad \texttt{SigOnCommit}\{\mathcal{U}(id_i, r_{id_i}) : \mathcal{V}(CLS3_{SK})\}(C_{id_i}, ob_i, sq_i)$
  $sq_{i+1} \leftarrow sq_i + 1;$ $\qquad\qquad\qquad\qquad\qquad sq_{i+1} \leftarrow sq_i + 1;$
end do;

Fig. 2. Issue Protocol

is not in the database, and inserts it. Then, $\mathcal{U}$ proves that C_{sq_i} and C'_{sq_i} are commitments to the same sequence number sq_i. Then, $\mathcal{U}$ uses $\texttt{PoKSigOnCommit}$ to prove in zero knowledge that she knows a $CLS3$ signature σ_i on the tuple (id_i, ob_i, sq_i) without disclosing σ_i. Additionally, $\mathcal{U}$ proves to $\mathcal{V}$ the knowledge of a CLS signature σ_i' on sq_i, without disclosing any useful information about it to $\mathcal{V}$. Finally, if every PoK succeeded, $\mathcal{U}$ obtains a signature σ_{i+1}' on $sq_{i+1} := sq_i + 1$, i.e., her next coupon becomes redeemable.

In the description above, it is assumed that $\mathcal{U}$ always outputs rej in case any obtained signature is invalid. Similarly, $\mathcal{V}$ must output rej in case any PoK fails.

5.3   Security Proofs

In this section we present a number of theorems stating the properties of our scheme. Due to space restrictions we omit some proofs.

Theorem 2. *The MCS proposed in Section 5 is correct. (Proof omitted)*

Theorem 3. *The MCS proposed in Section 5 is unsplittable.*

Proof (Sketch). Assume $\mathcal{A}$ is an adversary against unsplittability. It is possible to construct an algorithm $\mathcal{B}$, which outputs a forgery to one of the signatures $CLS3$ or CLS with at least half the success probability of $\mathcal{A}$ (minus some negligible term). $\mathcal{B}$ simulates the interface I_1, and must answer the queries made by $\mathcal{A}$. The only steps which $\mathcal{B}$ cannot trivially simulate are those which require the

Common input: public keys $CLS_{PK} = (\hat{a}, \hat{b}, \hat{c}, \hat{n})$, $CLS3_{PK} = (a_1, a_2, a_3, b, c, n)$,
  bases $\hat{g}, \hat{h}, g, h$ for internal use of $\texttt{PoKSigOnCommit}$ protocols
User's input: single coupon $(id_i, ob_i, sq_i, \sigma_i, \sigma_i')$
Vendor's input: private keys $CLS_{SK} = \hat{p}$, $CLS3_{SK} = p$, data base DB for coupon ids

User $\mathcal{U}$ Vendor $\mathcal{V}$

$r_{sq_i}', r_{sq_i} \in_R (0, 2^{l_n})$;
$C_{sq_i}' \leftarrow \hat{a}^{sq_i} \hat{b}^{r_{sq_i}'}$;
$C_{sq_i} \leftarrow a_3^{sq_i} b^{r_{sq_i}}$; $\xrightarrow{\quad C_{sq_i}', C_{sq_i}, id_i, ob_i \quad}$

$\qquad \texttt{PoKEqRep}\{(\tilde{sq}_i, \tilde{r}_{sq_i}, \tilde{sq}_i', \tilde{r}_{sq_i}') : C_{sq_i}' = \hat{a}^{\tilde{sq}_i} \hat{b}^{\tilde{r}_{sq_i}'} \wedge C_{sq_i} = a_3^{\tilde{sq}_i'} b^{\tilde{r}_{sq_i}} \wedge \tilde{sq}_i = \tilde{sq}_i'\}$

$\qquad \texttt{PoKSigOnCommit}\{(\tilde{sq}_i, \tilde{r}_{sq_i}', \tilde{\sigma}_i') : C_{sq_i}' = \hat{a}^{\tilde{sq}_i} \hat{b}^{\tilde{r}_{sq_i}'} \wedge CLS.\texttt{Verify}(\tilde{sq}_i, \tilde{\sigma}_i', CLS_{PK})\}$

$\texttt{PoKSigOnCommit}\{(\tilde{sq}_i, \tilde{r}_{sq_i}, \tilde{\sigma}_i) : C_{sq_i} = a_3^{\tilde{sq}_i} b^{\tilde{r}_{sq_i}} \wedge CLS3.\texttt{Verify}((id_i, ob_i, \tilde{sq}_i), \tilde{\sigma}_i, CLS3_{PK})\}$

                                                        check that id_i is not in DB;
$sq_{i+1} \leftarrow sq_i + 1$;   add id_i to DB;
$C_{sq_{i+1}}' \leftarrow C_{sq_i}' \hat{a}$; $C_{sq_{i+1}}' \leftarrow C_{sq_i}' \hat{a}$;
$\sigma_{i+1}' \quad \leftarrow \qquad \texttt{SigOnCommit}\{\mathcal{U}(sq_{i+1}, r_{sq_i}) : \mathcal{V}(CLS_{SK})\}(C_{sq_{i+1}}')$

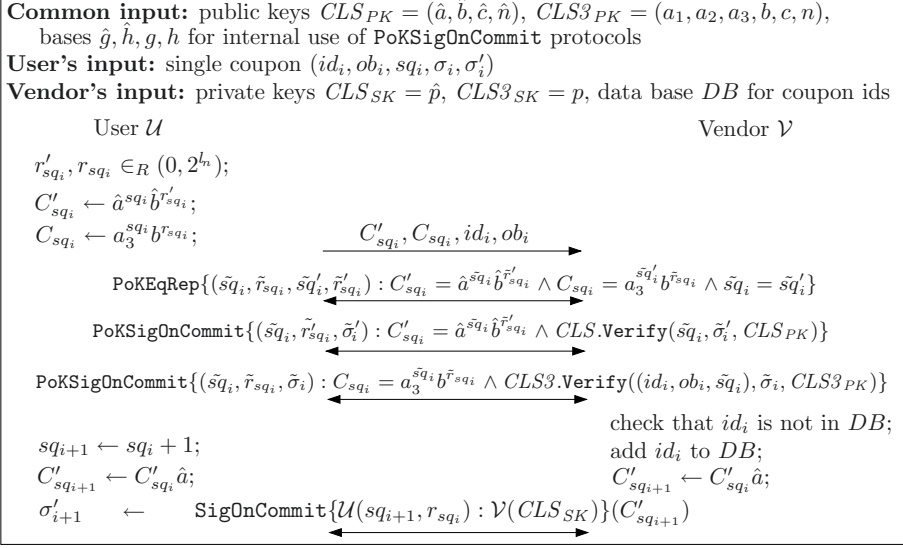**Fig. 3.** Redeem Protocol

generation of a signature. To accomplish this he has black box access to $\mathcal{A}$, and access to two signature oracles $CLS3.\texttt{Sign}(\cdot, CLS3_{SK})$ and $CLS.\texttt{Sign}(\cdot, CLS_{SK})$ (for two randomly chosen secret keys $CLS3_{SK}$ and CLS_{SK} unknown to $\mathcal{B}$). Each time $\mathcal{B}$ must sign a message sq_0 in clear, he simply queries the CLS oracle.

The execution of the $\texttt{SigOnCommit}$ protocol, in both the issue and redeem procedures, can be simulated towards $\mathcal{A}$ as described in the proof of [5, Lemma 6.1], where the actual signature computation is outsourced to the corresponding signature oracle.

Because of the soundness of every zero-knowledge PoK in the $\texttt{Issue}$ and $\texttt{Redeem}$ protocols, we can assume that during the protocol executions, $\mathcal{B}$ can extract (by using rewinding) all the witnesses for each PoK from $\mathcal{A}$. This allows $\mathcal{B}$ to obtain the attributes of all coupons, both issued and redeemed.

It is possible to prove that the number of unused redeemable coupons provided to $\mathcal{A}$ in the unsplittability game is at most χ_M (here it is important that $\mathcal{B}$ updates the counter χ_{sq} to avoid signing the same sequence number twice). In the last part of the unsplittability game $\mathcal{A}$ is able to redeem $\chi_M + 1$ coupons. Hence $\mathcal{B}$ is able to extract from $\mathcal{A}$ the information of $\chi_M + 1$ redeemable coupons $(id_i, ob_i, sq_i, \sigma_i, \sigma_i')$, for $0 \leq i \leq \chi_M$. In particular, at least one of these coupons has a signature σ_i on (id_i, ob_i, sq_i) or σ_i' on sq_i, which was not queried to the respective signature oracle and therefore is an existential forgery of one of the signature schemes. In order to identify the forgery, $\mathcal{B}$ stores every signature-message pair queried to the signature oracles.

Theorem 4. *The MCS proposed in Section 5 is unforgeable.*

Proof. The theorem trivially follows from Theorems 1 and 3.

Coupon objects are useful features that unavoidably come at a price: they can be trivially used by the vendor to link protocol runs (e.g. by assigning unique coupon objects to each user). Hence, our construction does not meet unlinkability as in Definition 3. However, in practice, if there are many redeemable coupons at any time for each possible coupon object, then this information by itself does not substantially harm privacy.

Theorem 5. *The MCS of Section 5 restricted to constant coupon objects is unlinkable.*

Proof (Sketch). Wlog assume we can guess the users $\mathcal{U}_0$ and $\mathcal{U}_1$ challenged by $\mathcal{A}$. The proof is based on the existence of simulators for each one of the proofs of knowledge employed in the protocols, and can be organized as a sequence of games [16]. We can construct a series of modified games from the unlinkability game, by substituting, one by one, every PoK and every commitment used in the Issue and Redeem protocols executed by the users $\mathcal{U}_0$ and $\mathcal{U}_1$. The hiding property of the commitment scheme can be used to replace the commitments by random values. In each transition, the adversary's success probability is modified only by a negligible amount.

In the last game, the adversary's view regarding the users $\mathcal{U}_0$ and $\mathcal{U}_1$ is completely simulated, thus his success probability is exactly $1/2$. This implies that his linkability advantage for the original game is negligible.

Complexity and Extensions. The computation and communication complexity of the issue protocol is linear in k, while the redemption complexity is constant in k. This improves the complexity of the scheme in [7], which is linear for both issue and redeem, but is less efficient than the scheme in [11], which has constant complexity for both protocols. However, for many applications the scheme in [11] is less practical than ours, because it lacks specific attributes per coupon, and it only offers weak unsplittability.

It is possible to extend the scheme by adding additional attributes to each coupon. For instance, we can easily implement validity periods by adding two attributes t^a and t^b, such that a coupon is only valid if a publicly known time variable belongs to the interval $[t^a, t^b]$. Furthermore, by using standard zero-knowledge interval protocols [2], it is possible for a user to prove that the coupon is valid at some precise date/time, without disclosing either t^a or t^b. Note that this has a similar effect on unlinkability as coupon objects.

6   Conclusion and Future Work

In this paper we introduced a privacy-protecting multi-coupon system, which improves previous proposals with regard to various aspects: better efficiency than [7], weaker assumptions than [11], and stronger security requirements. In particular, we provide an improved security model with a stronger definition of

unsplittability, which discourages sharing of multi-coupons without relying on the all-or-nothing principle. Unlike alternative approaches, our scheme does not encode valuable information into the coupons to dissuade users from sharing them. Therefore, it can be considered more privacy-friendly. Moreover, it can be extended with additional attributes such as validity periods.

Some open problems are to design a *MCS* with the properties above, but in which coupons can be redeemed in arbitrary order, and to develop *MCS*s for more general settings (e.g. multiple collaborating vendors).

References

1. Blundo, C., Cimato, S., De Bonis, A.: Secure e-coupons. Electronic Commerce Research 5(1), 117–139 (2005)
2. Boudot, F.: Efficient proofs that a committed number lies in an interval. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 431–444. Springer, Heidelberg (2000)
3. Brands, S.: A technical overview of digital credentials. research report (February 2002), http://www.xs4all.nl/#brands/
4. Camenisch, J., Hohenberger, S., Lysyanskaya, A.: Compact e-cash. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 302–321. Springer, Heidelberg (2005)
5. Camenisch, J., Lysyanskaya, A.: A signature scheme with efficient protocols. In: Cimato, S., Galdi, C., Persiano, G. (eds.) SCN 2002. LNCS, vol. 2576, Springer, Heidelberg (2003)
6. Canard, S., Gouget, A., Hufschmitt, E.: A handy multi-coupon system. In: Zhou, J., Yung, M., Bao, F. (eds.) ACNS 2006. LNCS, vol. 3989, pp. 66–81. Springer, Heidelberg (2006)
7. Chen, L., Enzmann, M., Sadeghi, A.-R., Schneider, M., Steiner, M.: A privacy-protecting coupon system. In: Patrick, A.S., Yung, M. (eds.) FC 2005. LNCS, vol. 3570, pp. 93–108. Springer, Heidelberg (2005)
8. Damgård, I., Fujisaki, E.: A statistically hiding integer commitment scheme based on groups with hidden order. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, Springer, Heidelberg (2002)
9. Kiayias, A., Tsiounis, Y., Yung, M.: Traceable signatures. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 571–589. Springer, Heidelberg (2004)
10. Lipmaa, H.: On diophantine complexity and statistical zero–knowledge arguments. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 398–415. Springer, Heidelberg (2003)
11. Nguyen, L.: Privacy-protecting coupon system revisited. In: Di Crescenzo, G., Rubin, A. (eds.) FC 2006. LNCS, vol. 4107, Springer, Heidelberg (2006)
12. Nguyen, L., Safavi-Naini, R.: Dynamic k-times anonymous authentication. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 318–333. Springer, Heidelberg (2005)
13. Odlyzko, A.: Privacy, economics, and price discrimination on the internet. In: ICEC 2003. Proceedings of the 5th international conference on Electronic commerce, pp. 355–366. ACM Press, New York (2003)

14. Park, K., Gómez, M.: The coupon report: A study of coupon discount methods. Technical report, Department of Applied Economics and Management, Cornell University (2004), http://aem.cornell.edu/research/researchpdf/rb0407.pdf
15. Persiano, P., Visconti, I.: An efficient and usable multi-show non-transferable anonymous credential system. In: Juels, A. (ed.) FC 2004. LNCS, vol. 3110, Springer, Heidelberg (2004)
16. Shoup, V.: Sequences of games: A tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332 (2004), http://eprint.iacr.org/
17. Syverson, P.F., Stubblebine, S.G., Goldschlag, D.M.: Unlinkable serial transactions. In: FC 1997. LNCS, vol. 1318, pp. 39–56. Springer, Heidelberg (1997)

Panel: RFID Security and Privacy

Kevin Fu

University of Massachusetts Amherst
Amherst, MA, USA
kevinfu@cs.umass.edu

Abstract. The panel on RFID security and privacy included Ross Anderson, Jon Callas, Yvo Desmedt, and Kevin Fu. Topics for discussion included the "chip and PIN" EMV payment systems, e-Passports, "mafia" attacks, and RFID-enabled credit cards. Position papers by the panelists appear in the following pages, and the RFID-enabled credit card work appears separately in these proceedings.

Position Statement in RFID S&P Panel: RFID and the Middleman

Ross Anderson

Cambridge University
http://www.ross-anderson.com

Abstract. Existing bank-card payment systems, such as EMV, have two serious vulnerabilities: the user does not have a trustworthy interface, and the protocols are vulnerable in a number of ways to man-in-the-middle attacks. Moving to RFID payments may, on the one hand, let bank customers use their mobile phones to make payments, which will go a fair way towards fixing the interface problem; on the other hand, protocol vulnerabilities may become worse. By 2011 the NFC vendors hope there will be 500,000,000 NFC-enabled mobile phones in the world. If these devices can act as cards or terminals, can be programmed by their users, and can communicate with each other, then they will provide a platform for deploying all manner of protocol attacks. Designing the security protocols to mitigate such attacks may be difficult. First, it will include most of the hot topics of IT policy over the last ten years (from key escrow through DRM to platform trust and accessory control) as subproblems. Second, the incentives may lead the many players to try to dump the liability on each other, leading to overall system security that is equivalent to the weakest link rather than to sum-of-efforts and is thus suboptimal.

1   Introduction

Card payment systems have come under repeated attack from forgers and other fraudsters. The mechanisms used in magnetic-strip cards, and attacks on the ATMs that rely on them, are documented in [2], while the back-end backing systems are described in [3]. Banks in Europe are now moving to smartcards following the EMV standard, branded in the UK and Ireland as 'Chip and PIN'. In the most commonly-deployed variant of this standard, bank cards contain a smartcard chip that will verify a customer PIN against a locally-stored value, and also one or more cryptogram generation keys. These are used to compute a message authentication code (MAC) on transaction data; the MAC is verified by the card-issuing bank. As many countries, including the USA, have not adopted EMV, the cards also have a magnetic strip for fall-back operation. The introduction of EMV has altered the fraud landscape, with a reduction in cardholder-present forged-card losses, coupled with increases in cardholder-not-present fraud and mag-stripe-fallback fraud.

Now that European cardholders are used to entering cards and PINs for all bank card transactions, rather than just ATM transactions, it has become much

easier for attackers using false terminals to harvest card and PIN data. Previously, card forgery attacks typically involved skimmers attached to the front of ATMs; now we see a spate of offences involving wiretapping of the links from EMV terminals to branch server equipment. With frauds reported in France and Italy, both card and PIN data were collected using surreptitiously-installed wiretap devices, which send the data to waiting criminals by wireless [11]. In a series of recent UK cases, card data have also been harvested using an in-store wiretap, but the PINs collected by observation. Many offences have been reported in garages, staffed by Sri Lankan Tamils, leading to ATM withdrawals in India, Malaysia or Thailand; it's reported that the Tamil Tigers use these forgeries for fundraising – presumably intimidating their UK expatriates into collaborating [8,12].

In addition to such fairly straightforward frauds, it turns out that the EMV protocols and their implementations are vulnerable to various middleperson attacks [1]. In addition to capturing card details and PINs for use in magnetic-strip terminals, villains can do various kinds of man-in-the-middle attack, and the transactions provided by cryptographic hardware security modules to support EMV have flaws that enable bank insiders to extract PINs. The growing number of bank customers who complain that stolen chip-and-PIN cards were used without the PIN possibly having been compromised suggests that at least some banks have corrupt staff who sell PINs to criminals; this was a known modus operandi with magnetic-strip cards in the 1990s [2]. Meanwhile, Murdoch and Drimer have shown that real-time man-in-the-middle attacks are feasible by implementing them [9]; and it's also been noted that a bank customer could use a middleperson device to fix the protocol by providing a trustworthy user interface – she could observe the actual data traffic between the card and the terminal, rather than the possibly false data displayed by the terminal [5].

2   Implications for RFID

The bank card industry in the USA and Japan is introducing RFID credit cards, described by Heydt-Benjamin et al. in [7]. Although the specifications are confidential, protocol dumps suggest that the mechanisms are very similar to EMV, although with the crypto (the MAC) missing. The authors of that paper showed that cloning and relay attacks work in principle by scanning a credit card in its sealed delivery envelope and making a purchase with it.

Meanwhile, the mobile-phone industry is introducing Near-Field Communications (NFC), whereby a phone acquires the capability to act as either an RFID device or an RFID terminal under program control. The NFC protocols themselves provide an abstraction layer between the four different RF interfaces already deployed, thus providing a clean interface for the software developer. The NFC Forum envisages about half a billion NFC-enabled handsets in use by 2011 [10].

This technology holds out the prospect of solving the problem of a trustworthy user interface. The plan is that instead of being a relatively dumb device, your credit card will be an application on your mobile phone. You bring your mobile

into close proximity with the merchant terminal, an application displays the sale amount, you authorise this, and the transaction goes through. It may also bring further security benefits, such as a single point of revocation [6].

However, without protocols giving robust protection against man-in-the-middle attacks, the trust that might be placed in this interface will be largely illusory. Worse, NFC is likely to make middleperson attacks much easier. At present, such an attack requires the construction of custom hardware; in future, an attack could be carried out by software installed on commodity mobile phones. One phone could act as a rogue merchant terminal to the cardholder, and communicate with another that acts as a card to a merchant elsewhere.

Transaction forwarding does have 'honest' uses. (Once when I stayed at a hotel in Malaŵi I found that the bill appeared on my credit card via a South African merchant in Rands – an obvious attempt to collect a slightly harder currency, which cost me no more.) However, most forwarding is likely to be objectionable. Someone thinking she's buying an hour on a parking meter in Baltimore for $2 might find out, when the credit card bill arrives, that the bank thinks she paid $2000 for casino chips in Macau.

Limiting transactions without cryptography to low values – as some suggest – won't solve all the problems. For example, a crook might collect large numbers of small payments from passers-by; this might be done by malware installed on the phone of an unwitting victim, which would steal a few dollars from everyone he passes. Theft might not be the only objective; RFID will be used for sports ticketing, so known hooligans who are banned from buying tickets will have every incentive to spoof the system.

If cryptography is introduced, it's not obvious how. The banks will want end-to-end protection, but merchants will want access to transaction data; and the police will want access to records for intelligence/evidence, raising all the old issues about escrow. It's unclear that much more can be done than is already being done with EMV – message authentication codes that might just be upgraded to digital signatures eventually.

It's also not obvious who will design any new protocol suites to support NFC payments. The NFC Forum limits itself to interoperability, and the protection issue are not just about bank payments but transport tickets, sports tickets, supermarket coupons and much else. How many protocol suites will there be? What sort of limits are desirable on relaying / cloning bits? Who is going to specify the protocols? (VISA? Microsoft?) Who will regulate them? (The FTC and the EU's DG Comp? The Federal Reserve and the European Central Bank?)

3   Conclusions

The introduction of RFID payments based on programmable devices such as NFC mobile phones may fix one of the problems underlying bank card fraud – the lack of a trustworthy user interface. However, this may well be at the cost of seriously exacerbating the other main problem – the vulnerability of current payment protocols to various man-in-the-middle attacks.

What's more, the responsibility for security is very widely dispersed, and defenders may hope they can rely on each others' mechanisms. Security economics teaches that we get a much more appropriate level of protection where this results from the sum of defenders' efforts than in the cases where it results from the weakest link – from the least awful of the disparate efforts of a number of possibly uncoordinated defenders [13]. Unfortunately, the security of RFID transactions looks set to become a matter of the weakest link. So a large-scale infrastructure may be deployed that's not only systemically vulnerable to man-in-the-middle attacks, but that actually provides the platform required for these attacks to be carried out easily. I'd suggest that application providers think hard about these issues now; it will be much more expensive later.

References

1. Adida, B., Bond, M., Clulow, J., Lin, A., Murdoch, S., Anderson, R.J., Rivest, R.: "Phish and Chips". In: Security Protocols Workshop (March 2006), http://www.ross-anderson.com
2. Anderson, R.J.: "Why Cryptosystems Fail". Communications of the ACM 37(11), 32–40 (1994)
3. Anderson, R.J.: Security Engineering – A Guide to Building Dependable Distributed Systems. Wiley, Chichester (2001)
4. Anderson, R.J.: Why Information Security is Hard – An Economic Perspective. In: Proceedings of the Seventeenth Computer Security Applications Conference, pp. 358–365. IEEE Computer Society Press, Los Alamitos (2001), http://www.cl.cam.ac.uk/ftp/users/rja14/econ.pdf
5. Anderson, R.J., Bond, M.: The Man-in-the-Middle Defence. In: Security Protocols Workshop (March 2006), http://www.ross-anderson.com
6. Baard, M.: Will new RFID technology help or hinder security? (April 27, 2005), http://searchsecurity.techtarget.com/originalContent/0,289142,sid14_gci1083417,00.html
7. Heydt-Benjamin, T.S., Bailey, D.V., Fu, K., Juels, A., O'Hare, T.: Vulnerabilities in First-Generation RFID-enabled Credit Cards. In: Dietrich, S., Dhamija, R. (eds.) FC 2007. LNCS, vol. 4886, pp. 2–14. Springer, Heidelberg (2007)
8. Jayawardhana, W.: Tamil Tigers suspected of scamming millions in Britain, http://lankapage.wordpress.com/2007/01/17/
9. Murdoch, S.J.: Chip & PIN relay attacks (February 6, 2007), http://www.lightbluetouchpaper.org/
10. Near Field Communication and the NFC Forum: The Keys to Truly Interoperable Communications (2006), www.nfc-forum.org
11. Clonavano carte con il bluetooth Scoperta nuova truffa telematica. In: la Repubblica (September 4, 2006), http://www.repubblica.it/2006/09/sezioni/cronaca/truffa-blue/truffa-blue/truffa-blue.html
12. Shoesmith, K.: Garage Scam funded Terror Group, Hull Daily Mail, p. 1, (January 16, 2007), http://www.srilanka-botschaft.de/NEWSupdates_neu/Press_Releases/Press_Pol_Government_Statement_070119bE.htm
13. Varian, H.: System Reliability and Free Riding, http://www.sims.berkeley.edu/resources/affiliates/workshops/econsecurity/econws/49.pdf

Position Statement in RFID S&P Panel: Contactless Smart Cards

Jon Callas

PGP Corporation
jon@pgp.com

1 Problem Statement

When considering the security and risks of RFID systems, we must be careful to consider the issues that are specific to the device being an RFID as opposed to the inherent security issues of the device itself. RFID devices fall into two main categories:

- Simple devices that are primarily echo-responders. While this is an over-simplification, for example these devices can often be re-programmed to reply with a new response (or none at all), they do little to no actual computing.
- Smart-card-like systems that are also referred to as "contactless" smart cards. These devices draw their power from induction and communicate by radio, but are otherwise ordinary smart cards.

It is these latter RFID devices that I am going to concern myself with in this article because they are in computation the equivalent of a traditional smart card. That they are "contactless" devices gives them a number of advantages over ordinary smart cards. They do not require a contact pad for communications and power. Often, they can transmit data at far higher speeds than their analogues. And lastly, operating them is much faster. Since the device needs merely to be close enough to a terminal to work, it is easy to use. Since it has a serial communications line that runs fast, it doesn't have to be in proximity for very long.

However, these very advantages create security issues that are unique to the RFID devices and are not shared by their analogues. These security issues can turn the devices from being security measures to security threats. It is some of these issues that I consider in this paper.

2 Action at a Distance

Contactless devices can be operated at a distance. This quality means that they can be read at a distance, and not only when their owner wants them to. When the device receives enough power and the proper query, it responds.

Fortunately, the distance that these devices can respond at is relatively short. They draw their power from the radio that communicates with them, and that

makes a maximum read distance of a small handful of meters, with arbitrarily large broadcast power. With reasonable expectations on power that be carried by an attacker (or drawn from ordinary electrical current), a single meter or so is the maximum distance that an attacker can be from the card.

Note that this is very different from the echo-responder types of RFIDs, which have been shown to be readable from tens of meters given a good antenna.

However, this means that an attacker with a backpack can still activate cards in public places, particularly on trains, in stores, or other crowded public places. It is not unreasonable for an attacker to build a surreptitious reader into a merchant kiosk or cash register.

3   Radio Shielding

The obvious defense against unauthorized reading is to shield the device. It is only a little tricky to shield the device. Aluminum foil, metalized plastic mesh, even duct tape or gaffer's tape can shield a contactless card. This reduces convenience, of course. I have an Oyster Card for the London Underground, and it is very convenient to be able to pull my wallet from my pocket, put the wallet on the yellow Oyster pad, and wait for the beep that says it's been registered. Nonetheless, this becomes a risk-versus-convenience tradeoff that I can easily adjust and test.

RFID-enabled passports [2] are being produced with a mesh in the cover to shield the contactless card in the passport. The first generation of cover shielding has not been as good as it could be [6], but no doubt it will get better. A secondary risk is that if the passport is partially open, then there is additional leaking. Fortunately, the amount of leakage is proportional to the sine of the angle formed by the passport cover, but there will always be leaks.

4   Data Security at a Distance

The obvious defense against the obvious leaks is to throw some cryptography and protocol design against an attacker and hope that defeats reading the card from a distance. In practice this has failed for a variety of reasons, including weak cryptography[1], keys derived from easily deduced information [5], or the ability for an attacker to create man-in-the-middle or relay attacks on the card's legitimate use [4].

5   Tracking

Even if the data is properly encrypted, there is still the possibility of using the device for tracking. For example, an RFID passport has the passport data encrypted, but this could be used to track the person who carries it. The contents of the data are irrelevant, the attacker simply identifies the owner with the encrypted data and uses that as a database index. Ideally, an RFID device would not give a constant answer to thwart simple tracking.

6   Presence Security

Since an RFID device, even with a completely secure protocol, will respond to a signal when it is given, this creates a difficult presence vulnerability, which I have also called the "one-bit attack" because the content of the response is irrelevant. The fact that the device has responded at all enables the attack. Using this, a thief can find car keys, passport, or credit cards in a vacant hotel room. Or more insidiously, use the passport that someone carries to trigger a bomb. At Black Hat in 2005, I described this possibility to Kevin Mahaffey of Flexilis, who implemented a proof of concept [6].

7   Appropriate Use of Technology

All of these problems mean that a systems designer must take into consideration whether the use of an RFID is an appropriate use of technology. Can the RFID be replaced with something else that has fewer downsides. For example, in the case of RFID passports, there is no variable data in it. It could be replaced with a two-dimensional barcode or a simple contact smart card with little or no computing power. The RFID system has vulnerabilities that require a series of additional protections including radio shielding, encryption, and access protocols. All of these protections are necessary because the RFID can be silently and surreptitiously read from a distance.

References

1. Bono, S., Green, M., Stubblefield, A., Juels, A., Rubin, A., Szydlo, M.: Security Analysis of a Cryptographically-Enabled RFID Device, http://www.rsa.com/rsalabs/staff/bios/ajuels/publications/pdfs/DSTbreak.pdf
2. Federal Register FR Doc 05-21284 (October 2005), http://edocket.access.gpo.gov/2005/05-21284.htm
3. Juels, A., Molnar, D., Wagner, D.: Security and Privacy Issues in E-passports IACR Cryptology ePrint Archive (2005), http://eprint.iacr.org/2005/095
4. Kfir, Z., Wool, A.: Picking Virtual Pockets using Relay Attacks on Contactless Smartcard Systems. IACR Cryptology ePrint Archive (2005), http://eprint.iacr.org/2005/052
5. Mahaffey, K.: RFID Passport Implementation Vulnerabilities: Technical Analysis, http://www.flexilis.com/download/RFIDPassportTechnicalAnalysis.pdf
6. Mahaffey, K., et al.: RFID Passport Shield Failure Demonstration: FLX[2006?0605] Video Security Brief, http://www.flexilis.com/download/RFIDPassportShieldFailureDemonstration.pdf and Demonstration movie http://www.youtube.com/watch?v=-XXaqraF7pI

Position Statement in RFID S&P Panel: From Relative Security to Perceived Secure

Yvo Desmedt⋆

Department of Computer Science, University College London
Gower Street, London WC1E 6BT, United Kingdom
y.desmedt@cs.ucl.ac.uk

Abstract. RFID is now in fashion. Exactly 20 years ago it was pointed out that identification based on electronic tokens suffer from the middle-man attack. So, obviously RFIDs do too. Worse, the middleman attack is even easier to set up. Privacy advocates have expressed concerns about the use of RFIDs. Two implementations are compared: the use of RFID cards in the underground in Shanghai (similarly for Singapore) and the use in the London system. We conclude that privacy concerns can some-times be addressed succesfully. We also address reliabilty concerns since RFID cards are easy to break. Finally we address the psychological issue that RFIDs are believed to be secure.

1 The Mafia Attack: A 20 Year Old Weakness

20 years ago academics [7] (see also [3]) pointed out that any electronic token when used for identification is vulnerable to the following scenario [7, p. 26]:

> B is the owner of a mafia-owned restaurant, C is a member of the same mafia-gang and D is a jeweller. A and D are not aware of the following fraud. At the moment that A is ready to pay and ready to prove his identity to B, B informs C that the fraud is starting. This is done by using a secret radio-link between C and B. The identification card of C communicates also, using such a radio-link, with the equipment of B. At this point, C makes his choice of the diamond he wants to buy and so D is starting to check "C's" (in fact A's) identity. While D is checking the identity, C and B's role is only to sit in the middle between A and D. So B and C pass all questions and all answers related to the mathematical part of the identification going from D to A and vice-versa. So even if D is aware that an identification procedure over the telephone could not work, another person could come physically to his store and D would not be aware that he is remotely checking A's identity.

Note that 20 years ago RFID cards were not popular. The aforementioned sce-nario, which the authors called the *mafia fraud*, was described for cards that require contact, such as old-fashioned chip-cards.

So, one can wonder what did happen during the last 20 years. Obviously, the attack is much easier to mount against RFIDs than against cards that require contact. Indeed, in the case of RFIDs, B does not need to receive the card, as in the case of chip-cards. For RFIDs, B can just approach A close enough. So, B does not need to have access to the card reader.

Although this man-in-the-middle attack in the context of indentification has been reinvented many times and often given different names, such as relay attack, the 20 year old prior research demonstrates that the attack is not typical to RFIDs, but just easier to mount. Solutions against this type of attack have for example been proposed in [3,4] (see also [5]).

Experiments to verify the ease of mounting such attacks have been done in different laboratories, e.g. [2]. In these experiments, the attacker B needs to power up the RFID, limiting the accessibility to A's card. The question is whether such an attack can be mounted while a legitimate third party, let say S, provides the power. As an example, when A uses its card, e.g. in shop S, then A's card is powered up by S's reader. The question is whether at that time B can mount the attack. If so, then the distance between A and B could be significantly larger. Such an attack would further undermine the security of RFIDs.

2   Is Privacy a Real Concern?

In this section we will mainly make two points, being that:

1. privacy issues depend on the implementation. Systems based on RFID can be designed to deal with privacy issues.
2. the breach of privacy due to poorly implemented RFID systems should be evaluated relative to the other violations of privacy in our society. We question whether in societies that are already running a big brother state the violations of privacy due to RFID may be relatively low to the other breaches.

We illustrate above comparing two underground (metro) systems.

We start by describing the use of RFID in the Shanghai Metro & Light Rail system. When passengers buy a ticket they obtain an RFID card. They can pay cash to buy this ticket. When they leave the underground station the machine that checks tickets eats your card. Cards are then reused.

The London tube system uses a combination of paper tickets with a magnetic stripe and RFID cards (called Oyster cards). First of all Oyster cards are heavily promoted. Indeed, paper tickets can cost the double of the price. However, to be able to buy Oyster cards passengers need to provide their name, their postcode, their home phone number and their e-mail address [9].

Let us now discuss the impact on privacy. Trivially, the Shanghai RFID card system could be used to know from which station to what destination it was used. However, linking the passenger with the card used is much more difficult than in London. So, for all practical purposes the Shanghai underground RFID system does not affect the privacy of its users. On the other hand the London system is big brothers dream come true. Indeed, in the London one the information

about the travel can be linked to the individual. Moreover, in London it is illegal to have someone else borrow your card. Note that the Singapore Mass Rapid Transit (SMRT) has from privacy viewpoints similar properties for single ride tickets to the one in Shanghai. Moreover in Singapore the General Ticketing Machines (GTMs) can be used to top-up cards using cash [8]. So, a properly designed RFID system has a minimal effect on one's privacy.

We now analyze the *relative* impact of poorly implemented use of RFID on privacy overall. Continuing with the example of the London Tube system. As is well known Britain is the country in the world with the most security cameras per habitant. Moreover, the underground stations of the tube have plenty of cameras. However, it should be pointed out that even with today's technology RFIDs (when no fraud is being committed) allow a more accurate identification than when using pattern recognition applied to images from cameras, in particular analog ones. However, cell phones (or mobiles as they are called in the UK) allow to locate an individual with an accuracy of a few meter (or even better). So, these concerned about violations by the authorities of their privacy should look at the bigger picture. Evidently, the use of RFIDs may, in some cases, allow outsiders who do not have access to the cell phone location facility, to obtain private information about its bearer. One should note that other technology besides RFID can be used to covertly collect private information (see, e.g. [6]).

3   Availability and Reliabilty

Many RFID cards, such as Oyster cards are easy to destroy [1]. Sitting on them may break them. New passports in several countries are now by default equipped with an RFID. When leaving Australia, the immigration officer will scan the RFID tag of such passports. The author of this paper has traveled twice with such a passport to Australia. Each time the scanner was unable to read the RFID tag. A backup system, set up in a different location than the normal immigration booths, was then used. Each time (so far) the backup RFID scanner was able to read the tag. It is not clear to the author what will happen when the RFID in his passport no longer works. Obviously, since RFIDs are not extremely reliable, one should caution users to rely on these!

4   Conclusion: Why RFIDs?

RFIDs seem to increase security compared to other means of identification. However, they make the man-in-the-middle attack easier. There are privacy concerns about these, although they may be dealt with, or, at least in some case, there are worse threats to privacy. Moreover, their reliabilty is questionable. So, one can wonder why RFID.

It seems that a clear advantage of RFIDs is the ease of revocation. Take the example of mechanical keys to enter buildings and rooms. Although more reliable and providing more privacy, these keys are hard to revoke. However, the same property is true for cards that require to make a contact.

Based on the wide deployment of RFIDs it seems that there is the believe that they increase security. In a society in which, accordingly to the Pope [10], God has been replaced by science, it seems rather strange that unscientific believes on RFIDs are being accepted. However, as researchers in economics know (see the work of Nobel Laureate Kahneman), there is a link between psychological and Behavioral Economics. His theory states that humans will not make the logical economic decision, but will be biased based on psychological factors. It seems that the same is true in information security. This seems to introduce the need to study the new(?) discipline which could be called "Psychological Aspects of Security." It should be noted that Microsoft already succeeded in convincing its users that it is good for computers to reboot them.

Finally we conclude by stating that dismissing RFID in general may not be the correct statement: the Singapore's and Shanghai's metro systems illustrate this.

References

1. AlFaraj, A.: Personal communication (2006)
2. AlFaraj, A.: Rfid insecurity for entity authentication. Master's thesis University College London, Computer Science (2006)
3. Bengio, S., Brassard, G., Desmedt, Y.G., Goutier, C., Quisquater, J.-J.: Secure implementations of identification systems. Journal of Cryptology 4, 175–183 (1991)
4. Beth, T., Desmedt, Y.: Identification tokens — or: Solving the chess grandmaster problem. In: Menezes, A.J., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 169–176. Springer, Heidelberg (1991)
5. Brands, S., Chaum, D.: Distance-bounding protocols. In: Helleseth, T. (ed.) EU-ROCRYPT 1993. LNCS, vol. 765, pp. 344–359. Springer, Heidelberg (1994)
6. Desmedt, Y.: Establishing Big Brother using covert channels and other covert techniques. In: Anderson, R. (ed.) Information Hiding. LNCS, vol. 1174, pp. 65–71. Springer, Heidelberg (1996)
7. Desmedt, Y., Goutier, C., Bengio, S.: Special uses and abuses of the Fiat-Shamir passport protocol. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 21–39. Springer, Heidelberg (1988)
8. Smrt, tickets, http://www.smrt.com.sg/trains/tickets.html
9. Transport for London, Get Oyster Card, https://sales.oystercard.com/oyster/lul/guestFirstIssue.do
10. Pope Benedict XVI: Address of his Holiness Benedict XVI to the Members of the Pontifical Academy of Sciences (November 6, 2006), http://www.vatican.va/holy_father/benedict_xvi/speeches/2006/november/documents/hf_ben-xvi_spe_20061106_academy-sciences_en.html

A Model of Onion Routing with Provable Anonymity

Joan Feigenbaum[1,⋆], Aaron Johnson[1,⋆⋆], and Paul Syverson[2,⋆⋆⋆]

[1] Yale University
{Joan.Feigenbaum, aaron.johnson}@yale.edu
[2] Naval Research Laboratory
syverson@itd.nrl.navy.mil

Abstract. Onion routing is a scheme for anonymous communication that is designed for practical use. Until now, however, it has had no formal model and therefore no rigorous analysis of its anonymity guarantees. We give an IO-automata model of an onion-routing protocol and, under possibilistic definitions, characterize the situations in which anonymity and unlinkability are guaranteed.

Keywords: Security, privacy, anonymity, onion routing.

1 Introduction

Anonymity networks allow users to communicate while hiding their identities from one another and from third parties. We would like to design such networks with strong anonymity guarantees but without incurring high communication overhead or much added latency. Many designs have been proposed that meet these goals to varying degrees [1].

Of the many design proposals, onion routing [8] has had notable success in practice. Several implementations have been made [8,13,6], and there was a similar commercial system, Freedom [2]. As of September 2006, the most recent iteration of the basic design, Tor [6], consists of over 750 routers, each processing an average of 100KB/s. Onion routing is a practical anonymity-network scheme with relatively low overhead and latency. It provides two-way, connection-based communication and does not require that the destination participate in the anonymity-network protocol. These features make it useful for anonymizing much of the communication that takes place over the Internet today, such as web browsing, chatting, and remote login.

Many Tor users communicate with web-based businesses and financial services. Chaum [4] was the first to note that even the best ecash design fails to be anonymous if the network identifies the customer. Even if a client is not hidden from the service, *e.g.*, she's using ordinary credit cards, she may desire privacy

---

from her network-service provider, which might be her employer or just an ISP that is not careful with logs of its users' activities. Examples of the threat posed by both of these situations have been all too frequent in the news. Businesses also make integral use of Tor to protect their commercial interests from competitors or to investigate the public offerings of their competitors without being observed. One vendor discovered by using Tor that its competitor had been offering a customized web site just for connections from the vendor's IP address.

Low latency and other performance characteristics of Tor can be demonstrated experimentally; anonymity-preserving properties cannot. Also, even with careful design, vulnerabilities can persist. The initial Tor authentication protocol had a cryptographic-assumption flaw that left it open to man-in-the-middle attacks. The revised authentication protocol was then proven to be free of such flaws [7]. As Tor is increasingly relied upon for sensitive personal and business transactions, it is increasingly important to assure its users that their anonymity will be preserved. Long-established components of such assurance in system security include a formal model, proving security guarantees in that model, and arguing that the model captures essential features of the deployed system. These are what we provide in this paper.

An onion-routing network consists of a set of onion routers and clients. To send data, a client chooses a sequence of routers, called a *circuit*, and constructs the circuit using the routers' public keys. During construction, a shared symmetric key is agreed upon with each router. Before sending data, these keys are used to encrypt each packet once for each router in the circuit and in the reverse of the order that the routers appear in the circuit. Each router uses its shared key to decrypt the data as it is sent down the circuit so it is fully decrypted at the end. Data flowing up to the client has a layer of encryption added by each onion router, all of which are removed by the client. The layered encryption helps hide the data contents and destination from all but the last router and the source from all but the first router. The multiple encryption and decryption also makes it harder for an observer to follow the path the data takes through the network.

Anonymity has not yet been rigorously proven of onion routing. We thus propose a formal model of onion routing based on the Tor protocol and analyze the anonymity it provides. Our model is expressed using IO automata [9], which provide us with asynchronous computation and communication. We then suggest definitions of anonymity and unlinkability with respect to an adversary within this model. The adversary is local and active in that he controls only a subset of the routers but can perform any arbitrary computation with them. This is the adversary against which Tor was designed to protect [6] and is a good adversary model for the situation in which Tor is currently running. Finally, we provide necessary and sufficient conditions for anonymity and unlinkability to be provided to a user in our model. It should be noted that we do not analyze the validity of the cryptography used in the protocol and instead base our proofs on some reasonable assumptions about the cryptosystem.

We only consider possibilistic anonymity here. An action by user u is considered to be anonymous when there exists some system in which u doesn't perform the action, and that system has an execution that is consistent with what the adversary sees. The actions for which we consider providing anonymity are sending messages, receiving messages, and communicating with a specific destination. More refined definitions of anonymity, [12,5], incorporate probability. We leave for future work applying such definitions to our system, which could be done by defining a probability measure over executions or initial states. Also, for simplicity, the model includes almost no concept of time. We do add circuit identifiers to mimic an attacker's ability to do timing attacks: the observation of distinctive timing patterns in a stream of data, whether inherent or attacker-induced. There is no timestamp included with actions, though, so there is only an ordering on actions.

The main result we show is that the adversary can determine a router in a given user's circuit if and only if it controls an adjacent router, with some other minor conditions. In particular, the adversary can determine which user owns a circuit only when the adversary controls the first hop. The set of users which have an uncompromised first hop form a sender "anonymity set," among which the adversary cannot distinguish. Similarly, the adversary can determine the last router of a circuit only when it controls it or the penultimate router. Such circuits provide receiver anonymity. Also, a user is "unlinkable" to his destination when he has receiver anonymity or his sender anonymity set includes another sender with a destination that is different or unknown to the adversary.

The first-hop/last-hop attack is well-known [13], but we state it in full detail and show that, in a reasonable formal model, it is the only attack that an adversary can mount. Also, our results persist with or without some of the nontrivial design choices, such as multiple encryption and stream ciphers. This doesn't imply that these features are unnecessary – they may make attacks more difficult in practice and meet other goals such as data integrity, but it does illuminate their effect on the security of the protocol. Finally, we present the first formal network model and protocol definition for onion routing, and give definitions for anonymity and unlinkability within that model.

2   Related Work

Numerous papers have informally discussed the security of the design of onion routing and related systems, as well as theoretical and experimentally demonstrated attacks. There have also been numerous formalizations of anonymous communication. However, formal analyses have primarily been of systems other than onion routing, *e.g.*, DC nets and Crowds. (Cf. [1] for examples of all of these.)

Recent papers have formalized systems similar to onion routing but without persistent circuits. Camenisch and Lysyanskaya [3] prove that the cryptography their protocol uses doesn't leak any information to nodes in the path other than the previous and next nodes, but leave open what anonymity it provides. This

question is answered in part by Mauw et al. [10], who formalize a similar connectionless protocol in an ACP-style process algebra and, under a possibilistic definition of anonymity, show that it provides sender and receiver anonymity against a global passive adversary. Cryptography is dealt with using high level assumptions, similar to our work. Their model and analysis has much in common with this paper, but it does differ in several important ways. First, the protocol they investigate is connectionless: each data "onion" stores the identities of the routers it will pass through. This is significantly different from onion routing, which is circuit-based. Second, the analysis is done with respect to a passive adversary, which exhibits only a subset of the behavior of an active adversary. Third, in their model agents choose destinations asynchronously and the adversary must take into account every onion he has seen when trying to break anonymity. In our model all agents choose a single destination, which gives the adversary more power. In some ways, our work extends theirs, and several of the differences noted here appear in [10] as suggestions for further research.

3   Model

3.1   Distributed System

Our model of onion routing is based on IO automata [9]. This formalism allows us to express an onion-routing protocol, model the network, and make precise the adversary's capabilities. One of its benefits is that it models asynchronous computation and communication. Another is that every action is performed by a single agent, so the perspective of the adversary is fairly clear.

We model onion routing as a fully connected asynchronous network of IO automata. The network is composed of FIFO channels. There is a set of users U and a set of routers R. Let $N = U \cup R$. The term *agent* refers to any element of N. It is possible that $U \cap R \neq \emptyset$. In this case, user and router automata exist on the same processor. We assume that the users all create circuits of a fixed length l. (In the current Tor network, $l = 3$.) Each router-and-user pair shares a set of secret keys; however, the router does not know which of its keys belong to which user. This separates, for now, key distribution from the rest of the protocol. We assume that all keys in the system are distinct. Let K be the keyspace. The triple (u, r, i) will refer to the ith key shared by user u and router r.

Let P be the set of control messages, and $\bar{P}$ be the extension of P by encryption with up to l keys. The control messages will be tagged with a link identifier and circuit identifier when sent, so let the protocol message space be $M = \mathbf{N}_+ \times \mathbf{N}_+ \times \bar{P}$. We denote the encryption of $p \in P$ using key k with $\{p\}_k$, and the decryption with $\{p\}_{-k}$. For brevity, the multiply encrypted message $\{\{p\}_{k_1}\}_{k_2}$ will be denoted $\{p\}_{k_1,k_2}$. Brackets will be used to indicate the list structure of a message (*i.e.* $[p_1, p_2, \ldots]$).

The adversary in our system is a set of users and routers $A \subseteq N$. The adversary is active in the sense that the automata running on members of A are completely arbitrary. We call an agent a *compromised* if $a \in A$

3.2   Automata

We give the automata descriptions for the users and routers that are based on the Tor protocol [6]. We have simplified the protocol in several ways. In particular we do not perform key exchange, do not use a stream cipher, have each user construct exactly one circuit to one destination, do not include circuit teardowns, eliminate the final unencrypted message forward, and omit stream management and congestion control. We are also using circuit identifiers to mimic the effect of a timing attack. Section 4.7 discusses the effects of changing some of these features of our protocol.

During the protocol each user u iteratively constructs a circuit to his destination. u begins by sending the message $\{CREATE\}_{k_1}$ to the first router, r_1, on his circuit. The message is encrypted with a key, k_1, shared between u and r_1. r_1 identifies k_1 by repeatedly trying to decrypt the message with each one of its keys until the result is a valid control message. It responds with the message CREATED. (Note that this is different from the implemented Tor protocol, in which the CREATE message would be encrypted with the public key for r_1 rather than one of the shared keys it holds.) Given a partially-constructed circuit, u adds another router, r_i, to the end by sending the message $\{[EXTEND, r_i, \{CREATE\}_{k_i}]\}_{k_{i-1},\ldots,k_1}$ down the circuit. As the message gets forwarded down the circuit, each router decrypts it. r_{i-1} performs the CREATE steps described above, and then returns the message $\{EXTENDED\}_{k_{i-1}}$. Each router encrypts this message as it is sent back up the circuit.

Link identifiers are used by adjacent routers on a circuit to differentiate messages on different circuits. They are unique to the adjacent pair. Circuit identifiers are also included with each message and identify the circuit it is traveling on. They are unique among all circuits. Circuit identifiers are not used in the actual Tor protocol, and their only purpose here is to represent the ability of an adversary to insert and/or detect timing patterns in the traffic along a circuit. This reflects in our model the very real threat of timing attacks [11]. It has the added advantages of making it clear when this power is used and of being easy to remove in future model adjustments.

The user automaton's state consists of the sequence of routers in its circuit, a number that identifies its circuit, and a number that indicates the state of its circuit. We consider the final router in the circuit to be the destination of the user. The user automaton runs two threads, one to extend a circuit that is called upon receipt of a message and the other to start circuit creation that is called at the beginning of execution. We express these in pseudocode rather than IO automata, but note that the state changes in a particular branch occur simultaneously in the automaton. $k(u, c, b)$ refers to the key used by user u with router c_b in the bth position in circuit c. The automaton for user u appears in Automaton 1.

The router automaton's state is a set of keys and a table, T, with a row for each position the router holds in a circuit. Each row stores the previous and next hops in the circuit, identifying numbers for the incoming and outgoing links, and the associated key. There is only one thread and it is called upon receipt of a

Automaton 1. User u

1: $c \in \{(r_1, \ldots, r_l) \in R^l | \forall_i r_i \neq r_{i+1}\}$; init: arbitrary $\triangleright$ User's circuit
2: $i \in \mathbf{N}$; init: random $\triangleright$ Circuit identifier
3: $b \in \mathbf{N}$; init: 0 $\triangleright$ Next hop to build
4: **procedure** START
5:     SEND($c_1, [i, 0, \{CREATE\}_{k(u,c,1)}]$)
6:     $b = 1$
7: **end procedure**
8: **procedure** MESSAGE(msg,j) $\triangleright$ $msg \in M$ received from $j \in N$
9:     **if** $j = c_1$ **then**
10:         **if** $b = 1$ **then**
11:             **if** $msg = [i, 0, CREATED]$ **then**
12:                 $b++$
13:                 SEND($c_1, [i, 0, \{[EXTEND, c_b, \{CREATE\}_{k(u,c,b)}]\}_{k(u,c,b-1),\ldots,k(u,c,1)}]$)
14:             **end if**
15:         **else if** $b < l$ **then**
16:             **if** $msg = [i, 0, \{EXTENDED\}_{k(u,c,b-1),\ldots,k(u,c,1)}]$ **then**
17:                 $b++$
18:                 SEND($c_1, [i, 0, \{[EXTEND, c_b, \{CREATE\}_{k(u,c,b)}]\}_{k(u,c,b-1),\ldots,k(u,c,1)}]$)
19:             **end if**
20:         **else if** $b = l$ **then**
21:             **if** $msg = [i, 0, \{EXTENDED\}_{k(u,c,b-1),\ldots,k(u,c,1)}]$ **then**
22:                 $b++$
23:             **end if**
24:         **end if**
25:     **end if**
26: **end procedure**

message. In the automaton for router r, we denote the smallest positive integer that is not being used on a link from r to q or from q to r as $minid(T, q)$. The automaton for router r appears in Automaton 2.

3.3   System Execution

We use standard notions of *execution* and *fairness*. An execution is a possible run of the network given its initial state. Fairness for us means that any message an automaton wants to send will eventually be sent and every sent message is eventually received.

   We introduce the notion of a *cryptographic* execution. This is an execution in which no agent sends a control message encrypted with active keys it doesn't possess before it receives that message. We restrict our attention to such executions, and must require our encryption operation to prevent an attacker from outputting a control message, with more than negligible probability, when it is encrypted with keys he doesn't possess. This is reasonable because we can easily create a ciphertext space that is much larger than the rather limited control message space P. Note that this precludes the use of public key encryption to

---

Automaton 2. Router r

1: $keys \subseteq K$, where $|keys| \geq |U| \cdot \lceil \frac{l}{2} \rceil$; init: arbitrary $\qquad \triangleright$ Private keys
2: $T \subset N \times \mathbf{N} \times R \times \mathbf{Z} \times keys$; init: $\emptyset$ $\qquad \triangleright$ Routing table
3: **procedure** MESSAGE($[i, n, p], q$) $\qquad \triangleright [i, n, p] \in M$ received from $q \in N$
4: $\quad$ **if** $[q, n, \emptyset, -1, k] \in T$ **then** $\qquad \triangleright$ In link created, out link absent
5: $\qquad$ **if** $\exists_{s \in R - r, b \in P} p = \{[\text{EXTEND}, s, b]\}_k$ **then**
6: $\qquad\quad$ SEND($s, [minid(T, s), b]$)
7: $\qquad\quad$ $T = T - [q, n, \emptyset, -1, k] + [q, n, s, -minid(T, s), k]$
8: $\qquad$ **end if**
9: $\quad$ **else if** $[s, m, q, -n, k] \in T$ **then** $\qquad \triangleright$ In link created, out link initiated
10: $\qquad$ **if** $p = \text{CREATED}$ **then**
11: $\qquad\quad$ $T = T - [s, m, q, -n, k] + [s, m, q, n, k]$
12: $\qquad\quad$ SEND($s, [i, m, \{\text{EXTENDED}\}_k]$)
13: $\qquad$ **end if**
14: $\quad$ **else if** $\exists_{m>0}[q, n, s, m, k] \in T$ **then** $\qquad \triangleright$ In and out links created
15: $\qquad$ SEND($s, [i, m, \{p\}_{-k}]$) $\qquad \triangleright$ Forward message down the circuit
16: $\quad$ **else if** $[s, m, q, n, k] \in T)$ **then** $\qquad \triangleright$ In and out links created
17: $\qquad$ SEND($s, [i, m, \{p\}_k]$) $\qquad \triangleright$ Forward message up the circuit
18: $\quad$ **else**
19: $\qquad$ **if** $\exists_{k \in keys} p = \{\text{CREATE}\}_k$ **then** $\qquad \triangleright$ New link
20: $\qquad\quad$ $T = T + [q, n, \emptyset, -1, k]$
21: $\qquad\quad$ SEND($q, [i, n, \text{CREATED}]$)
22: $\qquad$ **end if**
23: $\quad$ **end if**
24: **end procedure**

---

encrypt the packets because such messages can easily be constructed with the public keys of the routers.

Definition 1. *An* execution *is a sequence of states of an IO automaton alternating with actions of the automaton. It begins with an initial state, and two consecutive states are related by the automaton transition function and the action between them. Every action must be* enabled, *meaning that the acting automaton must be in a state in which the action is possible at the point the action occurs.*

Definition 2. *A finite execution is* fair *if there are no actions enabled in the final state. Call an infinite execution* fair *if every output action that is enabled in infinitely many states occurs infinitely often.*

Definition 3. *An execution is* cryptographic *if an agent sends a message containing $\{p\}_{k_1,\ldots,k_i}$ only when it possesses all keys $k_1, \ldots, k_i$, or when for the largest j s.t. the agent does not possess k_j, $1 \leq j \leq i$, the agent has already received a message containing $\{p\}_{k_1,\ldots,k_j}$.*

3.4 Distinguishability

Definition 4. *A configuration $C : U \rightarrow \{(r_1, \ldots, r_l, n) \in R^l \times \mathbf{N}_+ | \forall_i r_i \neq r_{i+1}\}$ maps each user to the circuit and circuit identifier in his automaton state.*

The actions we want to be performed anonymously are closely related to the circuits the users try to construct during an execution. In our model, all messages are sent along links of a circuit; these messages are all circuit-creation messages and thus are entirely determined by the circuit, so the sender or receiver of a given message corresponds directly to the path of the circuit. Therefore, in order to prove that certain actions are performed anonymously in the network, we can just show that the adversary can never determine this circuit information. This is a possibilistic notion of of anonymity. We do this by identifying classes of adversary-indistinguishable configurations.

Because $i \in N$ only sees those messages sent to and from i, an execution of a configuration C may appear the same to i as a similar execution of another configuration D that only differs from C in parts of the circuits that are not adjacent to i and in circuit identifiers that i never sees. To be assured that i will never notice a difference, we would like this to be true for all possible executions of C. These are the fair cryptographic executions of C, and likewise the executions of D should be fair and cryptographic. We will say that these configurations are indistinguishable if, for any fair cryptographic execution of C, there exists a fair cryptographic execution of D that appears identical to i, *i.e.* in which i sends and receives what appear to be the same messages in the same order.

Agent i's power to distinguish among executions is weakened by encryption in two ways. First, we allow a permutation on keys to be applied to the keys of encrypted or decrypted messages in an execution. This permutation can map a key from any router other than i to any other key of any other router other than i, because i can only tell that it doesn't hold these keys. It can map any key of i to any other key of i, because i doesn't know for which users and circuit positions its keys will be used. Second, i cannot distinguish among messages encrypted with a key he does not possess, so we allow a permutation to be applied to control messages that are encrypted with a key that is not shared with i. This second requirement must be justified by the computational intractability of distinguishing between encrypted messages with more than a negligible probability in our cryptosystem.

Definition 5. *Let D_A be a relation over configurations indicating which configurations are indistinguishable to $A \subseteq N$. For configurations C and C', $C \sim_{D_A} C'$ if for every fair cryptographic execution α of C, there exists some action sequence β s.t. the following conditions hold with C' as the initial state:*

1. *Every action of β is enabled, except possibly actions done by members of A.*
2. *β is fair for all agents, except possibly those in A.*
3. *β is cryptographic for all agents.*
4. *Let Ξ be the subset of permutations on the active keyspace $U \times R \times \lceil \frac{l}{2} \rceil$ s.t. each element restricted to keys involving $a \in A$ is a permutation on those keys. We apply $\xi \in \Xi$ to the encryption of a message sequence by changing every list component $\{p\}_{(u,r,i)}$ in the sequence to $\{p\}_{\xi(u,r,i)}$.*
   Let Π be the subset of permutations on $\bar{P}$ s.t. for all $\pi \in \Pi$, π is a permutation on the set $\{\{p\}_{k_1,...,k_i}\}_{p \in P}$, and $\pi(\{p\}_{k_1,...,k_i,k_a}) = \pi(\{p\}_{k_1,...,k_i})$ when

k_a is shared by the adversary. We apply $\pi \in \Pi$ to a message sequence by changing every message $\{p\}_{k_1,\ldots,k_i}$ in the message sequence to $\pi(\{p\}_{k_1,\ldots,k_i})$. Then there must exist $\xi \in \Xi$ and $\pi \in \Pi$ s.t. applying ξ and π to the subsequence of α corresponding to actions of A yields the subsequence of β corresponding to actions of A.

If $C \sim_{D_A} C'$, we say that C is *indistinguishable* from C' to A. It is clear that an indistinguishability relation is reflexive and transitive.

3.5   Anonymity and Unlinkability

The sender in this model corresponds to the user of a circuit, the receiver to the last router of the circuit, and the messages we wish to communicate anonymously are just the circuit control messages. The circuit identifiers allow the adversary to link together all the messages initiated by a user and attribute them to a single source. (Recall that in our model, users open a single circuit to a unique destination at one time.) Therefore sender anonymity is provided to u if the adversary can't determine which circuit identifier u is using. Similarly, receiver anonymity is provided to r for messages from u if the adversary can't determine the destination of the circuit with u's identifier. Also, unlinkability is provided to u and r if the adversary can't determine u's destination.

Definition 6. *User u has* sender anonymity *in configuration C with respect to adversary A if there exists some indistinguishable configuration C' in which u uses a different circuit identifier.*

Definition 7. *Router r has* receiver anonymity *on user u's circuit, in configuration C, and with respect to adversary A, if there exists some indistinguishable configuration C' in which a user with u's circuit identifier, if one exists, has a destination other than r.*

Definition 8. *User u and router r are* unlinkable *in configuration C if there is some indistinguishable configuration C' in which the destination of u is not r.*

4   Indistinguishable Configurations

Now we will show that sometimes the adversary cannot determine the path or identifier of a circuit. More specifically, an adversary can only determine which user or router occupies a given position in a circuit when the adversary controls it or a router adjacent to it on that circuit. Also, when the adversary controls no part of a circuit it cannot determine its identifier.

In order to do this, we must show that, given a pair of configurations (C, C') that are indistinguishable by these criteria, for every execution of C there exists an execution of C' that appears identical to the adversary. To do this we will start with a fair cryptographic execution of C, describe how to transform it, and prove that this transformed sequence forms a fair, cryptographic, and indistinguishable execution of C'. We will also show that a pair of configurations that are distinguishable by the described criteria allow no such transformation.

4.1   Message Sequences

To start, we observe that, in spite of the arbitrary actions of the adversary, the actions of the uncompromised users and routers in an execution are very structured. The protocol followed by the user and router automata defines a simple sequence of message sends and receives for every circuit. A user or router will only send a message as part of such a sequence.

The user subsequence consists of messages between the user and the first router on its circuit. The user u in configuration C begins the sequence by sending a CREATE message to $C_1(u)$, the first router in u's circuit in C; $C_1(u)$ responds with a CREATED message. Then for the remaining $l - 1$ routers on the circuit, the user sends an EXTEND message and $C_1(u)$ responds with an EXTENDED message. The user will only send a message as the next step in this sequence. Therefore we can take all actions in an execution by u and partition them into those that are part of this sequence and those that are not. Those that are not are "junk" receives that the adversary caused to be sent to u by not following the protocol.

A router performs a similar sequence when it is added to a circuit. The sequence begins when router r router receives a CREATE message from agent n with the smallest unused link identifier at that point between r and n. r responds with a CREATED message. Then r receives an EXTEND message with the identity of the next router q and an enclosed CREATE message. It passes the enclosed message on to q, receives a CREATED message back, and sends an EXTENDED message to n. Then r forwards up or down the circuit any further messages received. r will only send messages as part of such a sequence. We can therefore partition all actions by r in an execution into sequences of this type, in addition to a sequence for "junk" receives that aren't part of such a sequence and are a result of adversarial misbehavior. We will use the existence of such partitions of executions in our analysis.

4.2   Indistinguishable Users

Now we prove that an active adversary cannot determine which user creates a given circuit unless the first router on that circuit is controlled by the adversary or the owners of all the other circuits have been determined. That is, an adversary cannot distinguish between a configuration C and the configuration C' that is identical to C except for two circuits with uncompromised first routers that are switched between the circuit owners. In order to do so, we must show that, for any fair cryptographic execution of C, there exists some action sequence of C' satisfying the indistinguishability requirements of Definition 5. To do so, we simply swap between the switched users the messages that pass between them and the first routers on their circuits and switch the encryption keys of these messages.

Theorem 1. *Let u, v be two distinct users s.t. neither they nor the first routers in their circuits are compromised (*i.e., *are in A). Let C' be identical to C except the circuits of users u and v are switched. C is indistinguishable from C' to A.*

Proof Sketch: Let α be a fair cryptographic execution of C. To create a possible execution of C', first construct α' by replacing any message sent or received between u (v) and $C_1(u)$ $(C_1(v))$ in α with a message sent or received between v (u) and $C_1(u)$ $(C_1(v))$. Then let ξ be the permutation that sends u to v and v to u and other users to themselves. Create β by applying ξ to the encryption keys of α'.

To show that the actions in this sequence are enabled for uncompromised routers we observe that only message partitions for u, v, $C_1(u)$, and $C_1(v)$ have been changed. These are modified so that they remain valid partitions,. To show that the execution is fair we observe that no "new" valid messages or sequences can appear. The transformed sequence is cryptographic because the key permutations and message changes are applied to the entire sequence and the original sequence α was cryptographic. The permutation needed to make β look like α to A is just the reverse of the key permutation used to create β.   □

4.3   Indistinguishable Routers

Now we prove that an adversary cannot determine an uncompromised router on a given circuit unless it controls the previous or next router on that circuit. More formally, assume that the $(i-1)$st, ith, and $(i+1)$st routers of a user u's circuit in some configuration C are not compromised. We will show that C is indistinguishable from configuration C', where C' is identical to C except the ith router of u's circuit has been arbitrarily changed. The proof is similar to that of Theorem 1, although it is complicated by the fact that the identities of routers in a circuit are included in multiple ways in the circuit creation protocol. Given an execution of C, we identify those message that are part of the circuit creation sequence of the modified circuit and then change them to add a different router in the ith position. Then we show that, in the sense of Definition 5, from the adversary's perspective this sequence is indistinguishable from the original and could be an execution of C'.

Theorem 2. *Say there is some user $u \notin A$ s.t. u's circuit in C contains three consecutive routers, $r_{i-1}, r_i, r_{i+1} \notin A$. Let C' be equal to C, except r_i is replaced with r_i' in u's circuit, where $r_i' \notin A \cup \{r_{i-1}, r_{i+1}\}$. C' is indistinguishable from C to A. The same holds for uncompromised routers (r_i, r_{i+1}) if they begin u's circuit and are replaced with (r_i', r_{i+1}), or (r_{i-1}, r_i) if they end u's circuit and are replaced with (r_{i-1}, r_i').*

Proof Sketch: Let α be some fair cryptographic execution of C. Let $h(C(u), i)$ be the number of occurrences of the ith router in the circuit $C(u)$ among the first i routers. We modify α in steps to create an indistinguishable sequence β:

1. Replace all messages of the form $[\text{EXTEND}, r_i, \{\text{CREATE}\}_{u,r_i,h(C(u),i)}]$ with $[\text{EXTEND}, r_i', \{\text{CREATE}\}_{u,r_i',h(C'(u),i)}]$.
2. Consider the partitions of router r_{i-1}'s actions that each form a prefix of the sequence adding r_{i-1} to u's circuit as the $(i-1)$st router. Replace all

messages in these partitions that are to and from r_i with the same messages to and from r_i'. Modify the link identifiers on these messages so that they are the smallest identifiers in use between r_{i-1} and r_i' at that point in α. Increase link identifiers that are in use between r_{i-1} and r_i' to make room for these new connections and decrease link identifiers that are in use between r_{i-1} and r_i to fill in the holes created by the removed connections. Perform similar modifications for routers r_i and r_{i+1}.

3. Replace all keys of the form $(u, r_i, h(C(u), i))$ with $(u, r_i', h(C'(u), i))$. Increment as necessary the third component of the encryption keys used between u and r_i' to take into account that r_i' appears once more in $C'(u)$ than it does in $C(u)$. Also decrement as necessary the third component of the keys used between u and r_i to take into account that r_i appears once less in $C'(u)$ than it does in $C(u)$.

The actions in the transformed sequence are enabled because we convert the partitions involving r_i to involve r_i' instead, adjusting link and key numbering as needed to maintain global consistency. β is cryptographic because the key and message permutations used to create it are applied uniformly. The transformed execution is fair first because we modify partitions as a whole, so there are no partial unfinished sequences. Second, β is cryptographic, so no "junk" receives from the adversary could be valid messages in a transformed partition. An attack that under this reasoning can't occur is that a compromised router a can't send a valid EXTEND message directing the router r_i' at the end of u's partially-constructed circuit to create a link to a. Such a message, if a weren't prohibited from sending it, only enables router action when r_i' is the router at the end of the circuit, since another router would be using a different key. This would leave r_i' with an enabled action in β. Finally, the required permutations to make β appear like α to A are simply the reverse of those used to create β in the first place.   □

4.4   Indistinguishable Identifiers

Theorem 3. *Say there is some uncompromised user u s.t. all routers in $C(u)$ are uncompromised. Let C' be a configuration that is identical to C, except that u uses a different circuit identifier. C' is indistinguishable from C to A.*

Proof. Let α be a fair cryptographic execution of C. To create β, simply change every occurrence of u's circuit identifier in C ($C_{l+1}(u)$) to its identifier in C'. β is enabled, fair, and cryptographic for C' because no message containing $C_{l+1}(u)$ gets sent to the adversary in α and the protocol itself ignores circuit identifiers except to forward them on. It appears the same to A for the same reason.

4.5   Distinguishable Configurations

The relation D_A, when restricted to the transitive closure of pairs that are indistinguishable by Theorems 1–3, is symmetric, and therefore forms an equivalence relation. Let $\approx_{D_A}$ denote this relation.

We can easily tell which configurations are in the same equivalence class using a function ρ that reduces a circuit to an identifier, the compromised positions, and the positions adjacent to compromised positions. For convenience, in the following we take c_0 to refer to u.

Definition 9. *Let $\rho : U \times N^l \times \mathbf{N}_+ \times \mathcal{P}(N) \to \mathbf{N} \times \mathcal{P}(N \times \mathbf{N}_+)$ be:*

$$\rho(u, c, A) = \begin{cases} (c_{l+1}, \{(r,i) \in N \times \mathbf{N}_+ | c_i = r \wedge (c_{i-1} \in A \vee c_i \in A \vee c_{i+1} \in A)\}) \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{if } c_i \in A \text{ for some } i \\ (0, \emptyset) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{otherwise} \end{cases}$$

We overload this notation: $\rho(C)$ refers to the multiset formed from the circuits of configuration C adjoined with their user and reduced by ρ, i.e., $\rho(C) = \{\rho(u, C(u), A)|u \in U\}$. Thus ρ captures the indistinguishable features of a configuration according to Theorems 1, 2, and 3.

Now we show that the equivalence relation is in fact the entire indistinguishability relation and that Theorems 1, 2, and 3 characterize which configurations are indistinguishable.

Theorem 4. *If $C \sim_{D_A} D$ then $C \approx_{D_A} D$.*

Proof. We show the contrapositive. Suppose C and D are in different equivalence classes. Let the adversary run the automata prescribed by the protocol on the agents it controls. Let α and β be fair cryptographic executions of C and D respectively.

Partition the adversary actions of α into subsequences that share the same circuit identifier. There is at most one such partition for each circuit. Circuit positions that are created in the same partition belong to the same circuit. In each partition the adversary can determine the absolute location of a circuit position filled by a given compromised agent a by counting the total number of messages it sees after the initial CREATE. Clearly A can also determine the agents that precede and succeed a on the circuit and the circuit identifier itself. Therefore A can determine the reduced circuit structure $\rho(C)$ from α.

The adversary can use β in the same way to determine $\rho(D)$. It is easy to see that $C \approx_{D_A} D$ if and only if $\rho(C) = \rho(D)$, so $\rho(C) \neq \rho(D)$. Therefore A can always distinguish between C and D.

4.6   Anonymity

The configurations that provide sender anonymity, receiver anonymity, and unlinkability follow easily from Theorems 1, 2, 3, and 4. In the following, let u be a user, C be a configuration, r be a router, and A be the adversary.

Corollary 1. *u has sender anonymity in C with respect to A if and only if at least one of two cases holds. The first is that u and $C_1(u)$ are uncompromised and there exists another user $v \neq u$ s.t. v and $C_1(v)$ are uncompromised. The second is that u and all $C_i(u)$ are uncompromised.* □

Corollary 2. *r has receiver anonymity on u's circuit in C with respect to A if and only if at least one of two cases holds. The first is that u, r, and $C_{l-1}(u)$ are uncompromised and there exists another router $q \neq r$ s.t. q is uncompromised. The second is that u and all $C_i(u)$ are uncompromised.* □

Corollary 3. *u and r are unlinkable in C with respect to A if and only if at least one of two cases holds. The first is that u, r, and $C_{l-1}(u)$ are uncompromised, and there exists another router $q \neq r$ s.t. q is uncompromised. For the second case it must be that u and $C_1(u)$ are uncompromised and there exists another user $v \neq u$ s.t. v and $C_1(v)$ are uncompromised. Also, it must be that $C_l(v) \neq r$, or $C_{l-1}(v)$ and r are uncompromised and there exists another router $q \neq r$ s.t. q is uncompromised.*   □

4.7   Model Changes

We chose the described protocol to balance two goals. The first was to accurately model Tor. The second was to make it simple enough to be analyzed, and so that the main ideas of the analysis weren't unnecessarily complicated. Our results are robust to changes of the protocol, however. We can make the protocol simpler by removing multiple encryption and the circuit identifiers without weakening the indistinguishability results. Single encryption does allow the adversary to easily link entries in his routers by sending messages along the circuit. This power is already available in our model from circuit identifiers, though. In the other direction, we can make it more complicated with a stream cipher and multiple circuits per user without weakening the distinguishability results.

Stream ciphers are used in the Tor protocol and prevent signaling along a circuit using dummy messages. Sending such messages will throw off the counter by some routers on the circuit and the circuit will stop working. We can model a stream cipher by expressing the encryption of the ith message p with key k as $\{p\}_{(k,i)}$, and allowing a different permutation to be applied for every pair (k, i). This can only increase the size of the configuration indistinguishability relation. However, the proof for the distinguishability of configurations only relies on the ability of the adversary to decrypt using his keys, count messages, and recognize the circuit identifier. Therefore it still holds when the model uses a stream cipher.

Allowing users to create multiple circuits doesn't weaken the adversary's power to link together its circuit positions and determine their position, but the number of configurations that are consistent with this view does in some cases increase. Let users create an arbitrary number of circuits. The adversary can still link positions and count messages as before, so the adversary can distinguish configurations C and D if $\rho(C) \neq \rho(D)$. However, Theorems 1-3 no longer identify all indistinguishable configurations, because a circuit with an unknown user can belong to any user, not just a user with an uncompromised first router. It can, however, be shown that the converse of Theorem 4 continues to hold if we replace "$C \approx_{D_A} D$" with "$\rho(C) = \rho(D)$."

5   Conclusions

We have presented a model of onion routing and characterized when anonymity and unlinkability are provided. It is an asynchronous model using IO automata, and the protocol is based on Tor. The adversary we analyze is local and active in

the sense that he is allowed to run arbitrary automata but is limited to the view of a subset of users and routers that he controls. We show that the adversary can determine when his routers hold positions in the same circuit and where in the circuit they are located, and only this. Thus anonymity is generally provided when the first or last circuit router is uncompromised.

Two directions for future work on modeling onion routing are improving the model and improving the analysis. A big missing piece in the current model is the lack of time. Timing attacks are successful in practice, and we have only approximated them in our model with circuit identifiers. Also, we have simplified the Tor protocol by omitting key exchange, circuit teardowns, the final unencrypted message forward, and stream management and congestion control. Adding some or all of these would bring the model closer to reality. Towards improving the analysis, we have made several assumptions about the cryptosystem but have not exhibited an encryption scheme for which they hold. Also, probabilities in both the behavior of the users and the operation of system should be added to the model and analyzed according to probabilistic definitions of anonymity.

References

1. Anonymity bibliography, http://freehaven.net/anonbib/
2. Boucher, P., Shostack, A., Goldberg, I.: Freedom systems 2.0 architecture. White paper, Zero Knowledge Systems, Inc. (2000)
3. Camenisch, J., Lysyanskaya, A.: A formal treatment of onion routing. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 169–187. Springer, Heidelberg (2005)
4. Chaum, D.: Security without identification: Transaction systems to make big brother obsolete. CACM 28(10) (October 1985)
5. Díaz, C., Seys, S., Claessens, J., Preneel, B.: Towards measuring anonymity. In: Privacy Enhancing Technologies (2002)
6. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: 13th USENIX Security Symposium (2004)
7. Goldberg, I.: On the security of the Tor authentication protocol. In: Danezis, G., Golle, P. (eds.) PET 2006. LNCS, vol. 4258, pp. 316–331. Springer, Heidelberg (2006)
8. Goldschlag, D., Reed, M., Syverson, P.: Hiding routing information. In: Anderson, R. (ed.) Information Hiding. LNCS, vol. 1174, pp. 137–150. Springer, Heidelberg (1996)
9. Lynch, N.: Distributed Algorithms, 1st edn. Morgan Kaufmann, San Francisco (1996)
10. Mauw, S., Verschuren, J., de Vink, E.: A formalization of anonymity and onion routing. In: European Symposium on Research in Computer Security (2004)
11. Øverlier, L., Syverson, P.: Locating hidden servers. In: IEEE Symposium on Security and Privacy (2006)
12. Serjantov, A., Danezis, G.: Towards an information theoretic metric for anonymity. In: Dingledine, R., Syverson, P.F. (eds.) PET 2002. LNCS, vol. 2482, pp. 41–53. Springer, Heidelberg (2003)
13. Syverson, P., Tsudik, G., Reed, M., Landwehr, C.: Towards an analysis of onion routing security. In: Federrath, H. (ed.) Designing Privacy Enhancing Technologies. LNCS, vol. 2009, pp. 96–114. Springer, Heidelberg (2001)

K-Anonymous Multi-party Secret Handshakes

Shouhuai Xu[1] and Moti Yung[2]

[1] Department of Computer Science, University of Texas at San Antonio
shxu@cs.utsa.edu
[2] RSA Labs, EMC Corp. and Department of Computer Science, Columbia University
moti@cs.columbia.edu

Abstract. Anonymity-protection techniques are crucial for various commercial and financial transactions, where participants are worried about their privacy. On the other hand, authentication methods are also crucial for such interactions. Secret handshake is a relatively recent mechanism that facilitates privacy-preserving mutual authentication between communicating peers. In recent years, researchers have proposed a set of secret handshake schemes based on different assumptions about the credentials used: from one-time credentials to the more general PKI-like credentials. In this paper, we concentrate on k-anonymous secret handshake schemes based on PKI-like infrastructures. More specifically, we deal with the k-anonymous m-party ($m > 2$) secret handshake problem, which is significantly more involved than its two-party counterpart due to the following: When an honest user hand-shakes with $m - 1$ parties, it must be assured that these parties are distinct; otherwise, under the mask of anonymity a dishonest participant may clone itself in a single handshake session (i.e., assuming multiple personalities).

Keywords: privacy-preserving authentication, anonymity, unlinkability, secret handshake, credential systems, reusable credentials, multiple personality prevention.

1   Introduction

"Privacy-preserving authentications" are a class of mechanisms that can achieve the traditional authentication goals, and at the same time protect the anonymity of an authenticator, a verifier, or both. Such tools can be useful in transactions where users require anonymity, yet at the same time, the authenticity of a party is needed (e.g., to assure payments, commitment to a contract, or membership in a proper authorized group). It is not a surprise then that in the last two decades a large and diverse set of settings and methods for privacy-preserving authentications have been investigated, among them: (1) anonymous e-coins [10] whereby a merchant can verify a buyer's capability in paying a certain amount of money; (2) group signatures [11] whereby anyone can verify a user's membership in an existing group where the user's privacy within the group is maintained; (3) ring signatures/authentication [12,24,22] whereby an authenticator can hide itself among an ad hoc group of users; (4) authenticated key exchange while protecting the anonymity of the participants [18,21,8,1].

Secret handshakes are a new type of privacy-preserving authentications. They were introduced by Balfanz et al. [2] to fulfill the following functionality: *two* participating users authenticate each other in a way that no one reveals its own membership (or credential) unless the peer's legitimacy was already ensured of. As a prototypical setting pointed out in the past, the users can be, for example, CIA agents, and thus the all-or-nothing nature of the authentication result disclosure makes sense. They can also be members of a given commercial exchange or an industry sector, conducting preliminary negotiations regarding a deal, and must be authenticated to belong to the right group, or else (if the quorum is not completely from the group) no details should be learned by anyone (not even that there is indeed a quorum or a sub-quorum ready for the proposed deal in the legitimate group). There have been three different approaches to constructing secret handshake schemes.

- Secret handshakes based on one-time credentials. The pioneering two-party secret handshake scheme due to Balfanze et al. [2] and recent results due to Castelluccia et al. [9] and Jarecki et al. [19] adopted this approach. In this approach, the users are forced to use one-time credentials; otherwise, they suffer from the privacy degradation that all the sessions involving a same user (or credential) are trivially *linkable*. (The importance of ensuring unlinkability in anonymous transaction systems has been well recognized, e.g. [10,11].) The adoption of one-time credentials could severely limit the usefulness of the resulting secret handshake schemes, since an attacker can easily deplete an honest user's credentials.
- Secret handshakes based on special credential schemes such as group signatures and group key management. This approach was taken by Tsudik and Xu [26] to construct a flexible framework that can seamlessly facilitate two-party and multi-party secret handshakes. This framework does not suffer from issues imposed by one-time credentials. They also presented some concrete constructions.
- Secret handshakes based on PKI-like key infrastructures. This approach was adopted by the *k*-anonymous *two-party* secret handshake scheme due to Xu and Yung [27]. Intuitively, *k*-anonymity means that a participant can only hide itself among a set of *k* users, rather than among a certain entire population. Thus, the fulfilled *k*-anonymity is *weaker* than the anonymity offered by the aforementioned schemes. However, this approach is valuable because it makes a *weaker* assumption about the underlying credential schemes, namely that it can be built on top of PKI-like infrastructures that may have been more widely deployed. This approach is interesting also because the underlying PKI-like infrastructures themselves do not have to offer any anonymity protection (i.e., this approach can protect anonymity based on non-anonymous infrastructures). We note that in real life, it is highly desirable to exploit a credential mechanism in more than one way (namely, with or without anonymity), which was indeed the motivation for ring signatures (compared to group signatures).

1.1   Our Contributions

We present a multi-party secret handshake method that builds upon the k-anonymous *two-party* secret handshake scheme [27]. Our resulting protocol does not impose any new restrictions on, or introduce any new assumptions into, the setting of [27]. This implies that the resulting multi-party secret handshake schemes achieve *unlinkability* with *reusable* credentials (i.e., unlinkability is not achieved by means of one-time credentials), and that the resulting anonymity still falls into the k-anonymity framework [25], where k is an adjustable parameter indicating the desired anonymity assurance.

We further notice that a straightforward extension to the two-party secret handshake protocol of [27] is subject to a newly emerged attack (i.e., this attack does not have a counterpart in the case of two-party secret handshakes). This attack has to do with a dishonest participant cloning itself in a single handshake session (i.e., assuming multiple personalities), without others realizing that the parties behind supposedly different parties are merely dittos of the same actual misbehaving party (we further discuss these attacks in Section 4.1). Consequently, the resulting k-anonymous m-party ($m > 2$) secret handshake protocol is significantly more involved. Specifically, we present a detailed k-anonymous m-party secret handshake scheme based on a standard PKI, and sketch another scheme based on a symmetric key pre-distribution scheme. Both schemes are efficient and provably secure (in the random oracle model).

Other Related Works: The anonymity fulfilled in this paper falls into the k-anonymity framework due to Sweeney [25]. This framework was originally motivated to protect privacy in the context of database systems so that released information limits what can be revealed about properties of the entities that are to be protected. This anonymity assurance has recently been utilized in some other applications. See also [27] for other, more loosely related, prior works.

Organization: In Section 2, we briefly review the utilized cryptographic tools. In Section 3 we present a system model and definition of k-anonymous m-party secret handshake schemes. In Section 4 we present a scheme based on public key cryptosystems, whereas in Section 5 we sketch a scheme based on symmetric key cryptosystems. We conclude the paper in Section 6.

2   Cryptographic Tools

A function $\epsilon : \mathbb{N} \to \mathbb{R}^+$ is negligible if for any c there exists κ_c such that $\forall \kappa > \kappa_c$ we have $\epsilon(\kappa) < 1/\kappa^c$.

Public Key Cryptosystems. A public key cryptosystem consists of three polynomial-time algorithms: pKeyGen, pEnc, and pDec. The probabilistic key generation algorithm pKeyGen takes as input 1^κ and outputs a key pair (pk, sk). The probabilistic encryption algorithm pEnc takes as input a public key pk and a message m, and outputs a ciphertext c. The decryption algorithm pDec takes as input a ciphertext c and a private key sk, and returns a message m or $\perp$ (meaning that c is not valid). We will use public key cryptosystems that are secure against

adaptive chosen ciphertext attack (i.e., IND-CCA2) [23,14]. Practical schemes are available in [5,15,13].

Pseudorandom Functions. A pseudorandom function (PRF) family $\{f_k\}$ parameterized by a secret value k of length κ has the following property [16]: An adversary $\mathcal{A}$ cannot distinguish f_k, where $k \in_R \{0,1\}^\kappa$, from a perfect random function (with the same domain and range) with a non-negligible (in κ) probability.

Symmetric Key Cryptosystems. A symmetric key cryptosystem consists of three polynomial-time algorithms: sKeyGen, sEnc, and sDec. The probabilistic key generation algorithm sKeyGen takes as input 1^κ and outputs a key pair k. The encryption algorithm sEnc takes as input a key k and a message m, and outputs a ciphertext c. The decryption algorithm sDec takes as input a ciphertext c and a key sk, and returns a message m or $\perp$ (meaning that c is not valid). In parallel to the above public key cryptosystems, we will use symmetric key cryptosystems that are secure against adaptive chosen ciphertext attack or IND-CCA2 secure (we refer the reader to [20] for a thorough treatment on this subject).

Digital Signature Schemes. A digital signature scheme consists of three polynomial-time algorithms: pKeyGen, Sign, and Ver. The probabilistic key generation algorithm pKeyGen takes as input 1^κ and outputs a key pair (pk, sk). The signing algorithm Sign takes as input a message m and a private key sk, and outputs a signature σ. The verification algorithm Ver takes as input a message m, a public key pk, and a candidate signature σ, returns "accept" if σ is a valid signature and "reject" otherwise. A signature schemes should be existentially unforgeable under an adaptive chosen-message attack [17]. Basically, this means that an adversary $\mathcal{A}$ cannot output a valid signature on a message that was not signed by the signing oracle with a non-negligible probability in κ.

3   Model and Definition

Let U be a set of all possible users. Suppose there is a set of l groups $\mathbf{G} = \{G_1, \cdots, G_l\}$, where each group $G \in \mathbf{G}$ is a set of users (i.e., $G \subset \mathsf{U}$) and managed by an authority CA (the term "group" in this paper always refers to a set of users unless explicitly stated otherwise). In other words, a CA is responsible for admitting users into a group and revoking their memberships when the need arises – just like a certificate authority in a standard PKI. All the participants are modeled as probabilistic polynomial-time algorithms. For simplicity, we assume that each user is a member of exactly one group.

We assume that the communication channels are under an adversary's control. However, the channels are (semi-)synchronous so as to facilitate guaranteed delivery. We assume that there is a broadcast channel between the handshaking participants. The polynomial-time adversary is allowed to corrupt some of the participants at the start of the protocol.

Definition: A multi-party secret handshake scheme SHS consists of the following algorithms:

SHS.CreateGroup This algorithm is executed by an authority, CA, to establish a group G. It takes as input appropriate security parameters, and outputs a cryptographic context specific to this group. In particular, the context may include a data structure called certificate/membership revocation list, CRL, which is originally empty. The cryptographic context is made public.

SHS.AdmitMember This algorithm is run by a CA to admit a user to become a member of the group that is under the CA's jurisdiction. The CA admits members according to a certain policy, which is orthogonal to the focus of this paper. For example, the CA may interact with the user to verify its real identity and its ownership of the private key corresponding to a claimed public key. After executing this algorithm, the group state information (e.g., the list of the members' certificates) is appropriately updated, and the member holds some secret(s) as well as a membership certificate. We will identify an anonymous user through its pseudonym $U \in \mathsf{U}$, which can be included in its certificate.

SHS.Handshake$(U_1, \ldots, U_m)$ This protocol is executed by a set of $m > 2$ distinct anonymous users, where $\{U_1, \ldots, U_m\} \subset \mathsf{U}$ are just a set of placeholders, and U_1 plays the role of the initiator. (It is even true that U_1 does not know any other U_i's pseudonym before a successful handshake, and vice versa.) The input to this protocol includes the anonymous users' secrets, and possibly some public information regarding the current state of the system. The output of this protocol, upon completion, ensures that, for all $i, j \in \{1, \ldots, m\}$ and $i \neq j$, U_i discovers $U_j \in G$ if and only if U_j discovers $U_i \in G$. We say that the protocol returns "1" if the handshake succeeds (i.e., $U_1, \ldots, U_m$ all belong to G), and "0" otherwise.

SHS.RemoveUser This algorithm is executed by an authority CA. It takes as input its current CRL and U's certificate/pseudonym. The output includes an updated CRL which includes the newly revoked certificate U, and perhaps the updated list of the members' certificates/pseudonyms.

Security Definition: Consider a probabilistic polynomial-time adversary $\mathcal{A}$ that may have access to the following oracles:

- $\mathcal{O}_{CG}(\cdot)$: This activates a new CA to create a new group via operation SHS.CreateGroup. The identity, CA, may be given by $\mathcal{A}$ as the input. We assume that a CA is not under $\mathcal{A}$'s control before the new group is established. However, the CA may be corrupt immediately after its establishment (i.e., before admitting any user into the group).

- $\mathcal{O}_{AM}(\cdot, \cdot)$: The input includes the identity of a CA and, optionally, the identity U of a user that is under $\mathcal{A}$'s control. By $\mathcal{O}_{AM}(CA, U)$, the CA may admit the *corrupt* user U by executing SHS.AdmitMember; by $\mathcal{O}_{AM}(CA)$, the CA executes SHS.AdmitMember to admit an *honest* user and assigning it with a unique pseudonym U.

- $\mathcal{O}_{HS}(\cdot, \ldots, \cdot)$: The oracle will activate SHS.Handshake between U_1, $\ldots$, and U_m that are given by $\mathcal{A}$. A corrupt user will execute whatever is determined by the adversary.

– $\mathcal{O}_{RU}(\cdot,\cdot)$: The input includes the identity of a CA and a pseudonym U. The oracle actives SHS.RemoveUser to insert U into the corresponding CRL, and the system state information is appropriately updated.
– $\mathcal{O}_{Corrupt}(\cdot,\cdot)$: The input includes the identity of a CA, and possibly a pseudonym U issued by CA. By $\mathcal{O}_{Corrupt}(CA,U)$, the oracle returns U's current internal state information (including all secrets) to $\mathcal{A}$; by $\mathcal{O}_{Corrupt}(CA)$, the oracle returns CA's current internal state information (including all secrets) to $\mathcal{A}$. Once the CA or U is corrupt, it will execute what $\mathcal{A}$ is pleased of, until such a corruption is detected by some outside mechanism (e.g., intrusion detection systems). When the corruption of a user U is detected, it is excluded from the group via SHS.RemoveUser; when the corruption of an authority CA is detected, its group is simply excluded from the system.

Consider the following security properties: `correctness`, `resistance to impersonation attacks`, and k-`anonymity` specified through k-`resistance to detection attacks`, k-`unlinkability`, and k-`indistinguishability to eavesdroppers`. Compared with the two-party case, major changes are made to the `resistance to impersonation attacks` property.

`Correctness.` If m distinct honest users $U_1, \ldots, U_m$ belong to the same group, then SHS.Handshake$(U_1, \ldots, U_m)$ always returns "1;" otherwise, it returns "0."

`Resistance to impersonation attacks.` The adversary may corrupt as many members of the group G (needed and employed in the session), as long as there is at least one honest party that is not participating in a handshake session SHS.Handshake$(X_1, \ldots, X_m)$ (namely, by the pigeon hole principle, the adversary has to play for a party not present, perhaps by using proper group credentials that are already used within the same session). Formally, let $\Xi(X_i)$ denote the (honest or corrupt) user corresponding to the role of X_i in a handshake session, where the adversary intends to impersonate an honest $\Xi(X_i)$. Consider the experiment specified below.

Experiment RIA$_{SHS,\mathcal{A}}(1^\kappa)$:
    (stateInfo, CA, U, b, m) $\leftarrow \mathcal{A}^{\mathcal{O}_{CG}(\cdot),\mathcal{O}_{AM}(\cdot,\cdot),\mathcal{O}_{AM}(\cdot),\mathcal{O}_{HS}(\cdot,\ldots,\cdot),\mathcal{O}_{RU}(\cdot,\cdot),\mathcal{O}_{Corrupt}(\cdot,\cdot)}(\cdot)$
        where $1 \leq b \leq m$ and U is a member of the group managed by CA
    Return "1" if the following hold and "0" otherwise:
        $1 \leftarrow$ SHS.Handshake$(X_1, \ldots, X_m)$ where the role of X_b is played by U
        (1) There is no $\mathcal{O}_{Corrupt}(CA)$ query
        (2) There is no $\mathcal{O}_{AM}(CA,U)$ query
        (3) There is no $\mathcal{O}_{RU}(CA,U)$ query
        (4) $|\{\Xi(X_1), \ldots, \Xi(X_{b-1}), \Xi(X_{b+1}), \ldots, \Xi(X_m)\}$
            $\cap \{U' : \exists \mathcal{O}_{AM}(\cdot,U') \vee \exists \mathcal{O}_{Corrupt}(U')\}| \leq m-2$
        (5) $\exists X \in \{X_1, \ldots, X_{b-1}, X_{b+1}, \ldots, X_m\}$
            such that $\neg((\exists \mathcal{O}_{AM}(\cdot,\Xi(X)) \vee \exists \mathcal{O}_{Corrupt}(\Xi(X))) \wedge \neg \exists \mathcal{O}_{RU}(CA, \Xi(X)))$
            and $\Xi(x)$ does not participate in the session

The above (4) captures that there are at least two roles *corresponding to* two honest users (including U), whereas (5) captures that at least one role (corresponding to an honest user) *is being played* by the adversary. Let AdvRIA$_{SHS,\mathcal{A}}(1^\kappa) =$ Pr[RIA$_{SHS,\mathcal{A}}(1^\kappa)$ returns "1"], which is the probability that RIA$_{SHS,\mathcal{A}}(1^\kappa)$ returns "1", where probability is taken over all the tossed coins. A handshake scheme SHS is "`resistant to impersonation attacks`" if for $\forall \mathcal{A}$, AdvRIA$_{SHS,\mathcal{A}}(1^\kappa)$ is

negligible in the security parameter of SHS. We notice that this definition can be smoothly degenerated to the two-party case of $m = 2$ to accommodate its counterpart definition in [27]. This is because the above restrictions (4) and (5) imply that the adversary intends to impersonate an honest user.

k-resistance to detection attacks. Suppose there are β groups such that none of their members or CAs is corrupt. Ideally, "resistance to detection attacks" means that no adversary $\mathcal{A}$ who does not belong to any of the β groups, can successfully guess the membership of an anonymous (and honest) handshaking peer U, who is a member of one of the β groups, with a non-negligible advantage over $1/\beta$. In this paper, we pursue a weaker, but practical and useful (see [27] for discussions on the usefulness of k-anonymity in the context of secret handshakes), notion we call "k-resistance to detection attacks", where $2 \le k \le \beta$ is a parameter indicating the desired anonymity assurance. Intuitively, it means that no adversary $\mathcal{A}$, who does not belong to any of the k groups (including the group U belongs to), can successfully guess the membership of U with a non-negligible advantage over $1/k$. Formally, consider the experiment specified below.

Experiment $\mathsf{RDA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$:
    $(\mathsf{stateInfo}, b, m) \leftarrow \mathcal{A}^{\mathcal{O}_{CG}(\cdot),\mathcal{O}_{AM}(\cdot,\cdot),\mathcal{O}_{AM}(\cdot),\mathcal{O}_{HS}(\cdot,\dots,\cdot),\mathcal{O}_{RU}(\cdot,\cdot),\mathcal{O}_{Corrupt}(\cdot,\cdot)}(\cdot)$
        where $1 \le b \le m$
    Execute $\mathsf{SHS}.\mathsf{Handshake}(X_1, \dots, X_{b-1}, U_b, X_{b+1}, \dots, X_m)$
        where $X_1, \dots, X_{b-1}, X_{b+1}, \dots, X_m$ are placeholders for the other participants and
        $\mathcal{A}$ plays the role of at least one of them
    Let CA_i be the authority of group G_i, to which U_b belongs
    Return "1" if the following hold and "0" otherwise:
        (1) There is no $\mathcal{O}_{RU}(CA_i, U_b)$ query
        (2) There are at least k groups (including G_i) s.t. for each one managed by CA:
            (2.1) There is no $\mathcal{O}_{Corrupt}(CA)$ query
            (2.2) If there is an $\mathcal{O}_{AM}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query
            (2.3) If there is an $\mathcal{O}_{Corrupt}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query
        (3) $\mathcal{A}$ outputs i

Let $\mathsf{AdvRDA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa) = |\Pr[\mathsf{RDA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa) \text{ returns "1"}] - 1/k|$, which is the advantage that $\mathcal{A}$ successfully guesses the membership of an anonymous handshake peer X. The probability is taken over all the tossed coins. A scheme SHS is "k-resistant to detection attacks" if for $\forall \mathcal{A}$, $\mathsf{AdvRDA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ is negligible in the security parameter of SHS.

k-unlinkability. Suppose there are β groups such that none of their members or CAs is corrupt. Ideally, "unlinkability" means that no adversary $\mathcal{A}$ who does not belong to any of the β groups, can successfully associate two sessions involving a same honest user with a non-negligible advantage over $1/\beta$. (We remark that this corresponds to the worst case scenario that the honest user could have been identified by the adversary in one of the two sessions through whatever means; e.g., the adversary corrupted some users of the group the honest user belongs to when that specific session was conducted.) In this paper, we pursue a weaker, but still practical and useful, notion we call "k-unlinkability", where $2 \le k \le \beta$ is a parameter indicating the desired anonymity assurance. Intuitively, it means that no adversary $\mathcal{A}$ who does not belong to any of the k groups, can

successfully associate two sessions, s_1 and s_2, involving a same honest user with a non-negligible advantage over $1/k$. Formally, consider the experiment specified below.

Experiment $\mathsf{Unlink}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$:

$(s_1, s_2) \leftarrow \mathcal{A}^{\mathcal{O}_{CG}(\cdot,\cdot), \mathcal{O}_{AM}(\cdot,\cdot), \mathcal{O}_{AM}(\cdot), \mathcal{O}_{HS}(\cdot,\ldots,\cdot), \mathcal{O}_{RU}(\cdot,\cdot), \mathcal{O}_{Corrupt}(\cdot,\cdot)}(\cdot)$

Return "1" if the following hold and "0" otherwise:

(1) s_1 and s_2 involve a same user $U \in G_i$, where G_i is managed by CA_i

(2) There is no $\mathcal{O}_{RU}(CA_i, U)$ query

(3) There exist $z \in \{1, 2\}$ such that w.r.t. s_z the following holds:

There are at least k groups (including G_i) s.t. for each one managed by CA:

(3.1) There is no $\mathcal{O}_{Corrupt}(CA)$ query

(3.2) If there is an $\mathcal{O}_{AM}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query

(3.3) If there is an $\mathcal{O}_{Corrupt}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query

---

Let $\mathsf{AdvUnlink}_{\mathsf{SHS},\mathcal{A}}(1^\kappa) = |\Pr[\mathsf{Unlink}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ returns "1"$] - 1/k|$, which is the advantage that $\mathcal{A}$ successfully associates two handshake sessions to a same honest user. The probability is taken over all the tossed coins. A scheme SHS is "k-unlinkable" if for $\forall \mathcal{A}$, $\mathsf{AdvUnlink}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ is negligible in the security parameter of SHS.

k-indistinguishability to eavesdroppers. Consider an adversary $\mathcal{A}$ who corrupts some users, interacts with users, and observes a session of SHS.Handshake for incorrupt users $U_1, \ldots, U_m$. Suppose there are at least k groups (including the groups $U_1, \ldots, U_m$ belong to) such that none of their members or CAs is corrupt. Intuitively, this property means that $\mathcal{A}$ should not be able to learn from this handshake session anything that it did not already know. In order to capture this property, consider a simulated transcript of a handshake session, which is obtained by substituting all the strings derived from cryptographic secrets with random strings of appropriate lengths (i.e., no cryptographic secrets or memberships are involved). Yet, such a substitution cannot be detected by $\mathcal{A}$. Formally, consider the experiment specified below.

Experiment $\mathsf{INDeav}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$:

Let $R \notin \mathsf{U}$ be an algorithm/simulator having no access to any cryptographic secrets

stateInfo $\leftarrow \mathcal{A}^{\mathcal{O}_{CG}(\cdot,\cdot), \mathcal{O}_{AM}(\cdot,\cdot), \mathcal{O}_{AM}(\cdot), \mathcal{O}_{HS}(\cdot,\ldots,\cdot), \mathcal{O}_{RU}(\cdot,\cdot), \mathcal{O}_{Corrupt}(\cdot,\cdot)}(\cdot)$

Flip a random coin b

If $b = 0$ then give $\mathcal{A}$ a transcript $trans$ of SHS.Handshake$(U_1, \ldots, U_m)$

else give $\mathcal{A}$ a simulated transcript $trans$ of SHS.Handshake$(R, \ldots, R)$

$b' \leftarrow \mathcal{A}^{\mathcal{O}_{CG}(\cdot,\cdot), \mathcal{O}_{AM}(\cdot,\cdot), \mathcal{O}_{AM}(\cdot), \mathcal{O}_{HS}(\cdot,\ldots,\cdot), \mathcal{O}_{RU}(\cdot,\cdot), \mathcal{O}_{Corrupt}(\cdot,\cdot)}(\mathsf{stateinfo}, trans)$

Suppose $U_i \in G_i$ and G_i is managed by CA_i for $1 \le i \le m$

Return "1" if the following hold and "0" otherwise:

(1) $b' = b$

(2) There are no $\mathcal{O}_{RU}(CA_i, U_i)$ queries for all $i \in \{1, \ldots, m\}$

(3) There are at least k groups (including those the $U_1, \ldots, U_m$ belong to) such that for each group G managed by CA:

(3.1) There is no $\mathcal{O}_{Corrupt}(CA)$ query

(3.2) If there is an $\mathcal{O}_{AM}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query

(3.3) If there is an $\mathcal{O}_{Corrupt}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query

---

Let us define $\mathsf{AdvINDeav}_{\mathsf{SHS},\mathcal{A}}(1^\kappa) = |\Pr[\mathsf{INDeav}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ returns "1"$|b = 0] - \Pr[\mathsf{INDeav}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ returns "1"$|b = 1]|$, which is the advantage that $\mathcal{A}$ successfully distinguishes a real transcript from a simulated one. The probability is taken over all the tossed coins. A scheme SHS is "k-indistinguishable to eavesdroppers" if for $\forall \mathcal{A}$, $\mathsf{AdvINDeav}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ is negligible in the security parameter of SHS.

4   Public Key Cryptosystems Based Construction

Notations. Recall that we denoted by $\mathbf{G} = \{G_1, \cdots, G_l\}$ a set of groups, where G_z $(1 \leq z \leq l)$ is a set of users whose public keys are certified by an authority CA_z using a secure signature scheme. Without loss of generality, we assume that both the groups $G_1, \cdots, G_l$ and the users in a group $G \in \mathbf{G}$ are in an appropriate order (e.g., alphabetic), which means that partitions over the sets can be naturally defined. If X is a user, let $\mathsf{Group}(X)$ be a function that returns the identity of the group to which X belongs, and $Cert_X$ denote X's public key certificate. If $Cert$ is a certificate, let $\mathsf{CA}(Cert)$ be a function that returns the identity of the authority which issues it. For example, a user $X \in G_i$ owning a public key pk_X will be issued a certificate $Cert_X$ by authority CA_i. Moreover, $\mathsf{Group}(X)$ returns G_i and $\mathsf{CA}(Cert_X)$ returns CA_i.

Let κ_0 and κ_1 be additional security parameters and q be a prime of length κ_0. Assume $f, g : \{0,1\}^{\kappa_1} \times \{0,1\}^* \rightarrow \{0,1\}^{\kappa_1}$ are pseudorandom functions, and $h : \{0,1\}^* \rightarrow \mathbb{Z}_q$ is an ideal hash function [4].

4.1   Basic Ideas

Basic idea underlying the k-anonymous two-party secret handshake scheme. The basic idea underlying the two-party scheme [27] is to let each party "draft" on-the-fly a certain number w of users to form a "crowd" while ensuring that the drafting algorithms leak no information about the drafting party. Specifically, the first participant drafts a crowd $U_{1,1}, \ldots, U_{1,w}$ such that it plays the role of $U_{1,i}$ for some $1 \leq i \leq w$, and the second participant drafts a crowd $U_{2,1}, \ldots, U_{2,w}$ such that it plays the role of $U_{2,j}$ for some $1 \leq j \leq w$. The drafting algorithm ensures that, for $1 \leq z \leq 2$, $U_{1,z}$ and $U_{2,z}$ belong to the the same group (i.e., they are managed by the same authority). If we let the first participant encrypt a random value v_1 using the public key of $U_{2,i}$, and the second participant encrypts a random value v_2 using the public key of $U_{1,j}$, then a successful handshake (i.e., $i = j$) allows the participants to reach a common secret $F(v_1, v_2)$ for some appropriate function F.

Why a straightforward extension to the two-party secret handshake scheme is not secure? As briefly reviewed above, a successful two-party secret handshake allows the participants to reach a common secret $F(v_1, v_2)$ by ensuring, for instance, that v_1 is encrypted **only** using the other involved public key certified by the same CA (i.e., the CA that certified the public key of the participant who selected v_1). In a straightforward extension to the two-party case, a successful multi-party secret handshake would allow the participants to reach a common secret $F(v_1, v_2, \ldots, v_m)$, where v_i is a random value selected by the ith participant. To facilitate this, for instance, v_1 should be encrypted using **all** the other involved public keys certified by the same CA (i.e., the CA that certified the public key of the participant who selected v_1). However, this would allow a dishonest user to cheat an honest user into believing that it is handshaking with $m - 1 > 1$ peers, while it is indeed handshaking with a single

dishonest user (who can clone its participation and thus exhibit multiple personalities). This is because using the adversarial participant's private key alone is enough to derive $F(v_1, v_2, \ldots, v_m)$. The attack is specific to, and emerged in, the multi-party scenario, because in the case of two-party secret handshakes, $m = 2$ and thus the above two cases (i.e., "the **only** other key" vs. "**all** the other keys") amount to the same thing. (We remark that the capability that "one can impersonate multiple parties belonging to the same group" may not be harmful in some specific application scenarios, but should be eliminated in general, and is an attack where a quorum is required.)

How should the newly emerged attack be dealt with? We adopt a two-stage strategy of constructing k-anonymous multi-party secret handshakes. In the first stage (corresponding to Phase I-III in the construction), a preliminary secret handshake is conducted to detect if the participants indeed belong to the same group, but there is no guarantee that the participants are distinct. This is quite similar to the aforementioned straightforward extension to the two-party case.

In the second stage (corresponding to Phase IV-VI in the construction), effort is taken to ensure that the participants in a successful preliminary handshake are indeed distinct. This is fulfilled by forcing that the private keys corresponding to all the other involved public keys belonging to the same group are utilized for decryption. For simplicity, let's consider the case of three-party secret handshakes, where each participant draws a "crowd" of size w. We can let the first participant encrypt $v_{1,2,1}, \ldots, v_{1,2,w}$ to the second participant using the respective public keys drawn by the second participant, and encrypt $v_{1,3,1}, \ldots, v_{1,3,w}$ to the third participant using the respective public keys drawn by the third participant. Then we can let the second participant "translate" (i.e., decrypt and then encrypt) $v_{1,2,1}, \ldots, v_{1,2,w}$ to the third participant, and let the third participant "translate" (i.e., decrypt and then encrypt) $v_{1,3,1}, \ldots, v_{1,3,w}$ to the second participant. If we let each participant act as the first participant (this is necessary; otherwise, attacks on anonymity are easy to conceive), then it is clear that a successful handshake does result in a common secret $F(v_{1,2,i}, v_{1,3,i}, v_{2,1,i}, v_{2,3,i}, v_{3,1,i}, v_{3,2,i})$, where i indicates the location of the group, to which the participants belong (i.e., in the case of a successful handshake), in the crowds. We notice that special care must also be taken to ensure, for instance, that by simply observing the traffic flow it is not possible for the adversary to tell a successful (preliminary) secret handshake from a failed one. This combination of properties explains why the resulting protocol is seemingly quite involved.

4.2 Subroutines

Two subroutines are specified below. Specifically, rSelect is for selecting w groups from $\mathbf{G} = \{G_1, \cdots, G_l\}$ where w is a parameter that will be determined later, mSelect is for selecting w members from the w groups, rSelectVer and mSelectVer are for verifying that the selections are appropriately done. For simplicity, we assume $w|l$. Proof of Lemma 1 can be found in the full version of the present paper [28]. This lemma ensures that the adversary cannot tell who is the party

that draws a "crowd" in the above rSelect and mSelect algorithms. Therefore, we will simply treat the two processes as being ideal.

The algorithm $\mathsf{rSelect}(\mathbf{G}, U_1, w, n_1, \ldots, n_m)$ has the following steps:

1. Partition $\mathbf{G}$ into $\mathbf{G}_0, \cdots, \mathbf{G}_{w-1}$ where $\mathbf{G}_z = \{G_{z_0}, \cdots, G_{z_{l/w-1}}\}$ for $0 \le z \le w-1$. Assume $U_1 \in G_{i_u}$ for some $0 \le i \le w-1$ and $0 \le u \le l/w-1$.
2. Set $\eta = h(n_1, \ldots, n_m, 0)$ and $x = h(n_1, \ldots, n_m, 1, i)$. Choose $r \in_R \{0, 1, \cdots, \lfloor (q-1)^2 w/l \rfloor\}$ and set $y_i = u + r \cdot l/w$ (in $\mathbb{Z}$). Solve $y_i = \eta \cdot x + \theta_1 \mod q$ to get θ_1.
3. For $z = 0$ to $w-1$ (except $z = i$), set $y_z = \eta \cdot h(n_1, \ldots, n_m, 1, z) + \theta_1 \mod q$, and $s_z = y_z \mod l/w$.
4. Output $(\mathbf{G}^* = \{G_{z_{s_z}}\}_{z=0}^{w-1}, \theta_1)$, where $G_{z_{s_z}}$ is the group selected from $\mathbf{G}_z$ and $s_i = u$.

The algorithm $\mathsf{rSelectVer}(\mathbf{G}, \mathbf{G}^*, w, n_1, \ldots, n_m, \theta_1)$ has the following steps:

1. Partition $\mathbf{G}$ into $\mathbf{G}_0, \cdots, \mathbf{G}_{w-1}$ where $\mathbf{G}_z = \{G_{z_0}, \cdots, G_{z_{l/w-1}}\}$ for $0 \le z \le w-1$.
2. Parse $\mathbf{G}^*$ as $\{G_{z_{s_z}}\}_{z=0}^{w-1}$, and set $\eta = h(n_1, \ldots, n_m, 0)$.
3. Accept if for $z = 0$ to $w-1$, it holds that $s_z = y_z \mod l/w$ where $y_z = \eta \cdot h(n_1, \ldots, n_m, 1, z) + \theta_1 \mod q$; reject otherwise.

The algorithm $\mathsf{mSelect}(\mathbf{G}^*, X, w, n_1, \ldots, n_m)$ has the following steps:

1. Parse $\mathbf{G}^*$ as $\{G_{z_{s_z}}\}_{z=0}^{w-1}$. Assume X is the λ-th member of group $G_{a_{s_a}}$ for some $0 \le a \le w-1$ and $0 \le \lambda \le |G_{a_{s_a}}|-1$.
2. Set $\eta = h(n_1, \ldots, n_m, 2)$ and $x = h(n_1, \ldots, n_m, 3, a, s_a)$. Choose $r \in_R \{0, 1, \cdots, \lfloor (q-1)^2/|G_{a_{s_a}}| \rfloor\}$, and set $y_a = \lambda + r \cdot |G_{a_{s_a}}|$ (in $\mathbb{Z}$). Solve $y_a = \eta \cdot x + \theta_2 \mod q$ to get θ_2.
3. For $z = 0$ to $w-1$ (except $z = a$), set $\lambda_z = y_z \mod |G_{z_{s_z}}|$ where $y_z = \eta \cdot h(n_1, \ldots, n_m, 3, z, s_z) + \theta_2 \mod q$.
4. Output $(\mathbf{X} = \{X_{z_{s_z}, \lambda_z}\}_{z=0}^{w-1}, \theta_2)$, where $\lambda_a = \lambda$ and $X_{z_{s_z}, \lambda_z}$ is the λ_z-th member of group $G_{z_{s_z}}$.

The algorithm $\mathsf{mSelectVer}(\mathbf{G}^*, \mathbf{X}, w, n_1, \ldots, n_m, \theta_2)$ has the following steps:

1. Parse $\mathbf{G}^*$ as $\{G_{z_{s_z}}\}_{z=0}^{w-1}$, and parse $\mathbf{X}$ as $\{X_{z_{s_z}, \lambda_z}\}_{z=0}^{w-1}$.
2. Accept if for $z = 0$ to $w-1$, it holds that $\lambda_z = y_z \mod |G_{z_{s_z}}|$ where $y_z = \eta \cdot h(n_1, \ldots, n_m, 3, z, s_z) + \theta_2 \mod q$; reject otherwise.

Lemma 1. *Let q be a positive number super-polynomial in security parameter κ_0, and p_1 be uniformly distributed over $\mathbb{Z}_q$, and p_2, p_3 be some positive numbers bounded by some polynomials in κ_0. Let further (a, x, b) be uniformly distributed over $(\mathbb{Z}_q)^3$, and r be uniformly distributed over $\{0, \ldots, \lfloor (q-1)^2 p_2/p_3 \rfloor\}$. Then the distribution of $y = ax + b$ in $\mathbb{Z}$ and the distribution of $y' = p_1 + r \cdot p_3/p_2$ in $\mathbb{Z}$ are statistically close to each other. Moreover, the distribution of $y \mod q$ and the distribution of $y' \mod q$ are also statistically close to each other.*

4.3   The Construction

Since all the algorithms other than $\mathsf{SHS.Handshake}(U_1, \ldots, U_m)$ are merely as in a standard and well-known public key infrastructure (i.e., certifying, signing, etc.), we only specify $\mathsf{SHS.Handshake}(U_1, \ldots, U_m)$ which enables $U_1, \ldots, U_m$ to figure out if they belong to the same group.

1. Let U_i, $1 \le i \le m$, select and broadcast $n_i \in_R \{0, 1\}^{\kappa_2}$. This is for the peers to exchange nonces, and typically could have been done before the handshake protocol is activated.

2. Phase-I protocol for U_1:
   (a) Run algorithm $\mathsf{rSelect}(\mathbf{G}, U_1, w, n_1, \ldots, n_m)$ to obtain $(\mathbf{G}^*, \theta_1)$, where $U_1 \in G_{i_{s_i}}$.
   (b) Run algorithm $\mathsf{mSelect}(\mathbf{G}^*, U_1, w, n_1, \ldots, n_m)$ to obtain $(\mathbf{X}_1, \theta_{1,2})$. In order to clarify presentation, parse $\mathbf{X}_1$ also as $\{U_{1,z,z_{s_z},\lambda_{1,z}}\}_{z=0}^{w-1}$, where $U_{1,z,z_{s_z},\lambda_{1,z}} \in G_{z_{s_z}}$ and $0 \leq \lambda_{1,z} \leq |G_{z_{s_z}}| - 1$.
   (c) Broadcast $(\mathbf{G}^*, \mathbf{X}_1, \theta_1, \theta_{1,2}, \Delta_1 = \{Cert_{U_{1,z,z_{s_z},\lambda_{1,z}}}\}_{z=0}^{w-1})$.

3. Phase-I protocol for U_i ($2 \leq i \leq m$): Upon receiving $\{(\mathbf{G}^*, \mathbf{X}_\tau, \theta_1, \theta_{\tau,2}, \Delta_\tau = \{Cert_{U_{\tau,z,z_{s_z},\lambda_{\tau,z}}}\}_{z=0}^{w-1})\}_{\tau=1}^{i-1}$, check for every $1 \leq \tau \leq i-1$ if the selected groups (consistently indicated by $\mathbf{G}^*$, $\mathbf{X}_\tau$, and the certificates) are distinct, if $\mathsf{CA}(Cert_{U_i}) = \mathsf{CA}(Cert_{U_{\tau,j,j_{s_j},\lambda_{\tau,j}}})$ for some $0 \leq j \leq w-1$, if the groups are selected via $\mathsf{rSelectVer}(\mathbf{G}, \mathbf{G}^*, w, n_1, \ldots, n_m, \theta_1)$, if $Cert_{U_i} \notin \Delta_\tau$, and if the members are selected via $\mathsf{mSelectVer}(\mathbf{G}^*, \mathbf{X}_\tau, w, n_1, \ldots, n_m, \theta_{\tau,2})$. If any condition is not satisfied, quit; otherwise, execute as follows:
   (a) Set $\mathbf{G}' = \{G'_{z_{s_z}}\}_{z=0}^{w-1}$, where $G'_{z_{s_z}} = G_{z_{s_z}} - \{U_{1,z,z_{s_z},\lambda_{1,z}}, \ldots, U_{i-1,z,z_{s_z},\lambda_{i-1,z}}\}$.
   (b) Run algorithm $\mathsf{mSelect}(\mathbf{G}', U_i, w, n_1, \ldots, n_m)$, which returns $(\mathbf{X}_i, \theta_{i,2})$. In order to clarify presentation, parse $\mathbf{X}_i$ also as $\{U_{i,z,z_{s_z},\lambda_{i,z}}\}_{z=0}^{w-1}$, where $U_{i,z,z_{s_z},\lambda_{i,z}} \in G'_{z_{s_z}}$ and $0 \leq \lambda_{i,z} \leq |G'_{z_{s_z}}| - 1$.
   (c) Broadcast $(\mathbf{G}^*, \mathbf{X}_i, \theta_1, \theta_{i,2}, \Delta_i = \{Cert_{U_{i,z,z_{s_z},\lambda_{i,z}}}\}_{z=0}^{w-1})$.

4. Phase-II protocol for U_i ($1 \leq i \leq m$): At this point, U_i has received $\{(\mathbf{G}^*, \mathbf{X}_\tau, \theta_1, \theta_{\tau,2}, \Delta_\tau = \{Cert_{U_{\tau,z,z_{s_z},\lambda_{\tau,z}}}\}_{z=0}^{w-1})\}_{1 \leq \tau \leq m}$. For $\tau = i+1$ to m, U_i checks if $\mathsf{CA}(Cert_{U_{\tau,z,z_{s_z},\lambda_{\tau,z}}}) = \mathsf{CA}(Cert_{U_{i,z,z_{s_z},\lambda_{i,z}}})$ for $0 \leq z \leq w-1$, and if the members are selected by running $\mathsf{mSelectVer}(\mathbf{G}^*, \mathbf{X}_\tau, w, n_1, \ldots, n_m, \theta_{\tau,2})$. If not, quit; otherwise, execute as follows:
   (a) To simplify the presentation, rename $U_{\tau,z,z_{s_z},\lambda_{\tau,z}}$ to $U_{\tau,z}$ because once $1 \leq \tau \leq m$ and $0 \leq z \leq w-1$ are determined, $z_{s_z}, \lambda_{\tau,z}$ is uniquely defined. Therefore, the drawn $m \times w$ users can be denoted by a "matrix" $U_{1,0}, \ldots, U_{1,w-1}, \ldots, U_{m,0}, \ldots, U_{m,w-1}$, where $U_{i,0}, \ldots, U_{i,w-1}$ is the "crowd" drawn by U_i. We will call each column, $U_{1,z}, \ldots, U_{m,z}$, a "handshaking m-tuple of users" where $0 \leq z \leq w-1$.
   (b) For $z = 0$ to $w-1$,
      i. Choose $\delta_{i,z} \in_R \{0,1\}^{\kappa_1}$.
      ii. For every $\tau \in \{1, \ldots, m\} \setminus \{i\}$, encrypt $\delta_{i,z}$ using $pk_{U_{\tau,z}}$ (certified via $Cert_{U_{\tau,z}}$) to obtain $\alpha_{i,\tau,z} = \mathsf{pEnc}(pk_{U_{\tau,z}}, \delta_{i,z})$.
   (c) Broadcast $\Gamma_i = \{\alpha_{U_{i,1,z}}, \ldots, \alpha_{U_{i,i-1,z}}, \alpha_{U_{i,i+1,z}}, \ldots, \alpha_{U_{i,m,z}}\}_{z=0}^{w-1}$.

5. Phase-III protocol for U_i ($1 \leq i \leq m$): At this point, U_i (i.e., U_{i,j_i} for some $0 \leq j_i \leq w-1$) has received $(m-1)$ ciphertexts $(\alpha_{U_{1,i,j_i}}, \ldots, \alpha_{U_{i-1,i,j_i}}, \ldots, \alpha_{U_{i+1,i,j_i}}, \ldots, \alpha_{U_{m,i,j_i}})$. Then it decrypts them to obtain the corresponding plaintexts $\delta_{1,j_i}, \ldots, \delta_{i-1,j_i}, \delta_{i+1,j_i}, \ldots, \delta_{m,j_i}$.
   (a) Compute $\pi_{i,j_i} = \oplus_{\tau=1}^{m} \delta_{\tau,j_i}$. (Note that δ_{i,j_i} is selected by and known to U_i.) For every $z \in \{0, \ldots, w-1\} \setminus \{j_i\}$, set $\pi_{i,z}$ to be an element uniformly selected from $\{0,1\}^{\kappa_1}$.

(b) For $z = 0$ to $w - 1$, compute $\sigma_{i,z} = f_{g_{\pi_{i,z}}(1)}(i, z, \Gamma_1, \ldots, \Gamma_m)$. Broadcast $(\sigma_{i,0}, \ldots, \sigma_{i,w-1})$.

(c) For every $\tau \in \{1, \ldots, m\} \setminus \{i\}$: If $\sigma_{\tau,j_i} = f_{g_{\pi_{i,j_i}}(1)}(\tau, j_i, \Gamma_1, \ldots, \Gamma_m)$, this is a successful preliminary handshake; otherwise, it is not.

6. Phase-IV protocol for U_i ($1 \leq i \leq m$):

   Case I: The preliminary handshake is successful. For every $\tau \in \{1, \ldots, m\} \setminus \{i\}$, choose $\delta'_{i,\tau} \in_R \{0,1\}^{\kappa_1}$ and encrypt it using $pk_{U_{\tau,j_i}}$ (certified via $Cert_{U_{\tau,j_i}}$) to obtain $\alpha'_{i,\tau} = \mathsf{sEnc}(g_{\pi_{i,j_i}}(2), \mathsf{pEnc}(pk_{U_{\tau,j_i}}, \delta'_{i,\tau}))$. Finally, broadcast $\Gamma'_i = (\alpha'_{i,1}, \ldots, \alpha'_{i,i-1}, \alpha'_{i,i+1}, \ldots, \alpha'_{i,m})$.

   Case II: The preliminary handshake is not successful. Broadcast $\Gamma'_i = (\alpha'_{i,1}, \ldots, \alpha'_{i,i-1}, \alpha'_{i,i+1}, \ldots, \alpha'_{i,m})$, where $\alpha'_{i,\tau}$ is uniformly selected from the ciphertext space corresponding to sEnc and $\mathsf{pEnc}(pk_{U_{i,\tau}}, \cdot)$.

7. Phase-V protocol for U_i ($1 \leq i \leq m$): At this point, U_i (i.e., U_{i,j_i}) has received $(\alpha'_{1,i}, \ldots, \alpha'_{i-1,i}, \alpha'_{i+1,i}, \ldots, \alpha'_{m,i})$.

   Case I: The preliminary handshake is successful. Decrypt them to obtain the corresponding plaintexts $\delta'_{1,i}, \ldots, \delta'_{i-1,i}, \delta'_{i+1,i}, \ldots, \delta'_{m,i}$.

   (a) For every $\tau \in \{1, \ldots, m\} \setminus \{i\}$
       i. For every $b \in \{1, \ldots, m\} \setminus \{i, \tau\}$, encrypt to obtain $\alpha^*_{\tau,i,b} = \mathsf{sEnc}(g_{\pi_{i,j_i}}(2), \mathsf{pEnc}(pk_{U_{b,j_i}}, \delta'_{\tau,i}))$.

   (b) Broadcast $\{\{\alpha^*_{\tau,i,b}\}_{b \in \{1,\ldots,m\} \setminus \{i,\tau\}}\}_{\tau \in \{1,\ldots,m\} \setminus \{i\}}$.

   Case II: The preliminary handshake is not successful. Then broadcast $\{\{\alpha^*_{\tau,i,b}\}_{b \in \{1,\ldots,m\} \setminus \{i,\tau\}}\}_{\tau \in \{1,\ldots,m\} \setminus \{i\}}$ where $\alpha^*_{\tau,i,b}$ is uniformly selected from the ciphertext space w.r.t. sEnc and $\mathsf{pEnc}(pk_{U_{b,j_i}}, \cdot)$.

8. Phase-VI protocol for U_i ($1 \leq i \leq m$): At this point U_i (i.e., U_{i,j_i}) has received $\{\{\alpha^*_{\tau,i,b}\}_{b \in \{1,\ldots,m\} \setminus \{i,\tau\}}\}_{\tau \in \{1,\ldots,m\} \setminus \{i\}}$.

   Case I: The preliminary handshake is successful.

   (a) Decrypt them to obtain $\Theta' = \{\{\delta'_{\tau,b}\}_{b \in \{1,\ldots,m\} \setminus \{i,\tau\}}\}_{\tau \in \{1,\ldots,m\} \setminus \{i\}}$.

   (b) Set $\pi'_i = (\oplus_{\delta' \in \Theta'} \delta') \oplus (\oplus_{\tau \in \{1,\ldots,m\} \setminus \{i\}} \delta'_{i,\tau}) \oplus (\oplus_{\tau \in \{1,\ldots,m\} \setminus \{i\}} \delta'_{\tau,i})$.

   (c) Broadcast (i, σ'_i), where $\sigma'_i = f_{\pi'_i}(i, j_i, \Gamma'_1, \ldots, \Gamma'_m)$.

   (d) For every $\tau \in \{1, \ldots, m\} \setminus \{i\}$: If $\sigma'_\tau = f_{\pi'_i}(\tau, j_i, \Gamma'_1, \ldots, \Gamma'_m)$, this is a successful handshake with m distinct parties; otherwise, it is not with m distinct parties.

   Case II: The preliminary handshake is not successful. Broadcast (i, σ'_i) where σ'_i is randomly chosen from the range of $\{f_k\}$, and then quit.

Security: Proof of the following theorem is left to the full paper [28].

Theorem 1. *Assume that the public key cryptosystems, the signature schemes, and the pseudorandom functions are secure as specified in Section 2. Then the above scheme is a secure k-anonymous secret handshake scheme in the random oracle model.*

Efficiency Analysis and Improvement: The handshake protocol itself has 6 round-trips excluding the exchange of the plaintext nonces $n_1, \ldots, n_m$, which could have been done before the handshake protocol is executed (e.g., when parties exchange their cipher suits). We only consider the complexity involving

public key cryptography (which dominates the overall cost). Each party needs to verify $O(mw)$ signatures of the public key certificates (which only corresponds to a naive implementation), execute $O(m^2)$ encryptions if the ciphertext spaces can be more efficiently sampled than conducting encryption as is the case with the known notion of dense encryption functions, (otherwise $O(mw)$ encryptions suffice), and $O(m^2)$ decryptions. Each party needs to send $O(mw)$ ciphertexts.

Note that the performance can be improved [28].

5   Symmetric Key Cryptosystems Based Construction

As in the case of two-party case of [27], we notice that the basic idea underlying the above multi-party secret handshake scheme based on public key cryptosystems also applies to the setting of symmetric key cryptosystems. In this section we sketch such a scheme. This scheme is based on the key pre-distribution scheme of [7,6] which we now briefly review: A trusted third party chooses a bivariate symmetric polynomial $poly(x, y)$ of degree t over $\mathbb{F}_q$, where t is the tolerated number of corrupt users. A user with a unique identity i holds $poly(i, y)$. Then two users i and j can derive a common secret $poly(i, j) = poly(j, i)$.

The k-anonymous multi-party secret handshake scheme based on symmetric key cryptosystems is analogous to the one based on public key cryptosystems (i.e., each group is managed by a trusted third party or TTP just as in the key pre-distribution case, the users utilize the rSelect and mSelect to form crowds, no TTP is involved in a secret handshake, etc.), except that a TTP may also adopt a CRL to publish the identities that have been revoked.

6   Conclusion

We presented two multi-party secret handshake schemes: one detailed construction based on public key cryptosystems and one sketched construction based on symmetric key cryptosystems. The added requirements compared to the two party case ([27]) are non-trivial due to new attacks in this setting.

References

1. Aiello, W., Bellovin, S., Blaze, M., Ioannidis, J., Reingold, O., Canetti, R., Keromytis, A.: Efficient, dos-resistant, secure key exchange for internet protocols. In: Proc. of ACM CCS 2002, pp. 48–58
2. Balfanz, D., Durfee, G., Shankar, N., Smetters, D., Staddon, J., Wong, H.: Secret handshakes from pairing-based key agreements. In: Proc. of 2003 IEEE Symposium on Security and Privacy
3. Bellare, M., Boldyreva, A., Desai, A., Pointcheval, D.: Key-privacy in public-key encryption. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 566–582. Springer, Heidelberg (2001)

4. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Proc. ACM CCS 1993, pp. 62–73

5. Bellare, M., Rogaway, P.: Optimal asymmetric encryption. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 92–111. Springer, Heidelberg (1995)

6. Blom, R.: An optimal class of symmetric key generation systems. In: Beth, T., Cot, N., Ingemarsson, I. (eds.) Advances in Cryptology. LNCS, vol. 209, pp. 335–338. Springer, Heidelberg (1985)

7. Blundo, C., DeSantis, A., Herzberg, A., Kutten, S., Vaccaro, U., Yung, M.: Perfectly-secure key distribution for dynamic conferences. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 471–486. Springer, Heidelberg (1993)

8. Boyd, C., Mao, W., Paterson, K.: Deniable authenticated key establishment for internet protocols

9. Castelluccia, C., Jarecki, S., Tsudik, G.: Secret handshakes from ca-oblivious encryption. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 293–307. Springer, Heidelberg (2004)

10. Chaum, D.: Blind signatures for untraceable payments. In: Chaum, D. (ed.) CRYPTO 1982, pp. 199–203. Springer, New York (1983)

11. Chaum, D., Van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991)

12. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 174–187. Springer, Heidelberg (1985)

13. 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)

14. Dolev, D., Dwork, C., Naor, M.: Non-malleable cryptography. In: Proc. of ACM STOC 1991, pp. 542–552

15. Fujisaki, E., Okamoto, T., Pointcheval, D., Stern, J.: Rsa-oaep is secure under the rsa assumption. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 260–274. Springer, Heidelberg (2001)

16. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. Journal of the ACM 33(4), 792–807 (1986)

17. Goldwasser, S., Micali, S., Rivest, R.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Computing 17(2), 281–308 (1988)

18. Harkins, D., Carrel, D.: RFC 2409: The Internet Key Exchange (IKE). Internet Activities Board (1998)

19. Jarecki, S., Kim, J., Tsudik, G.: Authentication for Paranoids: Multi-Party Secret Handshakes. In: Zhou, J., Yung, M., Bao, F. (eds.) ACNS 2006. LNCS, vol. 3989, pp. 325–339. Springer, Heidelberg (2006)

20. Katz, J., Yung, M.: Complete characterization of security notions for probabilistic private-key encryption. In: Proc. of ACM STOC 2000, pp. 245–254

21. Krawczyk, H.: Sigma: The 'sign-and-mac' approach to authenticated diffie-hellman and its use in the ike-protocols. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 400–425. Springer, Heidelberg (2003)

22. Naor, M.: Deniable ring authentication. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 481–498. Springer, Heidelberg (2002)

23. Rackoff, C., Simon, D.R.: Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 433–444. Springer, Heidelberg (1992)

24. Rivest, R., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001)

25. Sweeney, L.: *k*-anonymity: A model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems 10(5), 557–570 (2002)
26. Tsudik, G., Xu, S.: A flexible framework for secret handshakes. In: Danezis, G., Golle, P. (eds.) PET 2006. LNCS, vol. 4258, pp. 295–315. Springer, Heidelberg (2006) (a one-page abstract appeared in ACM PODC 2005)
27. Xu, S., Yung, M.: k-anonymous secret handshakes with reusable credentials. In: Proc. ACM CCS 2004, pp. 158–167
28. Xu, S., Yung, M.: k-anonymous multi-party secret handshakes (2007), http://www.cs.utsa.edu/~shxu

Using a Personal Device to Strengthen Password Authentication from an Untrusted Computer[⋆]

Mohammad Mannan and P.C. van Oorschot

School of Computer Science
Carleton University, Ottawa, Canada
{mmannan, paulv}@scs.carleton.ca

Abstract. Keylogging and phishing attacks can extract user identity and sensitive account information for unauthorized access to users' financial accounts. Most existing or proposed solutions are vulnerable to session hijacking attacks. We propose a simple approach to counter these attacks, which cryptographically separates a user's long-term secret input from (typically untrusted) client PCs; a client PC performs most computations but has access only to temporary secrets. The user's long-term secret (typically short and low-entropy) is input through an independent personal trusted device such as a cellphone. The personal device provides a user's long-term secrets to a client PC only after encrypting the secrets using a pre-installed, "correct" public key of a remote service (the intended recipient of the secrets). The proposed protocol (`MP-Auth`) realizes such an approach, and is intended to safeguard passwords from keyloggers, other malware (including rootkits), phishing attacks and pharming, as well as to provide transaction security to foil session hijacking. We report on a prototype implementation of MP-Auth, and provide a comparison of web authentication techniques that use an additional factor of authentication (e.g. a cellphone, PDA or hardware token).

1 Introduction

Passwords enjoy ubiquitous use for online authentication. Although many more secure (typically also more complex and costly) authentication protocols have been proposed, the use of passwords for Internet user authentication remains predominant. Due to the usability and ease of deployment, most financial transactions over the Internet are authenticated through a password. Hence passwords are a prime target of attackers, for economically-motivated exploits including those targeting online bank accounts and identity theft.

Online banking – as one example of highly critical Internet services – often requires only a bank card number (as *userid*) and password. Users input these credentials to a bank website to access their accounts. An attacker can easily collect these long-term secrets by installing a keylogger program on a client PC, or embedding a JavaScript keylogger [29] on a phishing website. In today's

---

[⋆] Version: March 30, 2007.

Internet environment, software keyloggers are typically installed on a user PC along with common malware and spyware [25]. An increasing number of phishing sites also install keyloggers on user PCs, even when users do not download or click any link on those sites [1]. Client security is a big problem, regardless of the software/hardware platform used, as when plaintext sensitive information is input to a client PC, such malware has instant access, compromising (reusable) long-term secrets. We argue that for some common applications, passwords are too important to input directly to a typical user PC on today's Internet; and that the user PC should no longer be trusted with such plaintext long-term secrets, which are intended to be used for user authentication to a remote server.

To safeguard a long-term password, we build on the following simple idea: use a hand-held personal device, e.g., a cellphone or PDA to encrypt the password (combined with a server generated random challenge) under the public key of an intended server, and relay through a (possibly untrusted) PC only the encrypted result in order to login to the server website. This simple challenge-response effectively turns a user's long-term password into a one-time password in such a way that long-term passwords are not revealed to phishing websites, or keyloggers on the untrusted PC.

The resulting protocol, called `MP-Auth` (short for *M*obile *P*assword *Auth*entication), is proposed primarily to protect a user's long-term password input through an untrusted (or rather, untrustworthy) client PC. For usability and other reasons, the client PC is used for the resulting interaction with the website, and performs most computations (e.g. session encryption, HTML rendering etc.) but has access only to temporary secrets. The capabilities we require from a mobile device include encryption, alpha-numeric keypad, short-range network connection (wire-line or Bluetooth), and a small display. Although we highlight the use of a cellphone, the protocol can be implemented using any similar "trustworthy" device (e.g. PDAs or smart phones), i.e., one free of malware. There are known attacks against mobile devices [11], but the trustworthiness of such devices is currently more easily maintained than a PC, in part because they contain far less software; see Section 3.2 for further discussion of mobile device security. The use of a mobile device in MP-Auth is intended to protect user passwords from easily being recorded and forwarded to malicious parties, e.g., by keyloggers installed on untrustworthy commodity PCs.

Another simple attack to collect user passwords is phishing. Although phishing attacks have been known for at least 10 years (see [10]), few, if any, anti-phishing solutions exist today that are complete and deployable. In MP-Auth, we encrypt a password with the "correct" public key of a web server (e.g. a bank), so that the password is not revealed to any phishing websites. MP-Auth is intended to protect passwords from keyloggers as well as various forms of phishing (including deceptive malware, DNS-based attacks or *pharming*, as well as false bookmarks). New malware attacks (*bank-stealing Trojans* or *session-hijacking*, e.g. Win32.Grams [5]; see also CERT [22]) attempt to perform fraudulent transactions in real-time after a user has logged in, instead of collecting userids and passwords for later use. Most existing or proposed solution techniques are suscep-

tible to these new attacks, e.g., Phoolproof [27] (presented in FC'06), and two-factor authentication such as a password and a passcode generator token (e.g. SecurID). MP-Auth protects against session hijacking, by providing transaction integrity through a transaction confirmation step. Unlike standard two-factor techniques, MP-Auth does not store any secret on the mobile device.

Much of the related work in the literature concerns the trustworthiness of *public* computers, e.g., in Internet cafés and airport lounges. Home computers are generally assumed to be trusted. Solutions are primarily designed to deal with the problem of untrusted computers in public settings. In reality, most user PCs are not safe anywhere; an improperly patched computer – home or public – generally survives only minutes[1] when connected to the Internet. There are also now many anti-phishing proposals (e.g. [29], [36]), and software "tools" designed to detect spoofed websites (e.g. eBay toolbar, SpoofGuard, Spoofstick, Netcraft toolbar). However, most of these are susceptible to keylogging attacks.[2] On the other hand, several authentication schemes which use a trusted personal device, generally prevent keyloggers, but do not help against phishing or session hijacking attacks. In contrast, the goal of MP-Auth is to protect passwords from both keyloggers and phishing sites, and provide transaction security.

Our Contributions. We propose MP-Auth, a protocol for online authentication using a personal device such as a cellphone in conjunction with a PC. The protocol provides the following benefits without requiring a trusted proxy (e.g. [7]), or storing a long-term secret on a cellphone (e.g. Phoolproof [27]).

1. KEYLOGGING PROTECTION. A client PC does not have access to long-term user secrets. Consequently keyloggers (software or hardware) on the PC cannot access critical passwords.
2. PHISHING PROTECTION. Even if a user is directed to a spoofed website, the website will be unable to decrypt a user password. Highly targeted phishing attacks (*spear phishing*) are also ineffective against MP-Auth.
3. PHARMING PROTECTION. In the unlikely event of domain name hijacking [14], MP-Auth does not reveal a user's long-term password to attackers. It also protects passwords when the DNS cache of a client PC is poisoned.
4. TRANSACTION INTEGRITY. With the transaction confirmation step (see Section 2) in MP-Auth, a user can detect any unauthorized transaction during a login session, even when an attacker has complete control over the user PC (through e.g. SubVirt [16] or Blue Pill [30]).
5. APPLICABILITY TO ATMS. MP-Auth is suitable for use in ATMs, if an interface is provided to connect a cellphone, e.g., a wire-line or Bluetooth interface. This can be a step towards ending several types of ATM fraud (see Bond [4] for a list of ATM fraud cases).

---

[1] The average time between attacks is reported to be 5 minutes as of March 26, 2007; see at http://isc.sans.org/survivaltime.html
[2] PwdHash [29] can protect passwords from JavaScript keyloggers, but not software keyloggers on client PCs.

We analyzed MP-Auth using AVISPA [2]; no attacks were found. We have also implemented a prototype of MP-Auth for performance testing.

Organization. The MP-Auth protocol, threat model and operational assumptions are discussed in Section 2. A brief analysis of MP-Auth messages, and circumstances under which MP-Auth fails to provide protection are outlined in Section 3. Discussion on usability and deployment issues related to MP-Auth are provided in Section 4. Section 5 summarizes our MP-Auth prototype implementation. Related work is discussed in Section 6. Section 7 concludes.

2   MP-Auth: A Protocol for Online Authentication

We now describe the MP-Auth protocol, including threat model assumptions.

Threat Model and Operational Assumptions. The primary goals of MP-Auth are to protect user passwords from malware and phishing websites, and to provide transaction integrity. We assume that a bank's "correct" public key is available to users (see below for discussion on public key installation). We assume that mobile devices are malware-free. A browser on a PC uses a bank's SSL certificate to establish an SSL connection with the bank website (as per common current practice). The browser may be duped to go to a spoofed website, or have a wrong SSL certificate of the bank or the verifying certificating authority. The protocol does not protect user privacy (of other than the user's password) from an untrusted PC; the PC can record all transactions, generate custom user profiles etc. Visual information displayed to a user on a PC screen is also not authenticated by MP-Auth, i.e., a malicious PC can display misleading information to a user without being (instantly) detected. Denial-of-service (DoS) attacks are not addressed.

Protocol Steps in MP-Auth. For notation see Table 1. Before the protocol begins, we assume that user U's cellphone M is connected to B (via wire-line or Bluetooth). The protocol steps are described below (see also Fig. 1).

Table 1. Notation used in MP-Auth

|	
U, M, B, S	User, a cellphone, a browser on the user PC, and the server, respectively.
ID_S, ID_U	Server ID and user ID, respectively. ID_U is unique in the server domain.
P	Long-term (pre-established) password shared between U and S.
R_S	Random number generated by S.
$\{data\}_K$	Symmetric (secret-key) encryption of $data$ using key K.
$\{data\}_{E_S}$	Asymmetric (public-key) encryption of $data$ using S's public key E_S.
X, Y	Concatenation of X and Y.
K_{BS}	Symmetric encryption key shared between B and S (e.g. an SSL key).
$f(\cdot)$	A cryptographically secure hash function.
$v(\cdot)$	A visualization function that maps any arbitrary binary string into easy-to-read words [12].
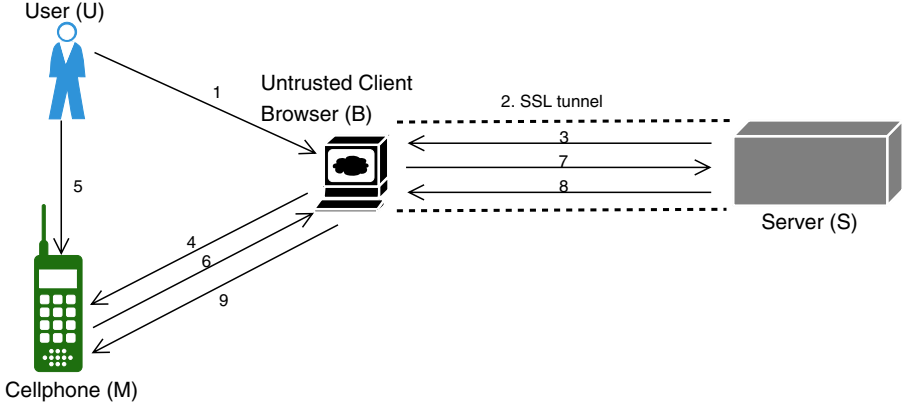
Fig. 1. MP-Auth protocol steps

1. U launches a browser B on the untrusted PC, and visits the bank website S.
2. B and S establish an SSL session; let K_{BS} be the established SSL secret key.
3. S generates a random nonce R_S, and sends the following message to B.

$$B \leftarrow S : \ \{ID_S, R_S\}_{K_{BS}} \qquad (2.1)$$

4. B decrypts message (2.1) and forwards it to M.

$$M \leftarrow B : \ ID_S, R_S \qquad (2.2)$$

We describe an additional step called *session ID verification* (see below) in cases where protecting the integrity of R_S is useful.

5. M prompts the user to input the userid and password for S. A userid (e.g. bank card number) may be stored on the cellphone for convenience; the password should not be stored or auto-remembered.
6. M generates a random secret nonce R_M and encrypts R_M using E_S. M calculates the session key K_{MS} and sends message (2.4) to B (here, the userid ID_U is, e.g., a bank card number).

$$K_{MS} = f(R_S, R_M) \qquad (2.3)$$
$$M \rightarrow B : \ \{R_M\}_{E_S}, \{f(R_S), ID_U, P\}_{K_{MS}} \qquad (2.4)$$

7. B (via SSL) encrypts message (2.4) with K_{BS}, and forwards the result to S.
8. From message (2.4), after SSL decryption, S decrypts R_M using its corresponding private key, calculates the session key K_{MS} (as in equation (2.3)), decrypts the rest of message (2.4), and verifies P, ID_U and R_S. Upon successful verification, S grants access to B on behalf of U. S sends the following message for M to B (indicating login success).

$$B \leftarrow S: \ \{\{f(R_M)\}_{K_{MS}}\}_{K_{BS}} \tag{2.5}$$

9. B forwards $\{f(R_M)\}_{K_{MS}}$ to M. M decrypts to recover $f(R_M)$ and verifies its local copy of R_M. Then M displays success or failure to U.

Transaction Integrity Confirmation. In MP-Auth, M and S establish a session key K_{MS} known only to them; malware on a user PC has no access to K_{MS}. Attackers may modify or insert transactions through the untrusted PC. To detect and prevent such transactions, MP-Auth requires explicit transaction confirmation by U (through M). The following messages are exchanged (after step 9) for confirmation of a transaction with summary details T (R_{S1} is a server generated random nonce, used to prevent replay).

$$M \xleftarrow{\{T, R_{S1}\}_{K_{MS}}} B \xleftarrow{\{\{T, R_{S1}\}_{K_{MS}}\}_{K_{BS}}} S \tag{2.6}$$

$$M \xrightarrow{\{f(T, R_{S1})\}_{K_{MS}}} B \xrightarrow{\{\{f(T, R_{S1})\}_{K_{MS}}\}_{K_{BS}}} S \tag{2.7}$$

M displays T to U in a human-readable way (e.g. "Pay \$10 to Vendor V from the Checking account"), and asks for confirmation (yes/no). When the user confirms T, the confirmation message (2.7) is sent from M to S (via B). From message (2.7), S retrieves $f(T, R_{S1})$, and verifies with its local copy of T and R_{S1}. Upon successful verification, T is committed. Instead of initiating a confirmation step after each transaction, transactions may be confirmed in batches (e.g. four transactions at a time); then, T will represent a batch of transactions in the above message flows.

In an environment where a client machine is less likely to have malware, e.g., an ATM, transaction confirmation may not be needed, if the session ID verification step (see below) is implemented. Also, some transactions may not require confirmation. For example, setting up an online bill payment for a phone company should require user confirmation, but when paying a monthly bill to that account, the confirmation step can be omitted. Fund transfers between user accounts without transaction confirmation may pose no significant risks to users. A bank may configure the set of sensitive transactions that will always require the confirmation step (a user may also add to that set).

Session ID Verification. To detect modification to R_S (when being forwarded to M), we add a session ID verification step after step 4. Both B and M compute a session ID $sid = v(R_S)$. B and M display sid to U. U proceeds only if both session IDs are the same. To minimize user errors, M shows a list of session IDs (one derived from R_S and others chosen randomly), and asks U to select the correct sid corresponding to the one displayed on B.

We assume that users will be able to distinguish differences in sid, especially when sid is easily human-verifiable, e.g., plain English words, distinct images. Note that malware on a PC can display any arbitrary sequence of words or images. Hence the session ID verification step may only help for ATMs (where we

assume an attacker may install a false keyboard panel and card reader on an otherwise trustworthy ATM). When a user accesses an online bank website from a PC, the transaction confirmation step must be implemented; omitting session ID verification in such a case may allow attackers (view-only) access to the user account, but the attackers cannot perform any (meaningful) transaction. (Note that for only viewing a user's transactions, attackers can deploy simple malware on the user PC to capture images of web pages containing the transactions.)

Password Setup/Renewal. In order to secure passwords from keyloggers during password renewal, we require that the password is entered through the cellphone keypad. We assume that the initial password is set up via a trustworthy out-of-band method (e.g. regular phone, postal mail), and U attempts a password renewal after successfully logged into S (i.e. K_{MS} has been established between M and S). The following message is forwarded from M to S (via B) during password renewal (P_{old} and P_{new} are the old and new passwords respectively).

$$M \xrightarrow{X, \text{ where } X = \{ID_U, P_{old}, P_{new}\}_{K_{MS}}} B \xrightarrow{\{X\}_{K_{BS}}} S \quad (2.8)$$

Public Key Installation. One of the greatest practical challenges of deploying public key systems is the distribution and maintenance (e.g. revocation, update) of public keys. MP-Auth requires a service provider's public key to be distributed (and updated when needed) and installed into users' cellphones. The distribution process may vary depending on service providers; we recommend that it not be primarily Internet-based. Considering banking as an example, we visualize the following key installation methods (but note that we have not user-tested these for usability): (i) at a bank branch, preferably during an account setup, (ii) through in-branch ATM interfaces (hopefully free of "fake" ATMs), (iii) through a cellphone service (authenticated download) as data file transfer.

A challenge-response protocol or integrity cross-checks (using a different channel, e.g., see [34]) should be used to verify the public key installed on a cellphone, in addition to the above procedures. For example, the bank may publish its public key on the bank website, and users can cross-check the received public key (e.g. comparing *visual hashes* [28]).

Requirements and Drawbacks of MP-Auth. MP-Auth requires users possess a malware-free (see Section 3.2) personal device. Public keys of each target website (e.g. bank) must be installed on the personal device. (We assume that there are only a few financially critical websites that a typical user deals with.) The correctness, i.e., integrity of installed public keys must also be maintained. A communication channel between a personal device and PC is needed, in such a way that malware on the PC cannot infect the personal device.[3] For ATMs, users must compare easy-to-read words [12] or easily distinguishable images [28] generated from random binary strings.

---

[3] The first *crossover* virus was reported [23] in February 2006.

3   Security and Attack Analysis

In this section, we provide a brief informal security analysis of MP-Auth. We motivate a number of design choices in MP-Auth messages and their security implications. We also list successful but less likely attacks against MP-Auth.

As a confidence building step, we have tested MP-Auth using the AVISPA (Automated Validation of Internet Security Protocols and Applications) [2] analysis tool, and found no attacks. AVISPA is positioned as an industrial-strength technology for the analysis of large-scale Internet security-sensitive protocols and applications. AVISPA test code for MP-Auth is available [17]. We have not at this point carried out other formal analyses or security proofs for MP-Auth.

3.1   Partial Message Analysis and Motivation

Here we provide motivation for various protocol messages and message parts. In message (2.1), S sends a fresh R_S to B, and B forwards ID_S, R_S to M. ID_S is included in message (2.2) so that M can choose the corresponding public key E_S. When U starts a session with S, a nearby attacker may start a parallel session from a different PC, and grab M's response message (2.4) (off-the-air, from the Bluetooth connection) to login as U. However, as S generates a new R_S for each login session (i.e. U and the attacker receive different R_S from S), sending message (2.4) to S by any entity other than B would cause a login failure.

The session key K_{MS} shared between M and S, is known only to them. Both M and S influence the value of K_{MS} (see equation (2.3)), and thus a sufficiently random K_{MS} is expected if either of the parties is honest (as well as capable of generating secure random numbers); i.e., if a malicious party modifies R_S to be 0 (or other values), K_{MS} will still be essentially a random key when M chooses R_M randomly. To retrieve P from message (2.4), an attacker apparently must guess K_{MS} (i.e. R_M) or S's private key. If both these quantities are sufficiently large (e.g. 160-bit R_M and 1024-bit RSA key E_S) and random, an offline dictionary attack on P becomes computationally infeasible. We encrypt only a small random quantity (e.g. 160-bit) by E_S, which should always fit into one block of a public key cryptosystem (including elliptic curve). Thus MP-Auth requires only one public key encryption. Browser B does not have access to K_{MS} although B helps M and S establish this key. With the transaction integrity confirmation step, all (important) transactions must be confirmed from M using K_{MS}; therefore, any unauthorized (or modified) transaction by attackers will fail as attackers do not have access to K_{MS}.

3.2   Attacks Against MP-Auth

Although MP-Auth apparently protects user passwords from malware installed on a PC or phishing websites, here we discuss some other possible attacks against MP-Auth which, if successful, may expose a user's plaintext password.

a) Mobile Malware. We have stated the requirement that the personal (mobile) device be trusted. An attack could be launched if attackers can compromise

mobile devices, e.g., by installing a (secret) keylogger. Malware in mobile networks is increasing as high-end cellphones (smart phones) contain millions of lines of code. For example, a Sept. 2006 study [11] reported that the number of existing malware for mobile devices is nearly 162 (in comparison, there are more than 100,000 viruses in the PC world[4]). Worms such as Cabir [8] are designed to spread in smart phones by exploiting vulnerabilities in embedded operating systems. Regular cellphones which are capable of running J2ME MIDlets have also been targeted, e.g., by the RedBrowser Trojan [9]. However, currently cellphones remain far more trustworthy than PCs, thus motivating our proposal.

In the future, as mobile devices increasingly contain much more software, the requirement of trustworthy cellphones becomes more problematic, and their use for sensitive purposes such as online banking makes them a more attractive target. Limited functionality devices (with less software, implying more trustworthy) may then provide an option for use with MP-Auth. Even if MP-Auth is implemented in such a special-purpose (or lower functionality) device, the device can hold several public keys for different services; in contrast, users may require a separate passcode generator for each service they want to access securely in standard two-factor authentication proposals. Another possibility of restricting mobile malware may be the use of micro-kernels [13], formally verifiable OS kernels [33], protections against virtual-machine based rootkits (VMBRs) [16], or a virtualized Trusted Platform Module (vTPM) [31] on cellphones to restore a trustworthy application environment. The Trusted Computing Group's (TCG's) Mobile Phone Work Group (MPWG) is currently developing specifications [24] for securing mobile phones. Anti-virus software (e.g. Trend Micro [32]) for mobile platforms may also help maintain trustworthiness of cellphones. Malware targeting mobile phones is still limited, and leveraging the experience of working to secure traditional PC platforms may help us achieve a relatively secure mobile computing environment. However, considering the current state of mobile phone security, MP-Auth would perform better on devices whose software upgrade is tightly controlled (e.g. only allowing applications which are digitally signed by a trustworthy vendor).

b) Common-Password Attacks. Users often use the same password for different websites. To exploit such behavior, in a common-password attack, attackers may break into a low-security website to retrieve userid/password pairs, and then try those in financially critical websites, e.g., for online banking. MP-Auth itself does not address the common-password problem (but see e.g., PwdHash [29]).

c) Social Engineering. Some forms of social engineering remain a challenge to MP-Auth (and apparently, other authentication schemes using a mobile device). For example, malware might prompt a user to enter the password directly into an untrusted PC, even though MP-Auth requires users to enter passwords only into a cellphone. In a "mixed" phishing attack,[5] emails are sent instructing users to call a phone number which delivers, by automated voice response, a message that

---

mimics the target bank's own system, and asks callers for account number and PIN. User habit or user instruction may provide limited protection against these.

4   Usability and Deployment

In this section, we discuss usability and deployment issues related to MP-Auth. Usability is a great concern for any protocol supposed to be used by general users, e.g., for Internet banking and ATM transactions. In MP-Auth, users must connect a cellphone to a client PC. This step is more user-friendly when the connection is wireless, e.g., Bluetooth, than wire-line. Then the user browses to a bank website, and enters into the cellphone the userid and password for the site (step 5 in MP-Auth, see Section 2). In ATMs, the password is entered if session ID verification is successful. We also assume that typing a userid and password on a cellphone keypad is acceptable in terms of usability, as many users are accustomed to type SMS messages or have been trained by BlackBerry/Treo experience. However, verification of session ID and transactions may be challenging to some users. We have not conducted any user study to this end.

During authentication the cryptographic operations a cellphone is required to perform in MP-Auth include: one public key encryption, one symmetric encryption and one decryption, one random number generation, and three cryptographic hash operations. The most expensive is the one public key encryption, which is a relatively cheap RSA encryption with short public exponent in our application; see Section 5 for concrete results. We now discuss other usability and deployment aspects which may favor MP-Auth (see also Section 6).

1. As it appears from the current trend in online banking (see [18]), users are increasingly required to use two-factor authentication (e.g. with a separate device such as a SecurID passcode generator) for login. Hence using an existing mobile device for online banking relieves users from carrying an extra device. Also, a user might otherwise require multiple hardware tokens for accessing different online accounts (from different banks).
2. The usability of four login techniques has been studied by Wu et al. [35] – two that send a one-time password as an SMS message, visually checking the session names displayed on the phone and untrusted PC, and choosing the correct session ID from a list of choices on the cellphone. Typing a one-time password is least preferred, yet in most two-factor authentication methods in practice, users must do so. In contrast, MP-Auth requires users to enter only long-term passwords. MP-Auth may also require users to compare session IDs by choosing from a list, which is reported to be more secure (the least spoofable of all) and easier than typing a one-time password [35].
3. MP-Auth offers cost efficiency for banks – avoiding the cost of providing users with hardware tokens (as well as the token maintenance cost). The software modification at the server-end is relatively minor; available SSL infrastructure is used with only three extra messages (between a browser and server) beyond SSL. MP-Auth is also compatible with the common SSL

setup, i.e., a server and a client authenticate each other using a third-party-signed certificate and a user password respectively.

4. Several authentication schemes involving a mobile device store long-term secrets on the device. Losing such a device may pose substantial risk to users. In contrast, losing a user's cellphone is inconsequential to MP-Auth assuming no *secret* (e.g. no "remembered password") is stored on the phone.

5. Public key distribution and renewal challenges usability in any PKI. Key updating is also troublesome for banks. However, key renewal is an infrequent event; we assume that users and banks can cope with this process once every two to three years. If key updates are performed through the mobile network or selected ATMs (e.g. within branch premises), the burden of key renewal is largely distributed. For comparison, hardware tokens (e.g. SecurID) must be replaced approximately every two to five years.

The above suggests that compared to available two-factor authentication methods, MP-Auth may be as usable or better. However, we hesitate to make strong statements without usability tests (c.f. [6]).

5   Implementation and Performance

We developed a prototype of the main authentication and session key establishment parts of MP-Auth to evaluate its performance. Our prototype consists of a web server, a Firefox Extension, a desktop client, and a MIDlet on the cellphone. We set up a test web server (bank), and used PHP `OpenSSL` functions and `mcrypt` module for the server-side cryptographic operations. The Firefox Extension communicates between the web server and desktop client. The desktop client forwards messages to and from the cellphone over Bluetooth. We did not have to modify the web server or Firefox browser for MP-Auth besides adding PHP scripts to the login page (note that Phoolproof [27] requires browser modifications). We used the `BlueZ` Bluetooth protocol stack for Linux, and Rococosoft's Impronto Developer Kit for Java. We developed a MIDlet – a Java application for Java 2 Micro Edition (J2ME), based on the Mobile Information Device Profile (MIDP) specification – for a Nokia E62 phone. For cryptographic

Fig. 2. MP-Auth login

operations on the MIDlet, we used the Bouncy Castle Lightweight Crypto API.

To measure login performance, we used MP-Auth for over 200 successful logins, and recorded the required time (excluding the user input time, i.e. userid and password). We carried out similar tests for regular SSL logins. The results are summarized in Table 2. We use `RSA` 1024-bit and `AES-128` (CBC) for public and symmetric key encryption respectively. `SHA-1` (160-bit) is used as hash function,

and `/dev/urandom` and `Secure-Random` (Java) are used as sources of randomness. Although regular SSL login is almost eight times faster than MP-Auth, on average, it takes less than a second for MP-Auth login. We believe that this

Table 2. Performance comparison between MP-Auth and regular SSL login

	Avg. Time (s)	[Min, Max] (s)
MP-Auth	0.62	[0.34, 2.28]
Regular SSL	0.08	[0.06, 0.22]

added delay would be acceptable, given that entering a userid and password takes substantial additional time.

6   Related Work

The most common types of existing online authentication techniques include password-only authentication, two-factor authentication, and transaction security mechanisms (e.g. secret SMS to a user's cellphone). Several solutions using a trusted device have been proposed, e.g., Phoolproof [27], BitE [20], camera-based authentication [7]. Due to page limits, our comparison of MP-Auth with related work is limited here; for more extensive discussion, see [18].

In contrast to two-factor authentication methods, by design MP-Auth does not provide attackers any window of opportunity when authentication messages (i.e. collected regular and one-time passwords of a user) can be replayed to login as the legitimate user and perform transactions on the user's behalf. The key observation is that, through a simple challenge-response, message (2.4) in MP-Auth (Section 2) effectively turns a user's long-term static password into a one-time password in such a way that long-term passwords are not revealed to phishing websites, or keyloggers on an untrusted PC. In contrast to transaction security mechanisms, MP-Auth protects both large and small transactions. Also, MP-Auth does not require text or voice communications airtime for web authentication or transaction security. (See also Section 4 for more comparison on usability and deployment issues.)

Balfanz and Felten [3] introduced the *splitting trust* paradigm to split an application between a small trusted device and an untrusted computer. Our work is based on such a paradigm where we provide the long-term password input through widely available cellphones, and use the untrusted computer for computationally intensive processing and display.

Parno et al. [27] proposed Phoolproof, a cellphone-based technique to protect users against phishing with less reliance on users making *secure* decisions. With the help of a pre-shared secret – established using an out-of-band channel, e.g., postal mail – a user sets up an account at the intended service's website. The user's cellphone generates a key pair $\{K_U, K_U^{-1}\}$, and sends the public key to the server. The user's private key and server certificate are stored on a cellphone for logins afterward. During login, a user provides userid and password to a website on a browser (as usual), while in the background, the browser and server authenticate (using SSL mutual authentication) through the pre-established client/server public keys in an SSL session (the browser receives the client public key from the cellphone).

Phoolproof assumes that users can correctly identify websites at which they want to set up an account. Users must *revoke* public/private key pairs in case of lost or malfunctioning cellphones, or a replacement of older cellphone models. Expecting non-technical users (e.g. typical bank customers) to understand concepts of creation and revocation of public keys may not be practical. In MP-Auth, users do not have to revoke any key or inform their banks when they lose, break or change their cellphones.

It is also assumed in Phoolproof that the (Bluetooth) channel between a browser and cellphone is secure. Seeing-is-believing (SiB) [21] techniques are proposed to secure local Bluetooth channels, requiring users to take snapshots using a camera-phone, apparently increasing complexity to users. In MP-Auth, we do not rely on the assumption that the local channel (between the cellphone and PC) is secure. Although MP-Auth may require users to visually verify a session ID to secure the local Bluetooth connection (for ATMs, when transaction integrity confirmation is omitted), users are not required to have a camera-phone or to take any picture. Also, Phoolproof does not aim to protect against session hijacking attacks.

Comparing MP-Auth with Existing Literature. Table 3 summarizes a comparison of MP-Auth with several anti-phishing proposals from the literature. An (✗) means a special requirement is needed. An empty box indicates the stated protection is not provided (first three columns) and the stated requirement is not needed (last four columns). A (—) represents non-applicability. (All ✓ and no ✗ would be optimal.) For example, Phoolproof [27] provides protection against phishing and keylogging, but it is vulnerable to session hijacking; it requires a malware-free mobile and stores long-term secrets on the mobile, but does not require a trusted proxy or trusted PC OS. We acknowledge that although this table may provide useful high-level overview, this does not depict an apple-to-apple comparison. Several solutions listed here require a trusted proxy, thus introduce an extra deployment burden, and present an attractive target to

Table 3. Comparing MP-Auth with existing literature. For details, see [18].

	Protection against			Requirement			
	Session-hijacking	Phishing	Key-logging	Trusted proxy	On-device secret	Trusted PC OS	Malware-free mobile
MP-Auth	✓	✓	✓				✗
Phoolproof [27]		✓	✓		✗		✗
BitE [20]			✓		✗	✗	✗
SpyBlock [15]	✓	✓	✓		—	✗	
Three-party [26]	—	—	✓		✗		✗
Camera-based [7]	✓	✓	✓	✗	✗		✗
Web-Auth [35]		✓	✓	✗	✗		✗
Guardian [19]			✓		✗		✗

determined attackers (providing access to many user accounts). Also, fraudsters may increasingly target mobile devices if long-term secrets are stored on them.

7 Concluding Remarks

We have proposed MP-Auth, a protocol for web authentication which is resilient to keyloggers (and other malware including rootkits), phishing websites, and session hijacking. Recently, many small-scale, little-known malware instances have been observed that install malicious software launching keylogging and phishing attacks; these are in contrast to large-scale, high-profile worms like Slammer. One reason for this trend might be the fact that attackers are increasingly targeting online financial transactions.[6] Furthermore, such attacks are fairly easy to launch; for example, attackers can gain access to a user's bank account simply by installing (remotely) a keylogger on a user PC and collecting the user's banking access information (userid and password). MP-Auth is designed to prevent such attacks. MP-Auth primarily focuses on online banking but can be used for general web authentication systems as well as at ATMs. Our requirement for a trustworthy personal device (i.e. free of malware) is important, and becomes more challenging over time, but as discussed in Section 3.2, may well remain viable. In our MP-Auth implementation, cryptographic computations and Bluetooth communications took less than a second for login (excluding the user input time), which we believe is an acceptable delay. Despite a main objective of preventing phishing and keylogging attacks, MP-Auth as presented remains one-factor authentication; thus an attacker who nonetheless learns a user password can impersonate that user. Consequently, the server side of MP-Auth must be trusted to be secure against both from insider attack and break-in.

Users often input reusable critical identity information to a PC other than userid/password, e.g., a passport number, social security number, driver's licence number, or credit card number. Such identity credentials are short, making them feasible to enter from a cellphone keypad. In addition to protecting a user's userid/password, MP-Auth may easily be extended to protect other identity credentials from the reach of online attackers, and thereby might be of use to reduce online identity theft. We believe that the very simple approach on which MP-Auth is based – using a cellphone or similar device to asymmetrically encrypt passwords and one-time challenges – is of independent interest for use in many other applications, e.g., traditional telephone banking directly from a cellphone, where currently PINs are commonly transmitted in-band without encryption.

We reiterate that although based on a very simple idea, MP-Auth has yet to be user-tested for usability; this (together with [18]) is an architecture and state-of-the-art paper. We encourage the security community to pursue alternate proposals for password-based online authentication which simultaneously address phishing, keylogging and session hijacking rootkits.

---

[6] According to a July 2006 report [1], 93.5% of all phishing sites target online financial services, e.g., online banking and credit card transactions.

Acknowledgements

References

1. Anti-Phishing Working Group: Phishing Activity Trends Report (July 2006)
2. Armando et al.: The AVISPA tool for the automated validation of Internet security protocols and applications. In: Etessami, K., Rajamani, S.K. (eds.) CAV 2005. LNCS, vol. 3576, Springer, Heidelberg (2005), http://www.avispa-project.org
3. Balfanz, D., Felten, E.: Hand-held computers can be better smart cards. In: USENIX Security (1999)
4. Bond, M.: Phantom withdrawals: On-line resources for victims of ATM fraud, http://www.phantomwithdrawals.com
5. CA Virus Information Center: Win32.Grams.I (February 2005)
6. Chiasson, S., van Oorschot, P., Biddle, R.: A usability study and critique of two password managers. In: USENIX Security (2006)
7. Clarke, et al.: The untrusted computer problem and camera-based authentication. In: Mattern, F., Naghshineh, M. (eds.) Pervasive Computing. LNCS, vol. 2414, pp. 114–124. Springer, Heidelberg (2002)
8. F-Secure. F-Secure virus descriptions: Cabir (June 2004)
9. F-Secure. F-Secure trojan information pages: Redbrowser.A, (March 2006)
10. Felten, E.W., Balfanz, D., Dean, D., Wallach, D.S.: Web spoofing: An Internet con game. In: National Information Systems Security Conference (October 1997)
11. Gostev, A., Shevchenko, A.: Kaspersky security bulletin, January - June 2006: Malicious programs for mobile devices (September 2006), http://www.viruslist.com
12. Haller, N.: The S/KEY one-time password system. RFC 1760 (February 1995)
13. Heiser, G.: Secure embedded systems need microkernels. ;login: (December 2005)
14. ICANN Security and Stability Advisory Committee: Domain name hijacking: Incidents, threats, risks, and remedial actions (July 2005), http://www.icann.org
15. Jackson, C., Boneh, D., Mitchell, J.: Spyware resistant web authentication using virtual machines, http://crypto.stanford.edu/spyblock
16. King, et al.: SubVirt: Implementing malware with virtual machines. In: IEEE Symposium on Security and Privacy (May 2006)
17. Mannan, M., van Oorschot, P.C.: AVISPA test code for Mobile Password Authentication (MP-Auth), http://www.scs.carleton.ca/~mmannan/mpauth
18. Mannan, M., van Oorschot, P.C.: Using a personal device to strengthen password authentication from an untrusted computer, Technical Report TR-07-11(March 2007), http://www.scs.carleton.ca/research/tech_reports/
19. Margolin, N.B., Wright, M.K., Levine, B.N.: Guardian: A framework for privacy control in untrusted environments. Technical Report 04-37 (University of Massachusetts, Amherst) (June 2004)
20. McCune, J.M., Perrig, A., Reiter, M.K.: Bump in the Ether: A framework for securing sensitive user input. In: USENIX Annual Technical Conference (2006)

21. McCune et al.: Seeing-is-believing: Using camera phones for human-verifiable authentication. In: IEEE Symposium on Security and Privacy (2005)
22. Milletary, J.: Technical trends in phishing attacks. US-CERT, Reading room article, http://www.us-cert.gov/reading_room/phishing_trends0511.pdf
23. Mobile Antivirus Researchers Association: Analyzing the crossover virus: The first PC to Windows handheld cross-infector (2006), http://www.informit.com
24. Mobile Phone Work Group (MPWG): TCG mobile trusted module specification, Draft, version 0.9 (September 2006)
25. Moshchuk, A., Bragin, T., Gribble, S.D., Levy, H.: A crawler-based study of spyware in the web. In: Network and Distributed System Security (NDSS) (2006)
26. Oprea, A., Balfanz, D., Durfee, G., Smetters, D.: Securing a remote terminal application with a mobile trusted device. In: Yew, P.-C., Xue, J. (eds.) ACSAC 2004. LNCS, vol. 3189, Springer, Heidelberg (2004)
27. Parno, B., Kuo, C., Perrig, A.: Phoolproof phishing prevention. In: Di Crescenzo, G., Rubin, A. (eds.) FC 2006. LNCS, vol. 4107, Springer, Heidelberg (2006)
28. Perrig, A., Song, D.: Hash visualization: A new technique to improve real-world security. In: Cryptographic Techniques and E-Commerce (CrypTEC) (July 1999)
29. Ross, B., Jackson, C., Miyake, N., Boneh, D., Mitchell, J.: Stronger password authentication using browser extensions. In: USENIX Security (2005)
30. Rutkowska, J.: Introducing Blue Pill, Presented at SyScan (2006), http://theinvisiblethings.blogspot.com/2006/06/introducing-blue- pill.html
31. Stefan Berger, R.C., Goldman, K.A., Perez, R., Sailer, R., van Doorn, L.: vTPM: Virtualizing the trusted platform module. In: USENIX Security (2006)
32. Trend Micro: Mobile security, http://www.trendmicro.com/en/products/mobile/tmms/evaluate/overview.htm
33. Tuch, H., Klein, G., Heiser, G.: OS verification — now! In: Hot Topics in Operating Systems (June 2005)
34. van Oorschot, P.C.: Message authentication by integrity with public corroboration. In: New Security Paradigms Workshop (NSPW) (September 2005)
35. Wu, M., Garfinkel, S., Miller, R.: Secure web authentication with mobile phones. In: DIMACS Workshop on Usable Privacy and Security Systems (July 2004)
36. Ye, Z.E., Smith, S., Anthony, D.: Trusted paths for browsers. ACM Transactions on Information and System Security (TISSEC) 8(2) (2005)

Scalable Authenticated Tree Based Group Key Exchange for Ad-Hoc Groups

Yvo Desmedt[1,*], Tanja Lange[2,**], and Mike Burmester[3,***]

[1] Information Security, Department of Computer Science,
University College London, UK
y.desmedt@cs.ucl.ac.uk
[2] Department of Mathematics and Computer Science
Technische Universiteit Eindhoven, Netherlands
tanja@hyperelliptic.org
[3] Department of Computer Science
Florida State University, USA
burmester@cs.fsu.edu

Abstract. Task-specific groups are often formed in an ad-hoc manner within large corporate structures, such as companies. Take the following typical scenario: A director decides to set up a task force group for some specific project. An order is passed down the hierarchy where it finally reaches a manager who selects some employees to form the group. The members should communicate in a secure way and for efficiency, a symmetric encryption system is chosen. To establish a joint secret key for the group, a group key exchange (GKE) protocol is used. We show how to use an existing Public Key Infrastructure (PKI) to achieve authenticated GKE by modifying the protocol and particularly by including signatures.

In this paper we recall a GKE due to Burmester and Desmedt which needs only $O(\log n)$ communication and computation complexity per user, rather than $O(n)$ as in the more well-known Burmester-Desmedt protocol, and runs in a constant number of rounds. To achieve authenticated GKE one can apply compilers, however, the existing ones would need $O(n)$ computation and communication thereby mitigating the advantages of the faster protocol. Our contribution is to extend an existing compiler so that it preserves the computation and communication complexity of the non-authenticated protocol. This is particularly important for tree based protocols.

Keywords: Key Distribution, Group Key Exchange, Tree based GKE, Ad-Hoc Groups, Forward Security, Authentication, Anonymity.

1 Introduction

Today several banks have branches worldwide. Moreover, stockowners of these banks often are in different countries. In general, globalization implies that

decision makers are in different locations. So efficient and secure communication within a group is very important. Clearly symmetric systems offer the higher throughput and so all group members must hold a common *secret key*. Equipping each user with a new key is called *group key exchange*. Given the distributed nature of the group, new keys can only be established through an insecure channel. Furthermore, the parties need to be assured that this key is shared with the correct group members, so the users must be authenticated. This shows that in financial cryptography group key exchange is an important primitive.

Another scenario is the need to set up ad-hoc groups. Organizations, such as financial ones, are usually organized in a hierarchical way. Often outside consultants are needed in virtual group meetings. The president of the organization does not know who these experts are. We find ourselves in a situation where members lower in the hierarchy decide who the outside-consultants are that should join the group. Again secure group communication is needed.

Once this symmetric key is established it can be used for basically all communication needs within the group. It not only works as the key in a symmetric system to ensure that data cannot be decrypted but also allows members within the group to communicate anonymously and the key can also be used in MACs to authenticate messages.

A simple solution for authenticated group key exchange is to have one user (the chair) choose the key and exchange it with the next user, who will exchange it with the next, etc., each by using the Diffie-Hellman KE protocol. The cost of this solution is that it requires $O(n)$ rounds and each user has constant communication and computation. We refer to this scenario as the *naive approach*.

So far, several group KE protocols have been proposed, most of which are extensions of the two-party Diffie-Hellman protocol [3,4,9,10,5]. Katz and Yung [12] designed a compiler that transforms any secure group KE protocol into an authenticated KE protocol. The model used is a refinement of models proposed by Bresson et al. in [2]. The additional requirements are: a PKI for digital signatures, and each message issued should be signed and checked by all participants.[1] To avoid replay attacks and loss of intermediate messages without noticing, each group member needs to maintain a counter for messages send during one KE and a signed string must include *all* group participants as well as a random nonce for *each* participant. As an example they consider the Burmester-Desmedt scheme [3,5] which requires a constant number of rounds but has communication complexity[2] $O(n)$ and computational complexity $O(n)$. Since there is another Burmester-Desmedt group key exchange protocol [4], we refer to the one in [3,5] as Burmester-Desmedt I (BD-I).

Although BD-I is very nice in its perfect symmetry, it is rarely used in real life application. As has been pointed out by many authors, e.g. [14, p.86] "One shortcoming of BD is the high communication overhead." It is for this reason that

---

[1] We like to remark that the PKI can also be replaced by an Identity-based Key Infrastructure [6].

[2] Although each sender only sends a constant number of strings (two), the number of received strings is $O(n)$. See Section 2.3 for more details.

tree based group key exchange protocols are considered superior, as we describe further. Tree based group key exchange protocols have become quite popular due to the work of Wong-Gouda-Lam [16]. They actually considered a generalization of trees, i.e., directed acyclic graphs. One should observe that many schemes can be described as trees. Indeed the Boyd and Nieto [1] scheme corresponds to a star which is a special tree. Certain trees provided better efficiency. We now discuss such a scheme.

The Burmester-Desmedt scheme II (BD-II) [4] is one of the schemes which uses a (binary) tree for group KE. Using the properties of trees, it can achieve *logarithmic complexity* while keeping the same *constant number of rounds*. (Note that the work in BD-II predates the one by Wong-Gouda-Lam.) Now if one were to use the Katz-Yung compiler [12] it would mitigate the advantage of BD-II as checking of $O(n)$ signatures would be required. In this contribution we give an *authenticated version of the BD-II group key exchange protocol* with an overall communication and computation complexity of $O(\log n)$ while running in a constant number of rounds. *These ideas extend to other group key exchange protocols, in particular tree based ones with logarithmic complexity.* For this reason we modify the Katz-Yung compiler.

After introducing the BD-II schemes and stating the security model we give a security proof for a passive adversary of one of the Burmester-Desmedt II schemes (although the paper contains several security claims, no proofs are given in [4]). We then extend the scheme to an authenticated KE by modifying the Katz-Yung compiler. The main problem with a direct application of the Katz-Yung compiler is that each party needs to verify $O(n)$ signatures which turns the overall complexity to $O(n)$ even though the underlying protocol allows for $O(\log n)$. However, the joint key is computed by any user U entirely from information provided by its at most $\log n$ ancestors along the tree and only those logarithmically many signatures need to be verified. We present the compiler in the most general setting applicable to any group KE protocol.

The BD-II group KE protocol is non-contributory in the sense that the key does not depend on the contribution of all members. In fact, no protocol with a computation or communication complexity lower than $O(n)$ can be fully contributory if it runs in a constant number of rounds and without delay. However, the Katz-Yung model allows for this since it does *not* deal with *active insiders*. We show that for a passive adversary the security of the protocols has a tight reduction to the DDH problem. Active external adversaries cannot insert messages in the name of a group member as the protocol is authenticated. There is no attack advantage of being a member of the system, i.e. being registered to the system's PKI, if one is not part of the group which is performing the group key exchange. When we speak of insiders we mean actual members of the group which are agreeing on a key.

For the BD-I protocol it has been shown in [10,15,7] that as few as 2 malicious insiders are enough to force the resulting group key to be any element they want, e.g. one known to an outsider, without needing any extra communication during the execution of the GKE. Our presented scheme is no more secure against

active insiders. More specifically, we shall use the same model as Katz-Yung [12] which does not deal with active malicious insiders that attempt to prevent an honest party from obtaining the common group key. The advantage of an active adversary is defined to be the advantage of obtaining the common group key and like Katz and Yung we assume the *"best-case" scenario* of *an adversary who delivers all messages intact to the appropriate recipient(s) as soon as they are sent.* ([12, Section 1.3]). Recent work by Katz and Sun Shin [11] formalizes insider attacks and defines the security of authenticated GKE protocols against malicious insiders in the framework of universal composablility. However, [7] demonstrates that dealing with active insiders is far from trivial. They criticize the model in [11] and present a different one which they consider to be more realistic. We do not want to enter into this controversy and exclude malicious insiders.

We recently noticed an independent result on authenticated tree based GKEs [13]. That paper is also based on the Burmester-Desmedt II GKE even though this is not stated in the paper. One shortcoming of that paper is that they require *each* party to check all signatures on all messages; this implies that they have $O(n)$ computation costs rather than the desired $O(\log n)$. It is because of that difference that the Katz-Yung compiler [12] needs to be adjusted. Furthermore, their group identifiers have length $O(n)$.

The remainder of this paper is organized as follows. We start by stating the security model and surveying one of the BD-II schemes. We then prove its security against passive adversaries and provide an authenticated variant based on the DDH assumption. A comparison with other schemes shows the advantages of our proposal.

2 Models

Unless explicitly mentioned we follow the same lines as Katz and Yung [12] who used the security model for group KE due to Bresson et al. [2]. We first introduce the notations and then briefly mention the oracles the adversary can query depending on the protocol. For full details we refer to [12].

2.1 Participants and Initialization

There is a polynomial-size set $\mathcal{P}$ of potential participants in the group key exchange, any subset of $\mathcal{P}$ may decide to establish a session key. We assume that during an initialization phase each participant in $\mathcal{P}$ runs an algorithm $\mathcal{G}(1^k)$ to generate a pair of public and private keys (PK, SK). The secret key is stored by the user and (certified) public keys are accessible to all participants.

2.2 Adversarial Model

We denote the instance i of user U as Π_U^i, each instance may be used only once. Each instance has associated with it the variables state_U^i, term_U^i, acc_U^i, used_U^i,

pid_U^i, the session ID sid_U^i, and the session key sk_U^i which we now explain. In our model, the partner ID pid_U^i contains a group identifier gid which identifies all partners involved in the current execution of the GKE. The other definitions are as in [2], so state_U^i represents the current (internal) state of instance Π_U^i, term_U^i acc_U^i and used_U^i take boolean values indicating whether the instance has been terminated or accepted or used, respectively. Most of these variables appear only implicitly except for pid_U^i, sid_U^i, and sk_U^i.

The adversary is assumed to have full control over all communication in the network. His interaction is modeled by the following oracles:

- $\mathsf{Send}(U, i, M)$ – to send the message M to instance Π_U^i and output the reply generated by this instance. This oracle may also be used to initiate a key exchange among a group $\{U_1, \ldots, U_n\}$ of users identified by some group identifier gid. The length of gid must correspond to the security parameter. This means that the first round of the KE protocol is executed upon receipt of this message.
- $\mathsf{Execute}(\mathsf{gid})$ – to execute the protocol between unused instances of players $U_1, \ldots, U_n \in \mathcal{P}$ determined by the group identifier gid and to output the transaction of the execution. The adversary has control over the number of players and their identities.
- $\mathsf{Reveal}(U, i)$ – to reveal the session key sk_U^i of player U belonging to instance i.
- $\mathsf{Corrupt}(U)$ – to output the long-term secret key SK_U of player U.
- $\mathsf{Test}(U, i)$ – to be issued the final test. Once the adversary decides that he has enough data he queries the Test oracle for a challenge. A random bit b is generated; if $b = 1$ then the adversary is given sk_U^i, otherwise he receives a random session key.

A *passive adversary* is given access to the $\mathsf{Execute}$, Reveal, $\mathsf{Corrupt}$ and Test oracles, while an *active adversary* is additionally given access to the Send oracle. Both types of adversaries are allowed to make adaptive queries before and after the Test oracle is queried.

Partnering. The session ID sid_U^i equals the concatenation of all messages sent and received by Π_U^i during the course of its execution. For the partner ID we deviate from the suggestion in [12] and *generalize their setting to the situation that not each party communicates with each of the others* during an execution of the protocol. Although this may seem as a slight adaptation of [12] (see also [2]), it allows us to dramatically improve on the efficiency of the schemes.

We assume that the group of users can be identified uniquely by a group identifier gid. This assumption holds true in all network protocols and fits well for hierarchical situations we encounter in the financial world. Then pid_U^i consists of the group identifier and the identities of the players in the group *with which* Π_U^i *interacts* during the KE protocol, i.e., *to which he sends messages or from which he receives messages*. Since the underlying unauthenticated protocol is assumed work, i.e., to provide each user with the same key, this implies that the union of all pid_U^i covers each user involved in the key exchange. Furthermore,

the graph displaying the communication must be connected. This is due to the fact that all users obtain the same key. In our compiler this ensures that each signature is checked by a group member which is connected to all others via checked paths.

Correctness. We require that for all U, U' and i, i' involved in the same key exchange and such that $\mathsf{acc}_U^i = \mathsf{acc}_{U'}^{i'} = \text{TRUE}$, the same valid session key is established $\mathsf{sk}_U^i = \mathsf{sk}_{U'}^{i'} \neq \text{NULL}$.

Security and Freshness. An instance Π_U^i is *fresh* unless one of the following is true: (1) at some point, the adversary queried $\mathsf{Reveal}(U, i)$ or $\mathsf{Reveal}(U', i')$ for any $\Pi_{U'}^{i'}$ in the same group (denoted by gid) as Π_U^i or (2) a query $\mathsf{Corrupt}(V)$ was asked before a query of the form $\mathsf{Send}(U', i', *)$ by V, where V and U' are in pid_U^i. (Note that our definition of pid_U^i only includes those users that U is directly interacting with, i.e., those providing input to the key computation of U).

The event Succ occurs if the attacker is successful, i.e., if he queries the Test oracle on a fresh instance Π_U^i for which $\mathsf{acc}_U^i = \text{TRUE}$ and guesses the bit b correctly. The advantage of attacker $\mathcal{A}$ against protocol P is defined as $\mathsf{Adv}_{\mathcal{A},P(k)} \stackrel{\text{def}}{=} |2 \cdot \mathsf{Pr}[\mathsf{Succ}] - 1|$.

Protocol P is a secure group KE protocol if it is secure against a passive adversary, i.e., for any PPT passive adversary $\mathcal{A}$ the advantage $\mathsf{Adv}_{\mathcal{A},P}(k)$ is negligible. Protocol P is a secure authenticated group KE (AKE) if it is secure against an active adversary.

We use $\mathsf{Adv}_P^{\mathsf{KE}}(t, q_{\mathrm{ex}})$ to denote the maximum advantage of any passive adversary attacking P, running in time t and making q_{ex} calls to the $\mathsf{Execute}$ oracle. For the authenticated group KE we use $\mathsf{Adv}_P^{\mathsf{AKE}}(t, q_{\mathrm{ex}}, q_{\mathrm{s}})$, where q_{s} refers to the number of Send queries. If the scheme achieves forward secrecy we use $\mathsf{Adv}_P^{\mathsf{AKE-fs}}(t, q_{\mathrm{ex}}, q_{\mathrm{s}})$.

The security of the example protocols will be based on the decisional Diffie-Hellman problem (DDHP), let $\mathsf{Adv}_G^{\mathsf{ddh}}(t)$ be the advantage of a PPT adversary against the DDHP in a group $G = \langle g \rangle$ of order ℓ. The DDHP is the problem of distinguishing the distributions of $\{(g^a, g^b, g^{ab}) : a, b \in_R \mathbb{Z}/\ell\mathbb{Z}\}$ from $\{(g^a, g^b, g^c) : a, b, c \in_R \mathbb{Z}/\ell\mathbb{Z}\}$ given g. For a single triple this amounts to deciding whether the triple $(g_1, g_2, g_3) \in G^3$ is a valid Diffie-Hellman triple, i.e., if $g_3 = g_2^{\log_g g_1}$.

2.3   Complexity

Group KE protocols are often carried out in dynamic sets of players. One important feature of good protocols is their scalability. To take this into account we always state the maximal complexity occurring for any user in the system. Furthermore, we consider the number of users n as the important parameter and ignore costs depending on the cryptographic primitive like the size of the underlying group $\langle g \rangle$.

The communication complexity is the maximal amount of information *sent and received* per user. We assume that broadcasting a message does not depend on the number of receivers, however, receiving l *different* messages means cost of l, even if this occurs in one round. Katz and Yung [12] mention *only sent messages* but use the term in the same way. We mention the received messages explicitly to take into account the costs for being online, and for receiving and storing messages. At the same time we also allow users to ignore communication not intended for them. The computation complexity is the maximal amount of computation during one execution of the protocol. In both cases we are interested in the dependence on n, all other variables like group size or security parameter are considered as constant in the big-O estimates.

3   The Burmester-Desmedt Scheme II

We now describe in detail one scheme with logarithmic complexity. This example serves two purposes: on the one hand it gives a good example of the advantage of our compiler over that of Katz and Yung and on the other hand it closes the gap that [4] was published without security proofs.

The BD-II scheme [4] is a compiler that transforms a two-party KE protocol into a multiparty key exchange protocol. There are two variants. The first is sequential with delays. The second is a multicast version with minimal delay.

To ease exposition we focus on the Diffie-Hellman version of the BD-II scheme [4, p. 127] (one can use other two party KE protocols as primitives) and use a cyclic group $G = \langle g \rangle$ of prime order (although the proof remains similar if the last condition is not satisfied). The advantage of BD-II over other group KE protocols is that the communication complexity, the computation complexity and memory reduce from $O(n)$ per party to $O(\log n)$ per party in the multicast version, and to constant in the sequential version which has $O(\log n)$ delay. The number of rounds each party is actively involved in is constant and equal to three for both variants.

In this section we briefly recall the BD-II protocol [4] in the multicast version and give a proof of security of the unauthenticated scheme; a sequential version is given in the Appendix.

Let $U_1, \ldots, U_n$ be the users who want to make a group KE. We now show how their index automatically determines their place in the (almost) binary tree as in Figure 1. This ordered tree has the property that user U_i is at level $\lfloor \log_2(i+1) \rfloor$ (a rooted tree version would need one more multiplication round and is therefore avoided). The set-up can be easily done as the position of user U_i corresponds to the binary expansion of $i + 1$, *i.e.*, if $i + 1 = (0 \ldots 0 i_j i_{j-1} \ldots i_1 i_0)_2$ is the binary expansion with i_j the leftmost non-zero bit then U_i is on level j and his parent has expansion $\text{parent}(i) + 1 = (0 \ldots 0 i_j i_{j-1} i_1)_2$ (one more leading zero). Likewise the children can easily be determined as their expressions are shifted to the left and they have the same initial binary representation with different concatenated tails.
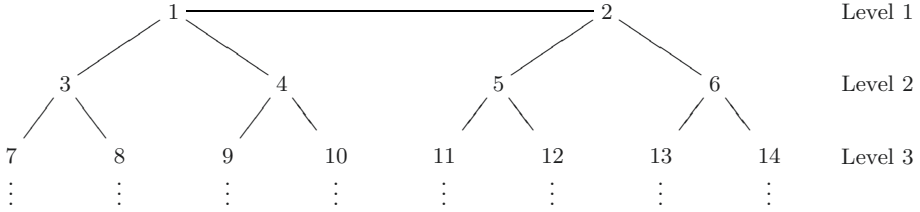
Fig. 1. The binary BD-tree in the BD-II scheme

The vertices 1 and 2 consider their respective opposite as parent. So all but the leaves of the tree each have one parent and two children. To ease notation let ancestors(i) be the set of indices of all ancestors of U_i, including i *but* having removed 1 and 2. Let parent(i), left child(i) and right child(i) be the indices of the parent of U_i and its left child or right child respectively.

Remark 1. One advantage of this scheme is that users only need to know users higher up in the hierarchy. This fits well to common applications of GKE in groups invoked by some manager higher up in the hierarchy who decides the members of the group and their respective function. Once the GKE protocol is started the users have already received some information publicly and their public keys are available in a public database.

While the BD-I scheme requires that *each* user needs to know the position of *every* other user relative to him, our scheme limits this to the knowledge of the positions of at most $O(\log n)$ users.

Note that all this is usually not a problem. The user name in the protocol should contain information about the user's position (the numbers $1, 2, \ldots, n$ can be encoded with $\log n$ bits) and this number is enough to determine the relative positions for both BD-I and BD-II. We only stress this observation since it is often claimed as a disadvantage of BD-II that the position needs to be known even though this is true even more for BD-I.

Protocol 1 (BD-II group KE)
Let $U_1, \ldots, U_n$ be the set of all users who want to generate a common conference key. Assume that they are arranged in the binary tree as in Figure 1. The key exchange is performed in a group $\langle g \rangle$ of order ℓ with generator g.

Step 1 *Each U_i, $i = 1, \ldots, n$, selects $k_i \in_R \mathbb{Z}/\ell\mathbb{Z}$, computes and sends $z_i = g^{k_i}$ to his parent and children.*

Step 2 *Each U_i, $i = 1, \ldots, n$, computes and multicasts to its descendants:*

$$X_{\text{left child}(i)} = (z_{\text{parent}(i)}/z_{\text{left child}(i)})^{k_i}, X_{\text{right child}(i)} = (z_{\text{parent}(i)}/z_{\text{right child}(i)})^{k_i}.$$

Step 3 *Each U_i, $i = 1, \ldots, n$, computes the conference key,*

$$K_i = z_{\text{parent}(i)}^{k_i} \prod_{j \in \text{ancestors}(i)} X_j.$$

Remark 2. Honest users compute the same key, namely $K = g^{k_1 k_2}$.

We prove this claim by induction. First observe that $K_1 = K_2 = g^{k_1 k_2}$. Next let $K_{\text{parent}(i)} = K$. Then since $K_{\text{parent}(i)} = z_{\text{parent}(\text{parent}(i))}^{k_{\text{parent}(i)}} \prod_{j \in \text{ancestors}(\text{parent}(i))} X_j$ it is obvious that we have:

$$K_i = z_{\text{parent}(i)}^{k_i} * (z_{\text{parent}(\text{parent}(i))}^{k_{\text{parent}(i)}})^{-1} * X_i * K_{\text{parent}(i)}.$$

With $X_i = (z_{\text{parent}(\text{parent}(i))}/z_i)^{k_{\text{parent}(i)}}$ and $z_{\text{parent}(i)}^{k_i} = z_i^{k_{\text{parent}(i)}}$ we get $K_i = K$.

4   Proof of Security Against Passive Attacker for BD-II

We first show that an attacker against Protocol 1 can be used to solve the decisional Diffie-Hellman problem The proof is an adaption of Burmester-Desmedt's proof for BD-I in [5].

Theorem 1. *Assuming the Decisional Diffie-Hellman problem is hard, protocol P is a secure group KE protocol. Namely*

$$\mathsf{Adv}_P^{\mathsf{KE}}(t, q_{\text{ex}}) \leq \mathsf{Adv}_G^{\mathsf{ddh}}(t'),$$

where $t' = t + O(|\mathcal{P}|q_{\text{ex}}t_{\text{exp}})$, q_{ex} is the number of Execute *queries, and t_{exp} is the time required to perform exponentiations in G.*

Note that the time for the execution of P is made explicit by the use of t_{exp}.

Proof. Given an algorithm $\mathcal{A}$ against P running in time t we show how to build a distinguisher $\mathcal{D}$ against the DDHP. First consider the case that $\mathcal{A}$ makes a single Execute query.

Let $\mathcal{D}$ be given a triple $(g_1, g_2, g_3) \in \langle g \rangle^3$. Now $\mathcal{D}$ can generate a valid transcript for $\mathcal{A}$ as shown below. Then $\mathcal{D}$ runs $\mathcal{A}$ on this transcript and outputs 1, i.e., the claim that (g_1, g_2, g_3) is a valid Diffie-Hellman triple, if $\mathcal{A}$ outputs 1 and outputs 0 otherwise.

Put $z_1' = g_1$ and $z_2' = g_2$. Randomly choose $c_3, \ldots, c_n \in_R \mathbb{Z}/\ell\mathbb{Z}$ and put $z_i' := z_{\text{parent}(\text{parent}(i))}'g^{-c_i}$ for $i \geq 3$. So, $z_3' := g_2 \cdot g^{-c_3}$, $z_5' := g_1 \cdot g^{-c_5}$, etc.

Consistent X_i''s are obtained by putting $X_i' := (z_{\text{parent}(i)}')^{c_i}$ for $i \geq 3$ as is easy to verify. As the c_i ($i \geq 3$) are distributed uniformly at random, the distribution of z_i' and X_i' is identical to that in P.

The transcript consists of $\mathsf{T} = (z_1', \ldots, z_n', X_3', \ldots, X_n')$. Upon the Test request, $\mathcal{D}$ issues $\mathsf{sk}' = g_3$. Indeed, if sk' is the valid group key then $g_3 = \mathsf{sk}' = z_1'^{\log_g z_2'} = g_1^{\log_g g_2}$, i.e., (g_1, g_2, g_3) is a valid Diffie-Hellman triple. So $\mathcal{D}$ succeeds with the same advantage as $\mathcal{A}$ and needs $(2n - 4)t_{\text{exp}}$ additional time for the exponentiations to generate the transcript.

If more than one execution of the protocol should be allowed, one can easily generate further triples $(g_1^{r_1}, g_2^{r_2}, g_3^{r_1 r_2})$ of the same type as (g_1, g_2, g_3) for random exponents $r_1, r_2 \in_R \mathbb{Z}/\ell\mathbb{Z}$.

Bounding the number n by the total number of participants $|\mathcal{P}|$ the claim follows.   □

Note that the computational complexity of computing the group key under a passive attack for an outsider corresponds to the Computational Diffie-Hellman problem, as is easy to verify (similar as in [5]).

This protocol does not involve any longterm secrets, so Corrupt queries need not be taken into account and the protocol automatically achieves forward security.

5   Authenticated Group Key Exchange

A direct application of the compiler of Katz and Yung [12] would transform Protocol 1 into an authenticated KE protocol with running time and communication complexity $O(n)$. We now provide our adjusted compiler which allows us to stay in $O(\log n)$. To simplify the presentation we assume that the reader is familiar with [12] and will only mention the differences.

To avoid replay attacks, Katz-Yung [12] introduce fresh randomness r_i per user U_i for each execution of the protocol, add a message number for each user, and make the signature contain information on the group of players in the AKE. We follow the same road and so we modify the protocol to always send $U|j|m$ (the user name, the message number and the message) and not only m or $U|m$. Let $\mathsf{Sign}_{SK}(m)$ output the signature on message m under secret key SK and let $\mathsf{Vrfy}_{PK}(m, \sigma) = 1$ if σ is the correct signature on message m under public key PK and 0 otherwise.

We observe that in BD-II user U_i computes the $X_{\mathrm{child}(i)}$ depending only on information from its parent and two children. To compute the group key $\mathsf{sk}_{U_i} = K_i$, user U_i ($i \neq 1, 2$) uses information coming only from the same branch of the binary tree from nodes on levels above U_i while U_1 and U_2 use information by their respective parents. In general, in each group KE protocol there is a clearly defined ordered set of messages used by a specific user U_i and our compiler requires to check only signatures on these messages. In most tree based systems the set of users any specific user communicates with is much smaller than the total set of users; even to the extent that the number of used messages is logarithmic in the total number of users. In BD-I, however, each user processes input from every user.

We formulate the compiler in a more general setting which contains Katz-Yung's as a special case, we use the term "multicast" to stress that not all users need to receive the message. To link the messages to the structure in which the users are arranged we let the set $\mathsf{rel}_U = \{V_1, V_2, \ldots, V_{t_U}\}$ be the set of users whose input is processed by user U at some point in the protocol and U itself. E.g. in the BD-I protocol rel_U is the whole group for any user U while in BD-II rel_U is the set of all ancestors and both children of U; it contains at most $O(\log n)$.

1. During the initialization phase, each party $U \in \mathcal{P}$ generates the verification/signing keys (PK'_U, SK'_U) by running $\mathcal{G}(1^k)$. This is in addition to any keys (PK_U, SK_U) needed as part of the initialization phase of P.

2. Let $U_1, \ldots, U_n$ be the identities of users wishing to establish a joint group key and let gid be their group identifier[3]. Each user U_i chooses some random nonce $r_i \in \{0,1\}^k$ and broadcasts $U_i|0|r_i$. So for each execution of the protocol fresh randomness is used and so replay is not possible.

   Let $\mathsf{rel}_U = \{V_1, V_2, \ldots, V_{t_U}\}$ be as above. Each instance Π_U^j stores the identities and their per-round randomness together with the group ID in $\mathsf{direct}_U^j = (\mathsf{gid}|V_1|r_1|\ldots|V_{t_U}|r_{t_U})$ and stores this as part of the state information.

3. The members of the group now execute the protocol P with the following changes:
   - Whenever instance Π_U^i is supposed to multicast $U|j|m$ as part of protocol P, the instance first computes $\sigma = \mathsf{Sign}_{SK_U'}(j|m|\mathsf{direct}_U^i)$ and then multicasts $U|j|m|\sigma$.
   - Before using message $V|j|m|\sigma$ the instance Π_U^i checks that (1) $V \in \mathsf{pid}_U^i$ [4], (2) j is the next expected sequence number for messages from V, and, finally, (3) that $\mathsf{Vrfy}_{PK_V'}(j|m|\mathsf{direct}_V^i, \sigma) = 1$. If any of these fail, Π_U^i aborts the protocols and sets $\mathsf{acc}_U^i = \mathrm{FALSE}$ and $\mathsf{sk}_U^i = \mathrm{NULL}$. Otherwise, Π_U^i continues as it would in P and uses the message m.

Remark 3. We like to stress that the authentication does not prevent attacks by malicious insiders. Like Katz and Yung state in [12, Section 2.1] these definitions cannot achieve "agreement" in the sense of [8], e. g. since the attacker could stop all communications by denial of service attacks.

Remark 4. The overhead introduced by the compiler does not change the complexity classes for communication and computation. The number of signatures a user makes is equal to the number of messages he sends, the length of the message is extended by adding the message number and direct_U^i. The latter has length equal to the number of partners U directly communicates with and thus is reflected by the computation costs. Reading through the list as part of the signing process does not change the complexity. The number of signature verifications equals the number of processed messages. If user U has to verify that $\mathsf{Vrfy}_{PK_V'}(j|m|\mathsf{direct}_V^i, \sigma) = 1$ then V must be in rel_U and thus U knows V's position and thus also direct_V.

By sticking to the very same security definition as [12] we show that our compiler works just as well, and in particular allows having a better complexity if the underlying protocol does.

---

[3] Katz and Yung suggest to use $\mathsf{gid} = U_1|\ldots|U_n$ which automatically forces them to deal with inputs of length $O(n)$. This does not pose a problem for them because BD-I has complexity $O(n)$ anyway. In practical situations, however, the group is formed by e. g. a manager in a bank or a network administrator at the same time as deciding membership and the number of actually used groups is on a much smaller scale than the total number of all possible groups. Quite commonly there exists a compact description, e.g. the name of the task force.

[4] Our definitions of pid and direct ensure that all players V sending necessary input are actually present together with their initial randomness.

Theorem 2. *If P is a secure group KE protocol achieving forward secrecy, then P' given by the above compiler is a secure group AKE protocol achieving forward secrecy. Namely, for q_s the number of Send queries and q_{ex} the number of Execute queries we obtain*

$$\mathsf{Adv}_{P'}^{\mathsf{AKE-fs}}(t, q_{ex}, q_s) \leq \frac{q_s}{2} \cdot \mathsf{Adv}_P^{\mathsf{KE}}(t', 1) + \mathsf{Adv}_P^{\mathsf{KE}}(t', q_{ex}) + |\mathcal{P}| \cdot \mathsf{Succ}_\Sigma(t') + \frac{q_s^2 + q_{ex}q_s}{2^k},$$

where $t' = t + (|\mathcal{P}|q_{ex} + q_s) \cdot t_{P'}$ and $t_{P'}$ is the time required for an execution of P' by any party and Succ_Σ is the success probability against the used signature scheme Σ.

The compiler maintains the number of rounds and the complexity class of the protocol P.

Proof. The proof follows the same lines at the corresponding one in [12]. We use an active adversary $\mathcal{A}'$ against P' to construct an adversary against P. There are three ways in which $\mathcal{A}'$ can succeed, namely by forging a signature if he has not queried the Corrupt oracle before, by repeating the information direct and thus reusing a signature obtained in a previous execution, or by distinguishing the key from random. It is only the latter that leads to an attack against P.

The contribution of the event Forge is as in [12], namely $\Pr[\mathsf{Forge}] \leq |\mathcal{P}| \cdot \mathsf{Succ}_\Sigma(t')$.

Due to the different definition of direct compared to nonces in [12] we need to reconsider the probability of Repeat. The probability that the nonce used by any user in response to an initial Send query was previously used by that user (in either another execution of Send or in an Execute query) is still bounded by $\Pr[\mathsf{Repeat}] \leq \frac{q_s(q_s + q_{ex})}{2^k}$.

The remainder of the considerations works as in [12]. Let Ex be the event that $\mathcal{A}'$ queries the Test oracle to an instance Π_U^i such that $\mathcal{A}'$ never made a query of the form $\mathsf{Send}(U, i, *)$, i.e., in the simulation $\mathcal{A}$ has the full transcript without patches from its own execution of P. Defining $\mathsf{Se} = \overline{\mathsf{Ex}}$ and considering the probabilities $\Pr_{\mathcal{A}', P'}[\mathsf{Succ} \wedge \mathsf{Ex}]$ and $\Pr_{\mathcal{A}', P'}[\mathsf{Succ} \wedge \mathsf{Se}]$ separately by constructing appropriate adversaries $\mathcal{A}_1$ and $\mathcal{A}_2$ one obtains the stated result.   □

Corollary 1. *The authenticated group key exchange protocol obtained from Protocol 1 by applying this compiler is secure against active attacks and has communication and computation complexity $O(\log n)$.*

Remark 5. The original paper [4] describes BD-II in more generality allowing more than two children per vertex. For k children this means that each vertex has to compute k values for X_i whereas the final computation of the key reduces to $\log_k n$ computations only. While the overall number of operations is smallest for $k = 2$; larger k might be an interesting alternative if the storage is restricted and only $\log_k n$ elements can be stored from the multicast to compute K. In the sequential version (see Appendix) a larger k reduces the delay while making each step more expensive. The maximal overall efficiency is obtained for $k = 2$.

In an authenticated version, however, at most $k + \log_k n$ signatures need to be verified and $k + 1$ messages must be signed. Additionally $k + 2$ exponentiations

are needed. In the ElGamal signature scheme, two exponentiations are needed per signature generation and a multi-exponentiation is used for verification. Accordingly a larger k reduces the computational complexity.

6   Comparison

To show the advantages of our group AKE we give an overview of the costs for the naive version, the BD-I version considered in [12] and the two authenticated BD-II versions proposed in this paper, in Table 6. We also take into account the scheme by Boyd and Nieto [1] and the scheme by Bresson et al. [2]. In the Boyd-Nieto scheme one user U_1 chooses the key and encrypts this key for all $n-1$ other group members. This long message is signed and broadcast to all participants who then check the signature and decrypt their part to obtain the joint key. If a public key system is used for the encryption this scheme runs in 1 round. Otherwise a further round is necessary in which each party U_i (including U_1) sends g^{k_i}. Then the Diffie-Hellman key $g^{k_1 k_i}$ is used to transmit the group key. This scheme requires U_1 to perform $O(n)$ computations and the message has length $O(n)$. We denote the first version by BN PK and the second by BN DH.

In [2] the authors use $g^{k_1 k_2 k_3 \ldots k_n}$ as the joint secret key. To make this possible, the users sequentially add their contribution k_i to the key which accounts for requiring n rounds and the last user needs to send $g^{(k_1 k_2 k_3 \ldots k_n)/k_i}$ to user U_i for each i, so he has communication and computation $O(n)$.

Except for [1] and [2], GKE schemes are balanced in workload in that all users have about the same amount of work. The table below states the *maximal* values per user. Katz and Yung use the same measures because maximal effort per user captures scalability. We use **p** to denote point-to-point communication and **b** to denote broadcast. As the set of users is finite we do not distinguish between multicast and broadcast, otherwise the BD-II protocols use only multicast while BD-I needs broadcast. Furthermore, we list the maximal length of the messages.

For the computation S means signatures, V means verifying, E stands for full exponentiation and M for multiplication. For Boyd-Nieto PK we assume ElGamal encryption; the costs for retrieving the public keys are not mentioned. We neglect the number of inversions as there are at most 2 of them. For the number of rounds we consider the maximal delay of the protocol. Except for the sequential version of BD-II and [2] this is the maximum number of rounds per user. Accordingly, our authenticated BD-II version achieves an overall complexity of $O(\log n)$ while Katz-Yung [12] based on BD I need $O(n)$ in communication and in computation. Additionally, the number of messages to be stored from the broadcast is $O(\log n)$, as only the messages of the maximal $\log n$ ancestors are needed. The sequential version has a delay of $\log n$ but reduces the requirements considerably, *e.g.,* no broadcast is assumed and far fewer operations. This might be interesting in restricted networks which need to minimize the computational and technical requirements trading it off for a longer overall execution of the protocol.

Table 1. A comparison of the overhead costs of six AKE schemes

	Rounds	messages	communication	length	computation
naive	$n-1$	$1\mathbf{b}, 2\mathbf{p}$	$1\mathbf{b}, 2\mathbf{p}$	$O(1)$	$2S, 3V, 3E, 2M$
[2]	n	$2\mathbf{b}$	$(n-1)\mathbf{p}, 2\mathbf{b}$	$O(1)$	nS, nV, nE, nM
BN PK [1]	1	$1\mathbf{b}$	$1\mathbf{b}$	$O(n)$	$1S, nV, 2(n-1)E, (n-1)M$
BN DH [1]	2	$1\mathbf{p}, 1\mathbf{b}$	$(n-1)\mathbf{p}, 1\mathbf{b}$	$O(n)$	$2S, nV, nE, (n-1)M$
BD-I	3	$2\mathbf{p}, 1\mathbf{b}$	$4\mathbf{p}, n\mathbf{b}$	$O(1)$	$2S, nV, 3E, (2n-1)M$
BD-II	3	$3\mathbf{p}, 1\mathbf{b}$	$6\mathbf{p}, (\log_2 n)\mathbf{b}$	$O(1)$	$2S, (\log_2 n)V, 4E, (\log_2 n)M$
BD-II seq.	$(\log_2 n)$	$5\mathbf{p}$	$6\mathbf{p}$	$O(1)$	$3S, 4V, 4E, 2M$

7   Conclusions

We have presented a compiler that adds authenticity to any group KE protocols while preserving computation and communication complexities of the original group KE protocol. In particular we have detailed a secure group AKE protocol based on the BD-II group KE which has a constant number of rounds and requires only $O(\log n)$ computation and communication. This is, essentially, an exponential saving compared to previously proposed AKE protocols.

An open problem [17] is to propose a proven secure contributory group key distribution that is as efficient as BD-II. This is motivated by the pseudorandomness transitory problem. An attacker is allowed to have temporary control over some users source of randomness; this can be full control or only enough influence to deviate from uniformity.

References

1. Boyd, C., Nieto, J.M.G.: Round-optimal contributory conference key agreement. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 161–174. Springer, Heidelberg (2002)
2. Bresson, E., Chevassut, O., Pointcheval, D., Quisquater, J.-J.: Provably authenticated group Diffie-Hellman key exchange. In: Proc. 8th Annual ACM Conference on Computer and Communications Security, pp. 255–264 (2001)
3. Burmester, M., Desmedt, Y.: A secure and efficient conference key distribution system. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 275–286. Springer, Heidelberg (1995)
4. Burmester, M., Desmedt, Y.: Efficient and secure conference key distribution. In: Lomas, M. (ed.) Security Protocols. LNCS, vol. 1189, pp. 119–130. Springer, Heidelberg (1997)
5. Burmester, M., Desmedt, Y.: A secure and scalable group key exchange system. Information Processing Letters 94(3), 137–143 (2005)
6. Burmester, M., Desmedt, Y.: Identity-based Key Infrastructures (IKI). In: SEC 2004. Security and Protection in Information Processing Systems, pp. 167–176. Kluwer, Dordrecht (2004)
7. Desmedt, Y., Pieprzyk, J., Steinfeld, R., Wang, H.: A Non-Malleable Group Key Exchange Protocol Robust Against Active Insiders. In: Katsikas, S.K., Lopez, J., Backes, M., Gritzalis, S., Preneel, B. (eds.) ISC 2006. LNCS, vol. 4176, pp. 459–475. Springer, Heidelberg (2006)

8. Fischer, M.J., Lynch, N.A., Patterson, M.S.: Impossibility of distributed consensus with one faulty process. Journal of the ACM 32(2), 374–382 (1985)
9. Ingemarsson, I., Tang, D.T., Wong, C.W.: A conference key distribution system. IEEE Trans. Inform. Theory 28, 714–720 (1982)
10. Just, M., Vaudenay, S.: Authenticated multi-party key agreement. In: Kim, K., Matsumoto, T. (eds.) ASIACRYPT 1996. LNCS, vol. 1163, pp. 36–49. Springer, Heidelberg (1996)
11. Katz, J., Shin, J.S.: Modeling Insider Attacks on Group Key-Exchange Protocols. ePrint archive, 163/2005
12. Katz, J., Yung, M.: Scalable protocols for authenticated group key exchange. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 110–125. Springer, Heidelberg (2003), www.cs.umd.edu/~jkatz/research.html
13. Nam, J., Lee, Y., Won, D.: Constant Round Group Key Exchange with Logarithmic Computational Complexity. ePrint archive, 284/2006
14. Kim, Y., Perrig, A., Tsudik, G.: Tree-based group key agreement. ACM Trans. Inf. Syst. Secur. 7(1), 60–96 (2004)
15. Pieprzyk, J., Wang, H.: Malleability attacks on multi-party key agreement protocols. In: Coding, Cryptography and Combinatorics. Progress in Computer Science and Applied Logic, vol. 23, pp. 277–288 (2004)
16. Wong, C.K., Gouda, M.G., Lam, S.S.: Secure group communications using key graphs. In: SIGCOMM, pp. 68–79 (1998), IEEE/ACM Trans. Netw. 8(1), 16–30 (2000)
17. Yung, M.: Comment made during the Financial Cryptography 2007 presentation

Appendix – A Peer-to-peer Version of BD-II

The following gives a sequential version with only peer to peer communication and constant memory requirements having a $O(\log n)$ delay by changing slightly the protocol. Note that the number of rounds in which a party is actively involved remains unchanged but a user at a leaf has $\log n$ delay. Notation is like before.

Protocol 2 (Peer-to-peer version of BD-II group KE)

Step 1 *Each U_i, $i = 1, \ldots, n$, selects $k_i \in_R \mathbb{Z}/\ell\mathbb{Z}$, computes and sends $z_i = g^{k_i}$ to his parent and children.*

Step 2 *Each U_i, $i = 1, \ldots, n$ computes $X_{p,i} = z_{\text{parent}(i)}^{k_i}$, $X_{l,i} = z_{\text{left child}(i)}^{k_i}$ and $X_{r,i} = z_{\text{right child}(i)}^{k_i}$.*

Step 3 *U_1 and U_2 now have the joint key $K_i = X_{p,i}$ and send $Y_{\text{left child}(i)} = K_i \cdot X_{l,i}$ and $Y_{\text{right child}(i)} = K_i \cdot X_{r,i}$ to their respective children.*

Step 4 *For $j = 2, \ldots, m$ do:*
each user U_i on level j computes the joint key as $K_i = Y_i/X_{p,i}$ and sends $Y_{\text{left child}(i)} = K_i \cdot X_{l,i}$ and $Y_{\text{right child}(i)} = K_i \cdot X_{r,i}$ to his children.

Remark 6. For this scheme it is even more obvious why honest users obtain the same key $K = g^{k_1 k_2}$.

First note that this holds for K_1 and K_2. Assume that the parent of U_i obtained $K_{\text{parent}(i)} = K$. Then U_i computes $K_i = X_{p,i} Y_i = X_{p,i}(X_{p,\text{child}(\text{parent}(i))})^{-1} K_{\text{parent}(i)} = K$, where child takes into account the correct value of left or right so that $\text{child}(\text{parent}(i)) = i$.

On Authentication with HMAC and Non-random Properties

Christian Rechberger and Vincent Rijmen

Graz University of Technology
Institute for Applied Information Processing and Communications
Inffeldgasse 16a, A–8010 Graz, Austria
Christian.Rechberger@iaik.tugraz.at
www.iaik.tugraz.at/research/krypto/

Abstract. MAC algorithms can provide cryptographically secure authentication services. One of the most popular algorithms in commercial applications is HMAC based on the hash functions MD5 or SHA-1. In the light of new collision search methods for members of the MD4 family including SHA-1, the security of HMAC based on these hash functions is reconsidered.

We present a new method to recover both the inner- and the outer key used in HMAC when instantiated with a concrete hash function by observing text/MAC pairs. In addition to collisions, also other non-random properties of the hash function are used in this new attack. Among the examples of the proposed method, the first theoretical full key recovery attack on NMAC-MD5 is presented. Other examples are distinguishing, forgery and partial or full key recovery attacks on NMAC/HMAC-SHA-1 with a reduced number of steps (up to 61 out of 80). This information about the new, reduced security margin serves as an input to the selection of algorithms for authentication purposes.

1 Introduction

Authentication services can be provided in a cryptographically secure way by using Message Authentication Codes (MACs). The two most popular algorithms in commercial applications are variants of CBC-MAC based on 3-DES or AES, and HMAC based on MD5 and SHA-1. NMAC and HMAC [1] are message authentication codes based on a hash function. HMAC has been included in standards like ANSI, IETF, ISO, or FIPS. Commercial applications often use HMAC with SHA-1 as underlying hash function. After the recent collision attacks on MD5 [21] and reduced versions of SHA-1 [20], the impact of new collision attacks on the security of MAC constructions needs to be considered. This issue was already briefly mentioned in [23] and the impact on early hash based MAC constructions like the prefix-, suffix-, or envelope-method was informally discussed in [18]. Of particular interest is the impact on HMAC, since *e. g.* NIST supports HMAC-SHA-1 even after 2010 [14], whereas support for SHA-1 as a hash function will be dropped. In addition HMAC-SHA-1 is continued to be used in new designs and applications, *e. g.* in [13].

The security proof of NMAC, HMAC and other constructions is based on some assumptions about the pseudo-randomness of the underlying hash function, concluding that collision resistance is not needed. On the other hand, earlier work [6,9] suggests that collision attacks on the underlying hash function can be used to weaken the security of HMAC when instantiated with a popular hash function. In particular, distinguishing and forgery attacks, but as yet no full key recovery attacks have been shown.

After introducing some terminology and technicalities related to probabilities in Section 2, we present the two key points of this paper which are as follows.

1. Previous work on the security of NMAC and HMAC [6,9] against differential attacks is based on the reuse of characteristics that were constructed in order to mount collision attacks on the underlying hash function. This can lead to optimistic conclusions on the security margin of NMAC and HMAC. We propose a general framework for classifying the non-random properties of compression functions in Section 3. In addition to putting the existing characteristics for MD5, SHA-0 and SHA-1 into this framework, we devise new characteristics suitable for more efficient attacks than previously known on NMAC/HMAC instantiated with reduced variants of SHA-1.

2. The ability to recover the secret key by using known text/MAC pairs is certainly the most dangerous attack in practice. Currently known methods for NMAC/HMAC only allow to recover an inner key. This allows forgery attacks but does give an attacker the same possibilities as having the key (or equivalent information).

   We show a new key recovery attack which can recover the full key of NMAC (or an equivalent information in the case of HMAC), hence have for the first time the potential to use a substantially smaller amount of text/MAC pairs than black-box attacks. The details depend on the hash function being used and are discussed in Section 4. There we also give examples for full MD5 and reduced SHA-1.

We summarize and discuss the impact of our results and the security margins offered by HMAC when instantiated with popular hash functions in Section 5. Conclusions and open problems are given in Section 6.

2   Characteristics and Probabilities

2.1   Terminology from Differential Cryptanalysis

Differential cryptanalysis was originally invented to attack DES and other block ciphers [3]. The key concept behind a differential attack is the definition of a *characteristic.* Considering two inputs to the same cryptographic primitive, a characteristic is defined as the sequence of differences between the intermediate results occurring at corresponding times during the processing of these two inputs. The power of the method lies in the fact that it is possible to predict the differences of intermediate variables without specifying the actual input values.

For linear functions, the output difference is fully determined by the input difference. For nonlinear functions, it is possible to predict the output difference with a certain probability.

The *probability of a characteristic* is defined as the fraction of the input pairs that exhibits the differences of the characteristic. These input pairs are called the *right pairs*. In a differential attack, the cryptanalyst first tries to define a characteristic with a high probability. Subsequently, the cryptanalyst searches for one or more right pairs. The complexity of the search is related to the probability of the characteristic, but there are some fine points to consider. The most important characteristics are those it is the easiest to find a right pair for.

In order to be of use in a collision attack on a hash function, a characteristic needs to result in output difference zero. In key recovery attacks also other characteristics can be of use, provided that their probability is high enough.

In a *related-key differential attack*, also characteristics with differences in the key input of the cryptographic primitive are allowed. This often allows to construct characteristics with a higher probability, but the attack scenario becomes less realistic.

2.2   Easy Relations

One approach to estimate the complexity of the search for a message pair is to count the number of conditions, as is done by Kim *et al.* However, when the message is under full control of the attacker, optimizations are possible. It was already observed in the early analysis of SHA [5] that some conditions can be expressed as linear relations between the message bits. When considering only messages that satisfy these relations, the probability of a characteristic increases. In the remainder of this paper, we will always quote the increased probability of a characteristic, *i. e.* for messages that satisfy the easy relations. Subsequently we will use $\mathcal{M}$ to express this set of linear relations between message bits.

The increase is significant, as can be illustrated by considering the different step transformations in the compression function of SHA-1. Characteristics for SHA-1 are built up of disturbances and their corresponding local collisions. Their probability and hence their contribution to the data complexity of attacks devised later on in the paper depends on the bit position in the word and the steps they cover. The reason is that the 3-input Boolean function f being used in the step transformation of SHA-1 changes with every round (group of 20 steps). Table 1 illustrates the different cases. The column 'total' refers to the number of conditions in the case where no relations between message bits are assumed. For all steps and for all bit positions it is possible to *improve the results by fixing some relations between message bits* (shown in column 'reduced'). The number of linear relations between message bits is actually the difference between both numbers. Note that Table 1 is simplified in the sense that local collisions at the border between rounds are not considered. Also note that the position of local collisions relative to each other can either improve the overall probability or lead to impossible differentials. Examples of these effects are in detail discussed

Table 1. Number of conditions for a local collision. Note that the given figures only hold if the five steps after the disturbance are within the same round.

bit position	function	total	reduced
$0, 2, \ldots, 30$	f_{IF}	9	5
1	f_{IF}	7	5
31	f_{IF}	7	4
$0, 2, \ldots, 30$	f_{XOR}	6	4
1	f_{XOR}	3	2
31	f_{XOR}	4	3
$0, 2, \ldots, 30$	f_{MAJ}	9	4
1	f_{MAJ}	6	4
31	f_{MAJ}	7	4

in [11]. In the attacks presented in this paper, these effects on the probability of the characteristics are taken into consideration.

For example, we can increase the probability of the 34-step SHA-1 characteristic as presented in [9] from 2^{-52} to 2^{-31}.

2.3   Multiple Characteristics in One Differential

To obtain better estimates for the complexity of the search phase, we can add up the probabilities of all characteristics that contribute to the same differential. Mendel *et al.* derive an analytical formula taking into account the effect of multiple characteristics based on a study of carry effects [11]. As an example consider the forgery attack on 37-step HMAC-SHA-1 given in Table 4. By considering the better lower bounds using the methods described above, we expect the forgery attack to be successful already after 2^{66} instead of 2^{68} chosen messages.

3   New Characteristics

In this section, we first categorize the known characteristics over the compression function of hash functions. We classify the characteristics into 6 different types, depending on whether the differences in the inputs h_i, m_i and the output h_{i+1} are equal to zero or not. Table 2 presents an overview of the different types and concrete examples from the literature.

To illustrate the connection between this classification and traditional nomenclature [12], we give some examples. A 1-block collision attack on the hash function can be constructed by using a type 2 characteristic for the compression function. An n-block collision attack on a hash function can be constructed by using a type 3 characteristic on the first block, a type 6 characteristic on the last block, and type 7 characteristics on the $n-2$ remaining blocks. Of course we can't use just any set of such characteristics: the input difference in the chaining variable h of the characteristic in one block needs to match the output difference of the characteristic in the previous block. A 1-block pseudo-collision can be

Table 2. Types of characteristics. 'Y' indicates a non-zero difference, 'N' indicates 'no difference'.

Type	h_i	m_i	h_{i+1}	Examples from literature
2	N	Y	N	MD4 [19,23]; SHA-0, [5,22]
3	N	Y	Y	MD5 [21]; SHA-1 [2,20]
4	Y	N	N	MD5 [7]
5	Y	N	Y	reduced SHA-1 [10]
6	Y	Y	N	MD5 [21]; SHA-1 [2,20]
7	Y	Y	Y	

Table 3. Newly presented characteristics over the compression function of SHA-1. For details, we refer due to space restrictions to the full version of the paper.

Type	# steps	p_{char}	p_{diff}	details
2	37	2^{-66}	2^{-64}	[17]
3	50	2^{-72}	2^{-72}	[17]
2	53	2^{-98}	$3 \cdot 2^{-98}$	[17]
6	61	2^{-101}	2^{-99}	Table 6

constructed using a type 4 or type 6 characteristic. Similarly, we can use a type 5 or type 7 characteristic in the first block of an n-block pseudo-collision.

In the setting of NMAC/HMAC, many of the characteristics mentioned in Table 2 can not be used. On reason is that their probability is too low (*e. g.* type 6 characteristics for MD5 and SHA-1). Another reason is that for type 3 or type 7 characteristics to be useful additional restrictions on the message difference m_i need to be obeyed. Section 4.2 will cover this issue. Hence the known type 3 characteristics are ruled out as well. The remaining type 2 or type 4 characteristics can be used to draw some conclusions about the security margin offered by a particular hash functions when used in HMAC/NMAC. However, we argue that this gives too optimistic conclusions. We use SHA-1 as an example, where a 34-step characteristics is the longest useful characteristics in the literature on collision search. Table 3 gives an overview of new characteristics over the compression function of SHA-1. We developed efficient search algorithms to find them. They are based on methods developed in [15], with the improvement that exact probabilities as described in [4,11] instead of Hamming weights are used to prune and rank them. In Table 3, p_{char} gives the probability of the characteristic with the highest probability. Additionally, the probabilities p_{diff} include the improvements from considering also less-probable characteristics with the same input and output differences.

For a characteristic through the compression function of SHA-1 to be of use the probability needs to be significantly higher than 2^{-160}. The reason for including characteristics of the same type but with less steps is that some attacks require characteristics with probability higher than 2^{-80}. The new type 3 characteristic given in this paper obeys an additional restriction on the message difference as will be needed in Section 4.2. Note also that this is the only

characteristic where the characteristics starts at step 0. It is an open problem to find characteristics spanning more steps and include the first steps. Automated approaches to construct characteristics that consider non-linear effects in an efficient way [4] might serve as important building blocks.

4   New Key Recovery Method for NMAC

Let $h(iv, m)$ denote the application of an iterative hash function h on message input m and with the chaining variable initialized to iv. The NMAC construction can then be described as follows:

$$NMAC(k_1, k_2, m) = h(k_2, h(k_1, m)). \tag{1}$$

We call the key k_1 and the corresponding h the *inner key* and the *inner hash*. We call k_2 the *outer key* and the corresponding h the *outer hash*. Note that an attacker can act like having the secret key only if both the inner key and the outer key (or equivalent information) has been obtained. Hence recovering both the inner and the outer key constitutes a *full key recovery*.

Generally speaking, a differential key-recovery attack can work as follows.

off-line preparation phase: Define the characteristic(s).
on-line data collection phase: Obtain the MAC values for pairs of texts with as difference(s) the input difference(s) of the characteristic(s).
off-line data processing phase: When a pair of texts results in a pair of MAC values with as difference the output difference specified by the characteristic(s), assume that this is a right pair. For a right pair, we have information on the intermediate values of the algorithm. This information can be exploited to partially recover the key.

If the characteristic(s) specified a non-zero difference in the key, then the attack is a related-key attack.

4.1   Recovering the Inner Key

In [6] a related-inner key characteristic is used to recover the inner key of NMAC. After a right pair has been found, their attack proceeds by applying small changes to both messages of the right pair and checking whether the modified pair still results in colliding tags. We will refer to this method as *KR1*. Together with the new characteristics presented in Section 3, we subsequently illustrate that the security margin of HMAC-SHA-1 is less than previously thought.

Example for Reduced NMAC-SHA-1. As an example, consider NMAC-SHA-1 where the inner hash is reduced to 61 of its 80 rounds. The best previously published attack applies to NMAC based on SHA-1 reduced to 34 steps. For the recovery of the inner key, we use *KR1* and the new type 6 characteristic given in Table 6. We expect to query a related-key NMAC oracle with 2^{99} message pairs

in $\mathcal{M}$ as specified by the characteristic to find a message pair that result in the same MAC. Afterwards, both the effort to recover enough state bits with *KR1* as well as a brute force phase to determine the remaining bits of the inner key k_1 are negligible compared to the online phase. Hence the total complexity is 2^{100} which is significantly less than a 2^{160} black box attack to recover the inner key.

4.2   Recovering the Outer Key

Once the inner key has been recovered, also the outer key can be attacked. Different combinations of characteristics over the inner and outer hash function can be used. We list here some possibilities.

1. Type 4 or type 5 over the outer hash combined with the trivial characteristic (input and output differences equal to zero) over the inner hash. This is a related-outer key attack.
2. Type 2, 3, 6 or type 7 over the outer hash combined with type 3, type 5 or type 7 over the inner hash. This is a possibly related-outer key attack with possibly related-inner keys.

In the latter case, the difference in the message input of the characteristic over the outer hash needs to match the (padded) output difference of the characteristic over the inner hash. The right pairs for the inner hash characteristic can be produced off-line; the right pairs for the outer hash characteristic need on-line queries.

Recovery of the Outer Key of NMAC-MD5 with *KR1*. We describe here how the inner key recovery attack of [6] can be extended to a full key recovery attack. The same characteristic [7] as for the inner key recovery is used, which has probability 2^{-46}. Note that the conditions in the last 5 steps can be ignored safely, because all resulting differentials can be efficiently enumerated and the output differences can be directly observed. Hence, the sum of their probabilities can be lower bounded by 2^{-41}. Next, *KR1* is used to recover 25 bits of the internal state using $25 \times 2^{42} \approx 2^{47}$ queries. For each query the first word needs to be controlled, hence requires $25 \times 2^{42+32} \approx 2^{79}$ computations. Using this information, the full key can be guessed with $2^{128-25} = 2^{103}$ trials. Recovering more bits of the state does not make the attack more efficient since the offline cost to prepare more queries would be higher than the final key guessing. Since the first word is fixed only 3 words (96 degrees of freedom) are left because of the input padding of the outer hash. This is enough to generate the required 2^{42} queries per bit without additional overhead.

New Key Recovery Method *KR2*. We propose here an attack strategy *KR2* which can be used to recover first the inner key and subsequently also the outer key of NMAC. In some settings, *KR2* proves to be advantageous, which is

shown in an example with step-reduced NMAC-SHA-1, where a speed up factor 2^{30} compared to *KR1* is achieved.

Suppose we have a suitable characteristic over the outer hash function. Let the set of keys and the set of messages over which the characteristic has improved probability q be denoted by $\mathcal{K}$, respectively $\mathcal{M}$, *i. e.* these are the keys and messages satisfying the easy relations. Furthermore, we assume the probability over the set of messages in $\mathcal{M}$ and the set of keys *not* in $\mathcal{K}$ that a pair of messages produces a collision for the inner hash function to be 2^{-l}.

The attack works if $q \gg 2^{-l}$. If after collecting the MAC values for $2q^{-1}$ message pairs in $\mathcal{M}$ with the input difference specified by the characteristic we have observed at least one pair with equal tags, then we conclude that with high probability the key is in $\mathcal{K}$. Otherwise, we conclude that with high probability the key is not in $\mathcal{K}$.

Up to here *KR1* can be reformulated similarly, assuming relations between bits in the state can be efficiently mapped to relations between bits of the key. *KR1* continues to use the same characteristic and submits slightly modified messages in order to deduce more bits of internal state.

The *KR2* method instead uses a set of completely different characteristics and recovers a few key bits with each of them. One key advantage of *KR2* in the outer hash setting is that a factor 2^x for offline computation is saved by not having to fix the first x bits in the input message. It depends to the compression function, wether this outweighs the disadvantage of having less optimal characteristics in the set of characteristics needed for the attack. Another advantage of *KR2* over *KR1* are the available degrees of freedom: since no message word is fixed it is possible to generate a higher number of distinct queries. This is important in the outer hash setting since here at most l degrees of freedom are given due to the padding of its input.

A detailed description of *KR2* as well as the non-random properties needed for the compression function to make it more efficient than *KR1* are given in Appendix A.

Recovery of the Outer Key of Reduced NMAC-SHA-1 Using *KR2*. For HMAC-SHA-1 where the outer hash is reduced to 34 steps, the type 5 characteristic with probability 2^{-148} from [10] can be used. Using *KR2* as proposed in this paper, key recovery is faster than brute force trials. Using 2^{153} queries, we recover 4 bits of key information. Hence the overall cost when using *KR2* for key recovery is 2^{156} which is more than 2^{30} times faster in this setting than *KR1* and hence slightly faster than brute force search.

5   Applications and Implications

In the following, we outline how the new characteristics and the key recovery method can be used to analyze popular authentication methods like HMAC or a new proposal for making digital signatures using hash functions safer.

Table 4. Old and new results on attacks on NMAC/HMAC when used with SHA-1. Table entries either compare to the previous results with ours, or give new results for variants with more steps.

	steps	**forgery**	data	truncation	source
HMAC-SHA-1	34 (0-33)	forgery	2^{53}	64	[9]
HMAC-SHA-1	34 (0-33)	forgery	2^{34}	64	[6]
HMAC-SHA-1	34 (0-33)	forgery	2^{32}	64	this paper
HMAC-SHA-1	37 (20-56)	forgery	2^{65}	96	this paper
	steps	**distinguisher**	data	truncation	source
HMAC-SHA-1	43 $(00 - 42)$	rectangle d.	$2^{154.9}$	160	[9]
HMAC-SHA-1	50 $(00 - 49)$	rectangle d.	$2^{153.5}$	160	this paper
HMAC-SHA-1	53 $(20 - 72)$	differential d.	$2^{99.5}$	128	this paper
HMAC-SHA-1	61 $(19 - 79)$	related-key differential d.	2^{100}	128	this paper

5.1   HMAC

Distinguishing or forgery attacks on NMAC can easily be translated to attacks on HMAC. For key recovery attacks without requiring related keys, instead of the actual key information, equivalent information is obtained. Related-key attacks on NMAC as described in this article can not be translated into attacks on HMAC. Details on attacks exploiting the new method and characteristics developed in this article are given in Table 4 and Table 5. Table entries either compare to the previous results, or give new results for variants with more steps.

On Truncation. We also tackle the issue of truncation, which is (based on [16]) widely recommended and commonly done in practice. In column labeled 'truncation', we give in both tables a typical value (multiples of 32 bits) for truncating the output of the MAC algorithm until which the respective attack is not stopped. Note that this does not contradict the general statement of [16] that truncating helps against certain attacks but is a specific property of the newly devised attacks. In fact, if the output is further truncated than noted, the attack is stopped.

Forgeries. Forgeries for NMAC can be constructed using the same characteristics as for a collision, because a collision for the underlying hash function can be converted trivially into a forgery for NMAC. Naturally, one expects that constructing a forgery for NMAC is more difficult than constructing a collision for the underlying hash function, because now there is a secret key involved. This is correct. However, by sticking to the terminology of differential cryptanalysis when we discussed characteristics, we in fact neglected to take into account the absence of a secret key in a collision attack. Hence, the figures we have given are the ones that are relevant for a forgery attack. For a collision attack (of an unkeyed hash function), they are too pessimistic because if both the message and the chaining variables are known, then state variables can be influenced for many steps (more than 30 in case of SHA-1).

Table 5. Summary of key recovery attacks

	type	steps	data	offline	truncation	source
NMAC-MD5	inner rel. key	all (0-63)	2^{47}	2^{47}		[6]
HMAC-SHA-1	inner key	34 (0-33)	2^{34}	2^{34}		[6]
NMAC-MD5	full rel. key	all (0-63)	2^{47}	2^{103}	64	this paper
NMAC-SHA-1	full rel. key	34 (0-33)	2^{153}	2^{156}	160	this paper
HMAC-SHA-1	inner key	34 (0-33)	2^{32}	2^{32}	64	this paper
NMAC-SHA-1	inner rel. key	61 (19-79)	2^{100}	2^{100}	128	this paper
HMAC-SHA-1	inner key	53 (20-72)	$2^{99.5}$	$2^{99.5}$	128	this paper

Distinguishers. When we succeed in constructing a forgery faster than with the black-box attack, we have distinguished NMAC from a pseudo-random function. Hence, a forgery attack implies a distinguishing attack. However, if the goal is to distinguish NMAC/HMAC instantiated with a PRF from NMAC/HMAC instantiated with an actual hash function, the birthday bound does not apply [9]. Hence, as listed in Table 4, the new distinguishing attacks can cover more steps than the new forgery attacks.

Also, a distinguishing attack doesn't need to be based on a collision. Also near-collisions can be used to distinguish NMAC from a pseudo-random function as shown by Kim *et al.* which build rectangle distinguishers. As shown in table 4 we also improve on this attack by simply extending the used characteristic for 7 more steps and apply some of the improvements mentioned earlier in the article.

Key Recovery. In Table 5 we summarize the new attacks that recover the full key of NMAC when observing a number of text/MAC pairs. Both attacks require related outer keys. We also add attacks that recover only the inner key, while noting that this allows forgery attacks but does give an attacker the same possibilities as having the full key.

5.2   Randomized Hashing

The RMX mode of operation is proposed as a means to provide a safety net in applications relying on hash functions, by reducing the impact of collisions [8]. We briefly discuss the applicability of our results to this mode. Put in a simple way, a hash function used in RMX mode doesn't need to be collision resistant. Second preimage resistance, or *e-SPR* resistance, is sufficient. Hence we explain here how our characteristics can be used in a preimage attack.

For a second preimage attack on 53-step HMAC-SHA-1, we can reuse the characteristic presented in Appendix B. In a differential second preimage attack, only one pair can be tried for each characteristic. Furthermore, there is no distinction between easy relations and others. Hence the probability of the characteristic is reduced to $2^{-151.5}$, and this is also the probability of success of the attack.

Note that if the first preimage consists of t message blocks, we can try our characteristic once in each of the t blocks, and hence multiply the probability of

success by t. Our results imply that SHA-1 reduced to 53 steps is not as e-SPR resistant as an ideal compression function used in the proof of security of [8].

6 Conclusions and Future Work

We presented a thorough security evaluation of the heavily used authentication method HMAC when used with MD5 and SHA-1. Even though recent results on the collision resistance of the employed hash function triggered renewed interest in the security offered by HMAC when used with the affected hash functions, our results are more general. Using our newly developed key recovery method it turns out that in addition to collision attacks, also other non-random properties of the employed hash function can be used.

The results are the first full key recovery attack for NMAC-MD5 and a decreased security margin offered by HMAC-SHA-1. Most of the attacks work even if the output of the MAC is truncated, which is commonly done in practice. It is an open problem if automated methods that efficiently include non-linear effects while searching for useful characteristics as *e. g.* proposed in [4] can be used to improve on the attacks presented in this article.

Despite the progress being made, message authentication algorithms like HMAC are less susceptible to problems in the underlying hash function than the standalone hash function. However, it seems prudent to evaluate other hash functions like RIPEMD-160 or members of the SHA-2 family as well as new hash function proposals against the framework of undesired properties as shown in this paper.

Acknowledgements

References

1. Bellare, M., Canetti, R., Krawczyk, H.: Keying Hash Functions for Message Authentication. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 1–15. Springer, Heidelberg (1996)
2. Biham, E., Chen, R., Joux, A., Carribault, P., Lemuet, C., Jalby, W.: Collisions of SHA-0 and Reduced SHA-1. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 36–57. Springer, Heidelberg (2005)
3. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. Journal of Cryptology 4(1), 3–72 (1991)

---

[1] The information in this paper is provided as is, and no guarantee or warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

4. De Cannière, C., Rechberger, C.: Finding SHA-1 Characteristics. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 1–20. Springer, Heidelberg (2006)
5. Chabaud, F., Joux, A.: Differential Collisions in SHA-0. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 56–71. Springer, Heidelberg (1998)
6. Contini, S., Yin, Y.L.: Forgery and Partial Key-Recovery Attacks on HMAC and NMAC Using Hash Collisions. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 37–53. Springer, Heidelberg (2006)
7. den Boer, B., Bosselaers, A.: Collisions for the Compression Function of MD5. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 293–304. Springer, Heidelberg (1994)
8. Halevi, S., Krawczyk, H.: Strengthening Digital Signatures via Randomized Hashing. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 41–59. Springer, Heidelberg (2006)
9. Kim, J., Biryukov, A., Preneel, B., Hong, S.: On the Security of HMAC and NMAC Based on HAVAL, MD4, MD5, SHA-0 and SHA-1 (extended Abstract). In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 242–256. Springer, Heidelberg (2006)
10. Lu, J., Kim, J., Keller, N., Dunkelman, O.: Differential and Rectangle Attacks on Reduced-Round SHACAL-1. In: Barua, R., Lange, T. (eds.) INDOCRYPT 2006. LNCS, vol. 4329, pp. 17–31. Springer, Heidelberg (2006)
11. Mendel, F., Pramstaller, N., Rechberger, C., Rijmen, V.: The Impact of Carries on the Complexity of Collision Attacks on SHA-1. In: Robshaw, M. (ed.) FSE 2006. LNCS, vol. 4047, pp. 15–17. Springer, Heidelberg (2006)
12. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press, Boca Raton, USA (1997)
http://www.cacr.math.uwaterloo.ca/hac/
13. M'Raihi, D., Bellare, M., Hoornaert, F., Naccache, D., Ranen, O.: HOTP: An HMAC-based One Time Password Algorithm. Informational RFC 4226 (December 2005)
14. National Institute of Standards and Technology: NIST's Policy on Hash Functions (2006),
http://www.csrc.nist.gov/pki/HashWorkshop/NISTon_HashFunctions.htm
15. Pramstaller, N., Rechberger, C., Rijmen, V.: Exploiting Coding Theory for Collision Attacks on SHA-1. In: Smart, N.P. (ed.) Cryptography and Coding. LNCS, vol. 3796, pp. 78–95. Springer, Heidelberg (2005)
16. Preneel, B., van Oorschot, P.C.: MDx-MAC and Building Fast MACs from Hash Functions. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 1–14. Springer, Heidelberg (1995)
17. Rechberger, C., Rijmen, V.: On Authentication with HMAC and Non-Random Properties. Cryptology ePrint Archive, Report 2006/342 (2006),
http://eprint.iacr.org/
18. Wang, X.: What's the Potential Danger Behind the collisions of Hash Functions. In: ECRYPT Conference on Hash Functions, Krakow (2005),
http://ecrypt.eu.org/stvl/hfw/
19. Wang, X., Lai, X., Feng, D., Chen, H., Yu, X.: Cryptanalysis of the Hash Functions MD4 and RIPEMD. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 1–18. Springer, Heidelberg (2005)
20. Wang, X., Yin, Y., Yu, H.: Finding Collisions in the Full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer, Heidelberg (2005)

21. Wang, X., Yu, H.: How to Break MD5 and Other Hash Functions. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)
22. Wang, X., Yu, H., Yin, Y.L.: Efficient Collision Search Attacks on SHA-0. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 1–16. Springer, Heidelberg (2005)
23. Yu, H., Wang, G., Zhang, G., Wang, X.: The Second-Preimage Attack on MD4. In: Desmedt, Y.G., Wang, H., Mu, Y., Li, Y. (eds.) CANS 2005. LNCS, vol. 3810, pp. 1–12. Springer, Heidelberg (2005)

A   Details on the New Key Recovery Method *KR2*

The proposed key recovery attack consists of two phases:

1. Online phase: in a chosen-message scenario, the attacker asks for b $(m, \text{NMAC}(m))$ pairs under the same unknown key of length $2l$. Analyzing the results, c linear relations between bits of k are deduced.
2. Offline phase: The rest of the key is guessed in a brute force manner.

The attack is more efficient than brute force, if $2b + 2^{l-c}$ is smaller than 2^l. Subsequently the online phase is described in more detail. Before that, some definitions are needed. Note that the attack applies to HMAC in exactly the same way, expect that instead of the key information, equivalent information is obtained.

Let $\mathcal{K}$ be a set of linear relations between bits in k, and let $p_{\mathcal{K}}$ be the probability that a k picked from a uniform distribution satisfies these linear relations. Likewise, let $\mathcal{M}$ be a set of linear relations in m.

Let q be the probability that there is a collision at the output of the first application of h if m and $m + \alpha$ are input under the assumption that the unknown k satisfies $\mathcal{K}$ and m satisfies $\mathcal{M}$. We write

$$q = \Pr(h(k,m) + h(k, m + \alpha) = 0 \mid k \in \mathcal{K}, \mathtt{m} \in \mathcal{M}). \tag{2}$$

Note that the probability to observe a colliding MAC is higher, namely $q + (1 - q) \cdot 2^{-l}$. Likewise we define q' as the probability for the case that k does not satisfy the relations given by $\mathcal{K}$.

$$q' = \Pr(h(k,m) + h(k, m + \alpha) = 0 \mid k \notin \mathcal{K}, \mathtt{m} \in \mathcal{M}). \tag{3}$$

1. Collect b MAC pairs under the unknown key with chosen messages m and $m + \alpha$ where $m \in \mathcal{M}$.
2. If we observe at least one colliding MAC pair, then $k \in \mathcal{K}$ with probability $1 - \epsilon_1$. If we do not observe a colliding MAC pair, then $k \notin \mathcal{K}$ with probability $1 - \epsilon_2$.
3. Note that ϵ_1 is small if q' is small and 2^{-l} is negligible. ϵ_2 can be derived by the approach described in [9, Section 6]. ϵ_2 can be made sufficiently small by choosing a high enough b. $b = 2 \cdot q^{-1}$ is enough for practical purposes. For the attack it is important that $q \gg q'$ to ensure a small ϵ_1. Note that given a sufficiently small ϵ_2, ϵ_1 can be estimated to be q'/q. For simplicity

we subsequently assume both ϵ_1 and ϵ_2 to be zero. Details can be found in Appendix A.1.

4. If $k \in \mathcal{K}$, the possible key space is reduced by a factor $p_{\mathcal{K}}^{-1}$. If $k \notin \mathcal{K}$, the possible key space is reduced by a factor $(1 - p_{\mathcal{K}})^{-1}$. For a given $p_{\mathcal{K}}$ the reduction of key entropy is hence

$$p_{\mathcal{K}} \cdot log_2(p_{\mathcal{K}}) + (1 - p_{\mathcal{K}}) \cdot log_2(1 - p_{\mathcal{K}}) \tag{4}$$

bits. Thus the expected reduction in key entropy in this step is at most one bit.

The above described key entropy reduction technique can be applied for any number c of triples $(\alpha_i, \mathcal{K}_{\rangle}, \mathcal{M}_{\rangle})$. To optimize the computational complexity of recovering the full key we choose c such that $2^{l-c} > 2 \cdot \sum_{i=1}^{c} q_i^{-1}$. Note that this assumes the relations between bits in k (i.e. $\mathcal{K}$) to be linearly independent.

It remains to be described how to find triples $(\alpha_i, \mathcal{K}_{\rangle}, \mathcal{M}_{\rangle})$ for specific cryptographic hash functions. One way to derive new characteristics for hash functions of the MD4 family like MD5 or SHA-1 is to simply rotate each of the 32-bit words of the inputs of a known characteristic over the same number of bit positions. This follows from two facts. Firstly the linear code describing the message expansion is invariant with respect to word rotation. Secondly, the used characteristic usually requires that there is no carry propagation in the modular addition. This condition has always a probability > 0, although rotation might increase or decrease it. Note that there are special cases of characteristics where this technique does not work for all rotation values.

A.1 ϵ_1

ϵ_1 can be derived as follows:

$$\Pr(\mathtt{k} \in \mathcal{K} \mid \mathtt{collision}) = \frac{\Pr(\mathtt{collision} \mid \mathtt{k} \in \mathcal{K}) \Pr(\mathtt{k} \in \mathcal{K})}{\Pr(\mathtt{collision})}$$

$$= \frac{(q + (1-q)2^{-l})p_{\mathcal{K}}}{(q + (1-q)2^{-l})p_{\mathcal{K}} + (q' + (1-q')2^{-l})(1 - p_{\mathcal{K}})}$$

$$= \frac{1}{1 + \frac{(q' + (1-q')2^{-l})(1 - p_{\mathcal{K}})}{(q + (1-q)2^{-l})p_{\mathcal{K}}}}$$

$$\approx 1 - \frac{(q' + (1-q')2^{-l})(1 - p_{\mathcal{K}})}{(q + (1-q)2^{-l})p_{\mathcal{K}}}$$

A.2   Extension of *KR2* to the Outer Hash Setting

Determine a characteristic (h'_{in}, m', h'_{out}) over the outer hash with probability p_2. Probability p_2 for this char needs to be better than 2^{-160+p_1}. We distinguish between two cases

1. $m' \neq 0$. Recover the inner key with complexity 2^{p_1}.
   Determine a characteristic over the inner hash that produced the output difference that after padding will produce the m' from above. Since the inner

key is known at this stage, we can take this into account when constructing the characteristic and during the search for right pairs. Say that the complexity to find a right pair is 2^{p_3}.

We need 2^{p_2} of these near-collisions. Here the offline cost is $2^{p_2+p_3}$. The number of chosen texts is $2^{p_1} + 2^{p_2}$. This allows to recover a certain number of relations between bit of the outer key and potentially can be repeated for some more bits.

2. $m' = 0$. Note that this implies $h'_{in} \neq 0$: as above but $p_1 = 0$ because we don't need collisions but just single messages. Note that the example for 34-step NMAC-SHA-1 in Section 4.2 is of that type.

B Characteristics

For the characteristics, we adopt the notation introduced in [4]. Here we briefly restate the relevant parts. $'x'$ denotes XOR difference of unknown sign, $'n'$ and $'u'$ denote differences of known sign, $'-'$ refers to no difference and $'1'$ and $'0'$ refer to a setting where not only there is no difference, but also the actual value for the bit is fixed. Column A_i shows the state variables and W_i the expanded message words. The values in the column $P_u(i)$ denote $-log_2(p_u(i))$, where $p_u(i)$ is the uncontrolled probability as defined in [4]. Due to space restrictions, only one characteristic is given. For all other characteristics we refer to the full version of the paper [17].

Table 6. Type 6 characteristic with probability 2^{-101} used for the 61-step (19-79) attack

i	∇A_i	∇W_i	$P_u(i)$
-4	------------------------------		
-3	---------------------------u-		
-2	------------------------------		
-1	0-----------------------------		
0	------------------------------	u-1----------------------------	0
1	------------------------------	n1-----------------------------	0
2	------------------------------	-1-----------------------------	0
3	------------------------------	10-----------------------------	0
4	------------------------------	1------------------------------	0
5	------------------------------	1u0----------------------------	1
6	-u----------------------------	0-0-----------------------n---	0
7	------------------------------	0n1-----------------------n--	2
8	--------------------------n-	u--n--------------------u-----	2
9	n-----------------------------	0n0u------------------u---u	4
10	-n--------------------------n	1-1n----------------un-u--0	0
11	------------------------------	nuu------------------------n0	4
12	---------------------------n	uunu-----------------u---u-	5
13	n--------------------------u-	0nuu----------------n-u--u	4
14	u-----------------------------	101u----------------------n----	3
15	--------------------------n-	1nu-----------------u----n	3
16	---------------------------n	uu----------------------u--n-	1
-	...	...	-
57	------------------------------	------------------------1-	2
58	--------------------------n-	u----------------------u------	0
59	------------------------------	u---------------------0-n-	1
60	------------------------------	------------------------1---	0
61	------------------------------		

Hidden Identity-Based Signatures

Aggelos Kiayias* and Hong-Sheng Zhou*

Computer Science and Engineering
University of Connecticut
Storrs, CT, USA
{aggelos, hszhou}@cse.uconn.edu

Abstract. This paper introduces Hidden Identity-based Signatures (Hidden-IBS), a type of digital signatures that provide mediated signer-anonymity on top of Shamir's Identity-based signatures. The motivation of our new signature primitive is to resolve an important issue with the kind of anonymity offered by "group signatures" where it is required that either the group membership list is *public* or that the opening authority is *dependent* on the group manager for its operation. Contrary to this, Hidden-IBS do not require the maintenance of a group membership list and they enable an opening authority that is totally independent of the group manager. As we argue this makes Hidden-IBS much more attractive than group signatures for a number of applications. In this paper, we provide a formal model of Hidden-IBS as well as two efficient constructions that realize the new primitive. Our elliptic curve construction that is based on the SDH/DLDH assumptions produces signatures that are merely 4605 bits long and can be implemented very efficiently.

To demonstrate the power of the new primitive, we apply it to solve a problem of current onion-routing systems focusing on the Tor system in particular. Posting through Tor is currently blocked by sites such as Wikipedia due to the real concern that anonymous channels can be used to vandalize online content. By injecting a Hidden-IBS inside the header of an HTTP POST request and requiring the exit-policy of Tor to forward only properly signed POST requests, we demonstrate how sites like Wikipedia may allow anonymous posting while being ensured that the recovery of (say) the IP address of a vandal would be still possible through a dispute resolution system. Using our new Hidden-IBS primitive in this scenario allows to keep the listing of identities (e.g., IP addresses) of Tor users computationally hidden while maintaining an independent Opening Authority which would not have been possible with previous approaches.

1 Introduction

Anonymity and privacy is an issue of increasing concern in the Internet and the offering of services such as anonymous channels is an important aspect of the future Internet infrastructure if we want to retain fundamental rights such as free

---

speech. Still, anonymous systems are plagued by the potential of misuse and any system that permits strong anonymity seems to be doomed to be of limited use in one sense or another. To see this point consider the recent example of Tor [32], an onion-routing system, and how Tor traffic is currently handled by Wikipedia [35]. While Wikipedia allows HTTP "GET requests" from Tor, it does not allow editing (i.e., HTTP "POST requests") since allowing such requests opens the possibility to malicious users to vandalize the content of the web-site (actually the Wikipedia suggests to disable privacy in Tor in order to publish to the web-site through the onion-router, see [36]). For similar reasons, Tor's "exit policy" drops all SMTP packets (i.e., packets directed to port 25) to make sure that spammers do not take advantage of the anonymity offered by Tor.

The above two examples exemplify the fact that anonymous communication systems such as Tor *limit their scope* due to the potential of misuse. And it is conceivable that the increase of malicious activity trafficking through anonymous communication networks (that includes the distribution of child pornography for example) will force such networks to become even more restricted in scope something that in turn will nullify the purpose they were built originally (to protect free speech and enable anonymous communication for legal uses).

Misusing anonymity is by no means a new idea: for example the work of [33] shows how anonymous e-cash can be used to commit a perfect crime. For this reason primitives such as fair off-line cash [14,21] were proposed where it is possible for an authority to manage anonymity and reveal the identities of the entities behind a certain transaction given that certain conditions are satisfied. It should be stressed that the existence of such "anonymity mediation" systems are not restricting anonymity but rather *enhance it* since they make it possible to employ anonymous systems in cases where no such system may be allowed to exist (due to regulation and potential of misuse etc.).

Group signatures, introduced in [18], and further studied in a number of works [19,10,17,15,16,2,12,4,3,26,7,11,23,13,30,8,5,27,28,22,9,1] constitute a tool that can be used to offer such mediated anonymity. Indeed, in a group signature it is possible for users to join the group and obtain a credential from the group manager (GM); subsequently, users can issue signatures that a verifier can identify as signatures originating from a group member but she cannot tell which member is issuing the signature. At the same time an opening authority (OA) is capable, given an "offending" signature, to recover a piece of information that leads to the identity of the signer.

However, as we notice in this work, if one tries to employ group signatures to mediate anonymity in an anonymous credential system, a fundamental problem arises:

The Anonymity Catch-22 of Group Signatures. In Heller's novel [25] Catch-22 refers to a no-win situation; a certain setting where no matter what you do you lose. Here we argue that a similar "Catch-22" scenario occurs when one applies group signatures to mediate anonymity in an anonymous credential system.

To see the problem consider the following sequence of objectives: our primary goal is to (*i*) maximize anonymity and its scope; now given that perfect

anonymity would be of limited scope, this implies that we need to: (*ii*) employ an opening authority; now, once the OA is allowed, one would want this entity to be managed properly and thus this brings forth: (*iii*) the OA should be separated from the GM (the registration service) and preferably be a "threshold entity" where many share-holders should be allowed to participate equally in the decision-making process of opening an offending signature.

Now recall the following: in *all* group signature schemes the OA is incapable of recovering the identity of the signer without comparing the information recovered from the signature to a *name directory* (essentially a group membership database that acts as PKI) that is maintained by the GM (this is even true in recent "identity-based" group signature [34]). With respect to the membership directory thus, it should be that either (*iv*-1) the group member directory is public knowledge, or (*iv*-2) the group member directory is kept secret by the GM. But if (*iv*-1) is true, our objective (*i*) is violated: publishing the list of users that take advantage of an anonymous service in most cases would be the most serious privacy violation possible! (indeed publishing the list of users that use an anonymous service maybe enough to incriminate them if someone wishes their persecution). On the other hand, if (*iv*-2) is true, objective (*iii*) is violated since the OA cannot open an offending signature without the help of the GM. This means that the GM can effectively produce a *denial of service* to any entity that requires the assistance of the OA and thus the OA cannot really guarantee to a service provider that it can open an offending signature. This in turn leads to the OA being less credible and may lead to service providers restricting the use of the anonymous system something that in turn hurts anonymity. Thus no matter how one deploys group signatures, privacy is being reduced.

Resolving this "Anonymity Catch-22" issue of group signatures requires a new signature primitive that we introduce in this work:

Our Contribution: Hidden Identity-Based Signatures. In this work we propose a new digital signature scheme that offers anonymity that can be mediated and is based on the concept of Identity-based signatures (IBS) [31]. In a Hidden-IBS scheme, a signer obtains her signing key by communicating to an identity manager (IM) and negotiating her identity with IM. Given the secret-key the signer can produce signatures on a given message so that her identity is not revealed to the verifier. Still, the verifier is ensured of the fact that the identity negotiation has taken place between the signer and the IM and moreover that the signature *contains the name of the signer in enciphered form* and such name can be recovered by an opening authority.

Hidden-IBS resolve the Anonymity Catch-22 of group signatures since they allow the OA to recover the identity of the signer (i) without having to consult with the IM (which substitutes the GM in the Hidden-IBS setting) and (ii) without requiring the IM to publish a listing of users of the anonymous signatures. See figure 1.

We note that in a Hidden-IBS the identity of the signer may be equal to any piece of information that is considered acceptable under the policy of the
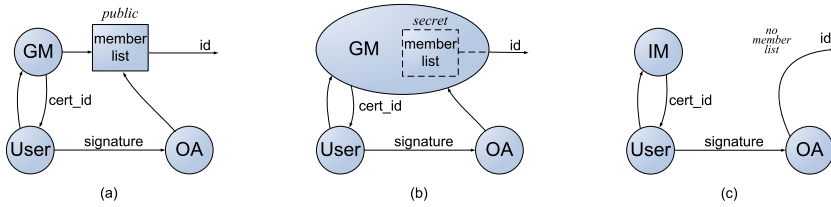
Fig. 1. Comparison of the opening functionality between group signatures and Hidden-IBS: (a) group signature with public group membership list, (b) group signature with secret group-membership list, (c) Hidden-IBS

IM, e.g., it can be the signer's e-mail address, the signer's IP address and so forth. Note that the IM and the signer may execute a multi-round protocol to establish the validity of the signer's identity (e.g., the IM may send a verification e-mail to the signer's e-mail account etc.).

In this work we present a formal model of Hidden-IBS, that captures two intuitive properties, misidentification-forgery and anonymity. We then present a construction over elliptic curve groups that is based on the Strong Diffie-Hellman assumption and the Decisional Linear Diffie-Hellman assumption that is merely 4605 bits long. In the full version [29] we also consider how the property of exculpability can also be achieved, and based on the Strong-RSA assumption and Decisional Composite Residuosity assumption, we present a construction which achieves security against a malicious IM. More discussions and all proofs are included in the full version.

Application to Onion Routing. We demonstrate how Hidden-IBS can be applied to onion-routing [24] and in particular to the Tor system [32,20] to allow mediation of anonymity and thus increase rather than limit the scope of such anonymous communication systems.

In Tor, each user transmits messages through a local Onion Proxy (OP) that allows a local host to build circuits hopping along a sequence of onion-routers (OR). Tor has an exit policy: packets that match certain conditions may be dropped (e.g., Tor drops packets directed to port 25 (SMTP) to prevent spammers from using Tor).

Our Hidden-IBS enhanced version of Tor is as follows: there will be certain types of traffic that the exit-policy of Tor will require to be signed with a Hidden-IBS. These may include HTTP POST requests directed to Wikipedia sites and traffic directed to the SMTP port (we stress that most of the traffic will be excluded from the requirement of being signed and thus the performance overhead of our extension would be low). Tor users that wish to use Tor for "sensitive traffic" will be given a list of IM's and explain the conditions of usage as well as they will be informed of the identity of the OA (which ideally will be a threshold entity). The IM's that will employed by Tor will require very little information from the users: in particular the identity of a signer can be simply *a verified e-mail address* or the signer's *IP-address* (of course the identity information required by the IM can be calibrated accordingly). The OP of a user will detect

that the user wishes to transmit something that according to the exit-policy must be signed and will redirect the user to obtain a Hidden-IBS secret-key that will allow the signing of the message. The Hidden-IBS will be injected into the packet itself (e.g., in the case of an HTTP POST we will use a special header field to contain the Hidden-IBS of the HTTP packet) and the signature will be verified by the Tor exit point. The ciphertext along with the necessary information to recover the identity of the user (if ever needed) will be posted in a public "Disputes&Grievances" database. The database will be designed in such a way so that it retains no publicly readable information about the identities of Tor users or the traffic they produce (only hashed packets and ciphertexts will be stored in the database). Still, the database will make it possible for any web-site that has received offending Tor traffic to submit a complaint to the OA that, if accepted, it will recover either the IP address or the e-mail of the culprit. Subsequently the identity may be blacklisted by the IM or receive negative points in a reputation system. Given that a Hidden-IBS signing credential would expire in short time periods the offender will have to face the outcome of his adverse behavior (lower reputation score with the IM, or being blacklisted etc.).

2   Hidden-IBS: Modelling

In this section, we give the definition of Hidden-IBS. The parties are involved in the scheme include the identity manager IM, the open authority OA, the users U, and the verifiers V,

Definition 1. *A* hidden identity-based signature *(Hidden-IBS) scheme is a digital signature scheme that consists of six polynomial-time algorithms* $\langle \mathtt{Setup}, \mathtt{Reg}, \mathtt{Sign}, \mathtt{RegCheck}, \mathtt{Verify}, \mathtt{Open} \rangle$. *The first three algorithms are probabilistic but the last three are not necessarily.*

$\mathtt{Setup}$: *The* $\mathtt{Setup}$ *algorithm includes* $\mathtt{SetupIM}$ *and* $\mathtt{SetupOA}$. *On input a security parameter, first the global system parameter is generated. Then on input a security parameter and the system parameter, the probabilistic algorithm* $\mathtt{SetupIM}$ *outputs the group verification key* pk_{IM} *and the signing key* sk_{IM} *for the identity manager, the probabilistic algorithm* $\mathtt{SetupOA}$ *outputs the public key* pk_{OA} *and the secret key* sk_{OA} *for the open authority. The* $\mathtt{Setup}$ *algorithm may include* $\mathtt{SetupUser}$; *on the input a security parameter and the system parameter, outputs* $\mathtt{id}$ *for both the identity manager and the user.*

$\mathtt{Reg}$: *A probabilistic algorithm that given an identity manager's verification key, an identity manager's signing key, a user's identity* $\mathtt{id}$ *outputs a membership certificate* $\mathtt{cert_{id}}$ *for the identity* $\mathtt{id}$. *We write* $\mathtt{Reg}(pk_{\mathsf{IM}}; sk_{\mathsf{IM}}, \mathtt{id})$ *to denote the registration algorithm.*

$\mathtt{RegCheck}$: *An algorithm for user's checking the validity of the certificate for her identity with respect to an identity manager's public key. We denote the application of the registration checking algorithm as* $\mathtt{RegCheck}(pk_{\mathsf{IM}}; \mathtt{id}, \mathtt{cert_{id}}) \in \{0, 1\}$.

Sign: *A probabilistic algorithm that given an identity manager's public key, an open authority's public key, a user's identity, a membership certificate on the user's identity, and a message m, outputs a signature for the message m. We write $\text{Sign}(pk_{\text{IM}}, pk_{\text{OA}}, \text{id}; \text{cert}_{\text{id}}, m)$ to denote the application of the signing algorithm.*

Verify: *An algorithm for establishing the validity of an alleged Hidden-IBS signature of a message with respect to an identity manager's verification key and an open authority's public key. If σ is a signature on a message m, then we have $\text{Verify}(pk_{\text{IM}}, pk_{\text{OA}}; m, \sigma) \in \{0, 1\}$.*

Open: *An algorithm that given a message, a valid Hidden-IBS signature on it, an identity manager's verification key, an open authority's public key, and an open authority secret key, determines the id directly. In particular $\text{id} \leftarrow \text{Open}(pk_{\text{IM}}, pk_{\text{OA}}; sk_{\text{OA}}, m, \sigma)$.*

Definition 2 (Correctness). *The correctness of the Hidden-IBS include the registration correctness, the signing correctness, and the opening correctness.*
Registration Correctness *means that the IM issues only one valid membership certificate for each different* id, *which is defined as below,*

$$\Pr \left[\begin{array}{l} (pk_{\text{IM}}, sk_{\text{IM}}) \leftarrow \text{SetupIM}(1^\lambda); (pk_{\text{OA}}, sk_{\text{OA}}) \leftarrow \text{SetupOA}(1^\lambda); \\ \text{cert}_{\text{id}} \leftarrow \text{Reg}(pk_{\text{IM}}, pk_{\text{OA}}; sk_{\text{IM}}, \text{id}); \\ \qquad : \text{RegCheck}(pk_{\text{IM}}, pk_{\text{OA}}; \text{id}, \text{cert}_{\text{id}}) = 1 \end{array} \right] = 1$$

Signing Correctness *ensures that the correctness of the underlying signing and verification algorithms for any valid signing key.*

$$\Pr \left[\begin{array}{l} (pk_{\text{IM}}, sk_{\text{IM}}) \leftarrow \text{SetupIM}(1^\lambda); (pk_{\text{OA}}, sk_{\text{OA}}) \leftarrow \text{SetupOA}(1^\lambda); \\ \text{cert}_{\text{id}} \leftarrow \text{Reg}(pk_{\text{IM}}, pk_{\text{OA}}; sk_{\text{IM}}, \text{id}); \text{RegCheck}(pk_{\text{IM}}, pk_{\text{OA}}; \text{id}, \text{cert}_{\text{id}}) = 1; \\ \sigma \leftarrow \text{Sign}(pk_{\text{IM}}, pk_{\text{OA}}, \text{id}; \text{cert}_{\text{id}}, m) \\ \qquad : \text{Verify}(pk_{\text{IM}}, pk_{\text{OA}}; m, \sigma) = 1 \end{array} \right] = 1$$

Opening Correctness *ensures that the* Open *algorithm can correctly identifies all signers from a valid signature, which is defined as below,*

$$\Pr \left[\begin{array}{l} (pk_{\text{IM}}, sk_{\text{IM}}) \leftarrow \text{SetupIM}(1^\lambda); (pk_{\text{OA}}, sk_{\text{OA}}) \leftarrow \text{SetupOA}(1^\lambda); \\ \text{cert}_{\text{id}} \leftarrow \text{Reg}(pk_{\text{IM}}, pk_{\text{OA}}; sk_{\text{IM}}, \text{id}); \text{RegCheck}(pk_{\text{IM}}, pk_{\text{OA}}; \text{id}, \text{cert}_{\text{id}}) = 1; \\ \sigma \leftarrow \text{Sign}(pk_{\text{IM}}, pk_{\text{OA}}, \text{id}; \text{cert}_{\text{id}}, m); \text{Verify}(pk_{\text{IM}}, pk_{\text{OA}}; m, \sigma) = 1 \\ \qquad : \text{Open}(pk_{\text{IM}}, pk_{\text{OA}}; sk_{\text{OA}}, m, \sigma) = \text{id} \end{array} \right] = 1$$

Definition 3 (Misidentification-Forgery). *We say a Hidden-IBS scheme is against misidentification-forgery attacks if for any PPT adversary $\mathcal{A}$, $\mathbf{Adv}_{\mathcal{A}}^{\text{misid}}(\lambda)$ is negligible in λ, where $\mathbf{Adv}_{\mathcal{A}}^{\text{misid}}(\lambda) = \Pr[\mathbf{Exp}_{\mathcal{A}}^{\text{misid}}(\lambda) = 1]$, where the experiment defined as in figure 2.*

Definition 4 (CCA2-Anonymity). *We say a Hidden-IBS scheme is against anonymity attacks if for any PPT adversary $\mathcal{A}$, $\mathbf{Adv}_{\mathcal{A}}^{\text{cca2}-\text{anon}}(\lambda)$ is negligible in λ, where $\mathbf{Adv}_{\mathcal{A}}^{\text{cca2}-\text{anon}}(\lambda) = \Pr[\mathbf{Exp}_{\mathcal{A}}^{\text{cca2}-\text{anon},1}(\lambda) = 1] - \Pr[\mathbf{Exp}_{\mathcal{A}}^{\text{cca2}-\text{anon},0}(\lambda) = 1]$, where the experiment defined as in figure 3.*

RegOracle(id)
    If $\mathrm{id} \in CU \cup HU$ then return $\bot$;
    $\mathrm{cert_{id}} \leftarrow \mathrm{Reg}(pk_{IM}, pk_{OA}; sk_{IM}, \mathrm{id})$;
    $MSG_{id} \leftarrow \emptyset; HU \leftarrow HU \cup \{\mathrm{id}\}$;
    Return 1;
	SignOracle(id, $\mathrm{cert_{id}}, m$)
	    If $\mathrm{id} \notin HU$ then return $\bot$;
	    $\sigma \leftarrow \mathrm{Sign}(pk_{IM}, pk_{OA}, \mathrm{id}; \mathrm{cert_{id}}, m)$;
	    $MSG_{id} \leftarrow MSG_{id} \cup \{m\}$;
	    Return σ;

CorruptUOracle(id)
    If $\mathrm{id} \notin HU$ then return $\bot$;
    $CU \leftarrow CU \cup \{\mathrm{id}\}; HU \leftarrow HU \backslash \{\mathrm{id}\}$
    Return $\mathrm{cert_{id}}$;
	CorruptOAOracle()
	    Return sk_{OA};

Experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathrm{misid}}(\lambda)$
    $(pk_{IM}, sk_{IM}) \leftarrow \mathrm{SetupIM}(1^\lambda); (pk_{OA}, sk_{OA}) \leftarrow \mathrm{SetupOA}(1^\lambda); HU \leftarrow \emptyset; CU \leftarrow \emptyset$
    $(m, \sigma) \leftarrow \mathcal{A}^{\mathrm{RegOracle}(), \mathrm{SignOracle}(), \mathrm{CorruptUOracle}(), \mathrm{CorruptOAOracle}()}(1^\lambda, pk_{IM}, pk_{OA})$
    If $\mathrm{Verify}(pk_{IM}, pk_{OA}; m, \sigma) = 1 \wedge \mathrm{Open}(pk_{IM}, pk_{OA}; sk_{OA}, m, \sigma) = \bot$ then return 1;
    If $\mathrm{Verify}(pk_{IM}, pk_{OA}; m, \sigma) = 1 \wedge \mathrm{Open}(pk_{IM}, pk_{OA}; sk_{OA}, m, \sigma) = \mathrm{id}$
        $\wedge m \notin MSG_{id} \wedge \mathrm{id} \notin CU$ then return 1;
    Return 0;

Fig. 2. Experiment of misidentification-forgery

Definition 5 (CPA-Anonymity). *We say a Hidden-IBS scheme is against CPA-anonymity attacks if for any PPT adversary $\mathcal{A}$, $\mathbf{Adv}_{\mathcal{A}}^{\mathrm{cpa-anon}}(\lambda)$ is negligible in λ, where $\mathbf{Adv}_{\mathcal{A}}^{\mathrm{cpa-anon}}(\lambda) = \Pr[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{cpa-anon},1}(\lambda) = 1] - \Pr[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{cpa-anon},0}(\lambda) = 1]$, where the experiment defined as in figure 4.*

3   Hidden-IBS: Construction

In this section we describe our first Hidden-IBS construction. It is geared towards producing short signatures and is suitable for relatively short identity strings (e.g., IP addresses of 32 bits). We let the IM use the Boneh-Boyen [6] signature to issue a certificate to each user identity. Once a user obtains the certificate from the IM, she can generate a Hidden-IBS signature for a message: the user uses Linear encryption [7] to "embed" her identity which can be opened by the OA; the user forms the signature based on a proof of knowledge that ensures her identity, her certificate, and the relations between them are properly formed. We present the details below:

Setup. This procedure first generates the system parameters including the bilinear group parameter $\langle p, g, \widehat{g}, \mathbb{G}, \widehat{\mathbb{G}}, \psi, \mathbb{G}_T, e \rangle$, random element $\widehat{h} \xleftarrow{\mathrm{r}} \widehat{\mathbb{G}} \backslash \{1\}$ and $h = \psi(\widehat{h})$, and a hash function $\mathcal{H}: \{0,1\}^* \rightarrow \mathbb{Z}_p$ which will be treated as a random oracle in the security proof. Then the algorithm $\mathrm{SetupIM}$ generates key pair (pk_{IM}, sk_{IM}): selects $x, y \xleftarrow{\mathrm{r}} \mathbb{Z}_p^*$ and compute $\widehat{X} = \widehat{g}^x$ and $\widehat{Y} = \widehat{g}^y$; sets $pk_{IM} = \langle \widehat{X}, \widehat{Y} \rangle$, and $sk_{IM} = \langle x, y \rangle$. The algorithm $\mathrm{SetupOA}$ generates key pair (pk_{OA}, sk_{OA}): selects $\widehat{w} \xleftarrow{\mathrm{r}} \widehat{\mathbb{G}} \backslash \{1\}$, selects $\delta, \xi \xleftarrow{\mathrm{r}} \mathbb{Z}_p^*$ and sets $\widehat{u}, \widehat{v} \in \widehat{\mathbb{G}}$ such that $\widehat{u}^\zeta = \widehat{v}^\eta = \widehat{w}$; sets $w = \psi(\widehat{w})$, $u = \psi(\widehat{u})$, $v = \psi(\widehat{v})$; note that $u^\zeta = v^\eta = w$ holds; sets $pk_{OA} = \langle u, v, w, \widehat{u}, \widehat{v}, \widehat{w} \rangle$ and $sk_{OA} = \langle \zeta, \eta \rangle$. Finally sets the public parameters

OpenOracle(m, σ)

    If $\texttt{Verify}(pk_{\mathsf{IM}}, pk_{\mathsf{OA}}; m, \sigma) = 1$

        then return $\texttt{Open}(pk_{\mathsf{IM}}, pk_{\mathsf{OA}}; sk_{\mathsf{OA}}, m, \sigma)$

    Return $\perp$;

CorruptIMOracle$()$

    Return sk_{IM};

Experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathrm{cca2-anon},b}(\lambda)$

    $(pk_{\mathsf{IM}}, sk_{\mathsf{IM}}) \leftarrow \texttt{SetupIM}(1^\lambda); (pk_{\mathsf{OA}}, sk_{\mathsf{OA}}) \leftarrow \texttt{SetupOA}(1^\lambda);$

    $(\mathtt{id}_0, \mathtt{id}_1, m) \leftarrow \mathcal{A}^{\mathsf{CorruptIMOracle}(), \mathsf{OpenOracle}()}(1^\lambda, pk_{\mathsf{IM}}, pk_{\mathsf{OA}});$

    $\mathtt{cert}_{\mathtt{id}_0} \leftarrow \texttt{Reg}(pk_{\mathsf{IM}}, pk_{\mathsf{OA}}; sk_{\mathsf{IM}}, \mathtt{id}_0); \mathtt{cert}_{\mathtt{id}_1} \leftarrow \texttt{Reg}(pk_{\mathsf{IM}}, pk_{\mathsf{OA}}; sk_{\mathsf{IM}}, \mathtt{id}_1);$

    $\sigma \leftarrow \texttt{Sign}(pk_{\mathsf{IM}}, pk_{\mathsf{OA}}, \mathtt{id}_b; \mathtt{cert}_{\mathtt{id}_b}, m);$

    $b^* \leftarrow \mathcal{A}^{\mathsf{CorruptIMOracle}(), \mathsf{OpenOracle}^{\neg \sigma}()}(1^\lambda, pk_{\mathsf{IM}}, pk_{\mathsf{OA}});$

    Return b^*;

Fig. 3. Experiment of CCA2-anonymity. In the experiment above, $\mathsf{OpenOracle}^{\neg \sigma}()$ operates as the $\mathsf{OpenOracle}()$ with the restriction that it will return $\perp$ if the adversary submit σ as the signature to be opened.

Experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathrm{cpa-anon},b}(\lambda)$

    $(pk_{\mathsf{IM}}, sk_{\mathsf{IM}}) \leftarrow \texttt{SetupIM}(1^\lambda); (pk_{\mathsf{OA}}, sk_{\mathsf{OA}}) \leftarrow \texttt{SetupOA}(1^\lambda);$

    $(\mathtt{id}_0, \mathtt{id}_1, m) \leftarrow \mathcal{A}^{\mathsf{CorruptIMOracle}()}(1^\lambda, pk_{\mathsf{IM}}, pk_{\mathsf{OA}});$

    $\mathtt{cert}_{\mathtt{id}_0} \leftarrow \texttt{Reg}(pk_{\mathsf{IM}}, pk_{\mathsf{OA}}; sk_{\mathsf{IM}}, \mathtt{id}_0); \mathtt{cert}_{\mathtt{id}_1} \leftarrow \texttt{Reg}(pk_{\mathsf{IM}}, pk_{\mathsf{OA}}; sk_{\mathsf{IM}}, \mathtt{id}_1);$

    $\sigma \leftarrow \texttt{Sign}(pk_{\mathsf{IM}}, pk_{\mathsf{OA}}, \mathtt{id}_b; \mathtt{cert}_{\mathtt{id}_b}, m);$

    $b^* \leftarrow \mathcal{A}^{\mathsf{CorruptIMOracle}()}(1^\lambda, pk_{\mathsf{IM}}, pk_{\mathsf{OA}});$

    Return b^*;

Fig. 4. Experiment of CPA-anonymity. In the experiment above, the $\mathsf{CorruptIMOracle}()$ used is same as that in the CCA2 version, and the $\mathsf{OpenOracle}()$ is not allowed.

for the Hidden-IBS as $\mathtt{pub} = \langle p, g, \widehat{g}, h, \widehat{h}, \mathbb{G}, \widehat{\mathbb{G}}, \psi, \mathbb{G}_T, e; \widehat{X}, \widehat{Y}; u, v, w, \widehat{u}, \widehat{v}, \widehat{w}; \mathcal{H} \rangle$. We still need to prescribe the form of the user identities: each identity is a short string with length ℓ. For example, it can be an IP address with $\ell = 32$ or a userid in a reputation system (e.g., using $\ell = 50$ we can allow 10 character long userids with 5 bits per character).

$\texttt{Reg}$. In the registration algorithm, the user sends her identity $\mathtt{id}$ to the IM. The IM verifies that $\mathtt{id}$ is acceptable (e.g., not being used before or not blacklisted etc.). We note that the $\mathtt{id}$ can also be a product of a negotiation between the IM and the users. Then the IM generates a BB signature $\langle s, r \rangle$ for $\mathtt{id}$, where $s \leftarrow g^{\frac{1}{x + \mathtt{id} + yr}}$, $r \xleftarrow{\mathrm{r}} \mathbb{Z}_p$, and sends $\langle s, r \rangle$ to the user by a secure communication channel.

$\texttt{RegCheck}$. Once receiving the signature $\langle s, r \rangle$ from the IM, the user verifies $e(s, \widehat{X}\widehat{g}^{\mathtt{id}}\widehat{Y}^r) = e(g, \widehat{g})$. The user sets her membership certificate to $\mathtt{cert}_{\mathtt{id}} = \langle s, r \rangle$.

$\texttt{Sign}$. With a membership certificate $\mathtt{cert}_{\mathtt{id}} = \langle s, r \rangle$ in hand, a user can compute a Hidden-IBS signature σ for message m. We first develop a proof of knowledge in figure 5, where the user proves her knowledge of $\mathtt{id}$ and $\mathtt{cert}_{\mathtt{id}}$, and proves

that `cert`$_{\mathtt{id}}$ is a BB signature of `id` from the IM. Then we transform the proof of knowledge into a signing algorithm by using the Fiat-Shamir heuristic.

`Verify`. The verifier can verify a message-signature pair by checking the equation

$$c =^? \mathcal{H}\big(m||S||\widehat{R}||U||V||\widehat{W}||U^c u^{-\xi_k}||V^c v^{-\xi_l}||\widehat{W}^c \widehat{w}^{-(\xi_k+\xi_l)}\widehat{g}^{-\xi_{\mathtt{id}}}$$
$$||\widehat{R}^c \widehat{g}^{-\xi_{r_2}}\widehat{h}^{-\xi_{r_1}}\widehat{Y}^{-\xi_r}||U^{-\xi_{r_1}}u^{\xi_{\delta_1}}||V^{-\xi_{r_1}}v^{\xi_{\delta_2}}||\widehat{R}^{-\xi_{r_1}}\widehat{g}^{\xi_{\delta_3}}\widehat{h}^{\xi_{\delta_4}}\widehat{Y}^{\xi_{\delta_5}}||e(g,\widehat{X}\widehat{W}\widehat{R})^{\xi_{r_1}}$$
$$e(S,\widehat{w})^{(\xi_k+\xi_l)}e(g,\widehat{w})^{-(\xi_{\delta_1}+\xi_{\delta_2})}e(S,\widehat{g})^{\xi_{r_2}}e(g,\widehat{g})^{-\xi_{\delta_3}}e(S,\widehat{h})^{\xi_{r_1}}e(g,\widehat{h})^{-\xi_{\delta_4}}$$
$$(e(g,\widehat{g})/e(S,\widehat{X}\widehat{W}\widehat{R}))^c\big)$$

`Open`. Given a message-signature pair as described above, the OA first verifies the message-signature pair. Next the OA uses her secret key $sk_{\mathsf{OA}} = \langle \zeta, \eta \rangle$ to open ciphertext $\langle U, V, W \rangle$ into $g^{\mathtt{id}}$ where $W = \psi(\widehat{W})$; considering that the identity space is small, the OA recovers `id` from $g^{\mathtt{id}}$.

Theorem 1. *The Hidden-IBS scheme is correct and secure satisfying misidentification-forgery and CPA-anonymity in the random oracle model under the SDH and the DLDH assumptions.*

4   Reducing Abuse in Anonymous Routing Systems

As mentioned in the introduction some internet services block certain types of traffic coming through anonymous routing systems in order to maintain the quality of their service (e.g., in the case of Wikipedia, POST requests coming from Tor are blocked to prevent vandalism). This practice stems from the fact that anonymous routing systems such as Tor have no built-in mechanisms to handle abusive users. In this section, we show how using our Hidden-IBS we can strengthen the Tor network with the capability to defend itself against such abusive users.

Our approach, outlined in figure 6, adds three entities to the Tor network deployment: the Identity Manager (IM) of a Hidden-IBS, a Disputes&Grievances database and the Opening Authority (OA) of the Hidden-IBS. Our basic idea is to show how a service web-site that receives Tor traffic can complain about malicious requests (e.g., vandalism in the case of Wikipedia) and recover some information about the offending users. In this way the anonymous routing system offers a mechanism to prevent abusive users from taking advantage of anonymity and thus its services can be granted higher functionality by service providers. Our enhancement to Tor will be totally transparent to service web-sites that receive Tor traffic.

More specifically now, the Hidden-IBS enhanced Tor works like this: certain packets generated by a Tor user are permitted through the Tor exit point only if they carry a Hidden-IBS. The Tor user's onion proxy (OP) catches this and assists the user to get the Hidden-IBS signing capability. Then any packet that needs to be signed is hashed and then signed. Tor exit points verify the Hidden-IBS signature on the hashed reconstructed packet and forward the packet (with the signature removed) to the web-site that the packet was directed while they write the hashed packet together with the signature to a Disputes&Grievances

$$\text{pub} = \langle p, g, \widehat{g}, \mathbb{G}, \widehat{\mathbb{G}}, \psi, \mathbb{G}_T, e; h, \widehat{h}; \widehat{X}, \widehat{Y}; u, v, w, \widehat{u}, \widehat{v}, \widehat{w} \rangle$$

User	Verifier
id, s, r	

$r_1, r_2, k, l \xleftarrow{\text{r}} \mathbb{Z}_p,$
$S = g^{r_1} s, \ \widehat{R} = \widehat{g}^{r_2} \widehat{h}^{r_1} \widehat{Y}^r,$
$\delta_1 = r_1 k, \ \delta_2 = r_1 l,$
$\delta_3 = r_1 r_2, \ \delta_4 = r_1^2, \ \delta_5 = r_1 r$
$U = u^k, \ V = v^l, \ \widehat{W} = \widehat{w}^{k+l} \widehat{g}^{\text{id}}$
$\theta_{\text{id}}, \theta_r, \theta_{r_1}, \theta_{r_2}, \theta_k, \theta_l \xleftarrow{\text{r}} \mathbb{Z}_p,$
$\theta_{\delta_1}, \theta_{\delta_2}, \theta_{\delta_3}, \theta_{\delta_4}, \theta_{\delta_5} \xleftarrow{\text{r}} \mathbb{Z}_p$
$B_1 = u^{-\theta_k}, \ B_2 = v^{-\theta_l},$
$B_3 = \widehat{w}^{-(\theta_k + \theta_l)} \widehat{g}^{-\theta_{\text{id}}},$
$B_4 = \widehat{g}^{-\theta_{r_2}} \widehat{h}^{-\theta_{r_1}} \widehat{Y}^{-\theta_r},$
$B_5 = U^{-\theta_{r_1}} u^{\theta_{\delta_1}}, \ B_6 = V^{-\theta_{r_1}} v^{\theta_{\delta_2}}$
$B_7 = \widehat{R}^{-\theta_{r_1}} \widehat{g}^{\theta_{\delta_3}} \widehat{h}^{\theta_{\delta_4}} \widehat{Y}^{\theta_{\delta_5}}$
$B_8 = e(g, \widehat{X} \widehat{W} \widehat{R})^{\theta_{r_1}} e(S, \widehat{w})^{\theta_k + \theta_l}.$
  $e(g, \widehat{w})^{-(\theta_{\delta_1} + \theta_{\delta_2})} e(S, \widehat{g})^{\theta_{r_2}}.$
  $e(g, \widehat{g})^{-\theta_{\delta_3}} e(S, \widehat{h})^{\theta_{r_1}} e(g, \widehat{h})^{-\theta_{\delta_4}}$

$$\xrightarrow{\quad S, \widehat{R}, U, V, \widehat{W} \quad}$$
$$\xrightarrow{\quad B_1, \dots, B_8 \quad}$$

$c \xleftarrow{\text{r}} \mathbb{Z}_p$

$$\xleftarrow{\qquad c \qquad}$$

$\xi_{\text{id}} = \theta_{\text{id}} + c \cdot \text{id}, \ \xi_r = \theta_r + c \cdot r,$
$\xi_{r_1} = \theta_{r_1} + c \cdot r_1, \ \xi_{r_2} = \theta_{r_2} + c \cdot r_2$
$\xi_k = \theta_k + c \cdot k, \ \xi_l = \theta_l + c \cdot l$
$\xi_{\delta_1} = \theta_{\delta_1} + c \cdot \delta_1, \ \xi_{\delta_2} = \theta_{\delta_2} + c \cdot \delta_2$
$\xi_{\delta_3} = \theta_{\delta_3} + c \cdot \delta_3, \ \xi_{\delta_4} = \theta_{\delta_4} + c \cdot \delta_4$
$\xi_{\delta_5} = \theta_{\delta_5} + c \cdot \delta_5,$

$$\xrightarrow{\quad \xi_{\text{id}}, \xi_r, \xi_{r_1}, \xi_{r_2}, \quad}$$
$$\xrightarrow{\quad \xi_k, \xi_l, \xi_{\delta_1}, \dots, \xi_{\delta_5} \quad}$$

$u^{\xi_k} B_1 =^? U^c, \ v^{\xi_l} B_2 =^? V^c$
$\widehat{w}^{\xi_k + \xi_l} \widehat{g}^{\xi_{\text{id}}} B_3 =^? \widehat{W}^c$
$\widehat{g}^{\xi_{r_2}} \widehat{h}^{\xi_{r_1}} \widehat{Y}^{\xi_r} B_4 =^? \widehat{R}^c$
$U^{\xi_{r_1}} u^{-\xi_{\delta_1}} B_5 =^? 1,$
$V^{\xi_{r_1}} v^{-\xi_{\delta_2}} B_6 =^? 1$
$\widehat{R}^{\xi_{r_1}} \widehat{g}^{-\xi_{\delta_3}} \widehat{h}^{-\xi_{\delta_4}} \widehat{Y}^{-\xi_{\delta_5}} B_7 =^? 1$
$e(g, \widehat{X} \widehat{W} \widehat{R})^{-\xi_{r_1}} e(S, \widehat{w})^{-(\xi_k + \xi_l)}.$
  $e(g, \widehat{w})^{(\xi_{\delta_1} + \xi_{\delta_2})} e(S, \widehat{g})^{-\xi_{r_2}}.$
  $e(g, \widehat{g})^{\xi_{\delta_3}} e(S, \widehat{h})^{-\xi_{r_1}} e(g, \widehat{h})^{\xi_{\delta_4}} B_8$
  $=^? (e(g, \widehat{g}) / e(S, \widehat{X} \widehat{W} \widehat{R}))^c$

Fig. 5. The hidden identity-based identification protocol
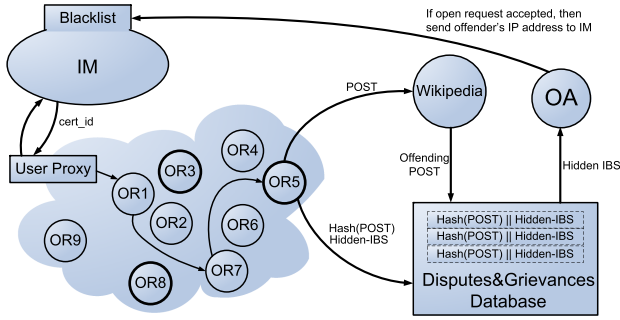
Fig. 6. Enhancing the Tor network with a mechanism to defend against anonymity abuse using the Hidden-IBS primitive. Note that we use IP addresses as user identities in the figure but other types of identities can be used, e.g., userids of a reputation system.

database. If any vandalism is caught by a service provider, the service-provider using the packet that was sent through Tor by the abusive user can retrieve the corresponding Hidden-IBS from the database and forward it to the OA along with a complaint report. Based on the properties of the Hidden-IBS scheme, the OA can open the signature and recover the identity of the abusive user. Subsequently the IM can be notified of the abusive user's identity and the user can be punished by being black-listed (or receiving a negative point in a reputation system). Below we describe in more details how we propose to deploy our Hidden-IBS enhanced Tor system for handling HTTP POST requests to Wikipedia. Note that all other traffic through Tor would be unaffected (i.e., it would not require a signature).

When the user first installs a Tor OP she can obtain a certificate cert_id for her identity id from the IM. The id that the user deposits to the IM can be the user's IP address or a long-lived userid in a reputation system. Subsequently whenever the user wants to send an HTTP POST the OP builds a route to a Tor exit point (in the figure, this route is OR1,OR7,OR5, and OR5 is the Tor exit point). When the user generates a POST request for a Wikipedia web-site the following things happen: (*i*) the user's browser passes the POST request, say post1 to the OP; (*ii*) the OP sanitizes post1 into post2 so that the header of post2 does not contain any unnecessary identity related information; (*iii*) the OP generates a random nonce and stored in a Nonce field into the header of post2, resulting to packet post3; (*iv*) the OP hashes post3 and signs the hash with the Hidden-IBS signing algorithm; (*v*) the OP creates a new field called Signature in the header of post3 and fills it with the generated signature; we call the modified post3 as post4; (*vi*) the OP forwards the post4 along the established circuit.

When a Tor exit point assembles a POST request such as post4 above, it parses the field Signature and obtains the Hidden-IBS signature; then it transforms post4 into post3 by throwing away the Signature field in the header

and computes the hash value of post3 to verify the signature (using the public-key of the IM). Finally, if the signature verifies, the exit point forwards post3 to the Wikipedia web-site; at the same time it submits the hash value and the Hidden-IBS signature to the Disputes&Grievances database.

Wikipedia may now keep the POST request coming through a Tor exit point (or in fact only the hash of the request suffices). If a certain posting is found to be offensive or abusive the web-site may search for the corresponding Hidden-IBS signature into the Disputes&Grieances database (that will be indexed based on the hash of the post). Then, once the hidden-IBS is recovered it can be submitted to the opening authority (OA) along with a complaint report. The OA uses his secret key to open the Hidden-IBS and recover offender's identity (e.g., her IP address), and then sends this identity to the IM. The IM may blacklist this identity which may result in refusing future registration requests originating from the offender's IP address for example. Other strategies may be followed here by the IM, for example if the identity is a userid in a reputation system the user may receive a negative point.

References

1. Ateniese, G., Camenisch, J., Hohenberger, S., de Medeiros, B.: Practical group signatures without random oracles. In: Cryptology ePrint Archive, Report 2005/385 (2005), http://eprint.iacr.org/2005/385/
2. Ateniese, G., Camenisch, J., Joye, M., Tsudik, G.: A practical and provably secure coalition-resistant group signature scheme. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 255–270. Springer, Heidelberg (2000)
3. Ateniese, G., de Medeiros, B.: Efficient group signatures without trapdoors. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 246–268. Springer, Heidelberg (2003)
4. Bellare, M., Micciancio, D., Warinschi, B.: Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In: Biham, E. (ed.) EUROCRPYT 2003. LNCS, vol. 2656, pp. 614–629. Springer, Heidelberg (2003)
5. Bellare, M., Shi, H., Zhang, C.: Foundations of group signatures: The case of dynamic groups. In: Menezes, A.J. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 136–153. Springer, Heidelberg (2005)
6. 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)
7. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
8. Boneh, D., Shacham, H.: Group signatures with verifier-local revocation. In: Atluri, V., Pfitzmann, B., McDaniel, P.D. (eds.) CCS 2004, pp. 168–177. ACM Press, New York (2004)
9. Boyen, X., Waters, B.: Compact group signatures without random oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 427–444. Springer, Heidelberg (2006)
10. Camenisch, J.: Efficient and generalized group signatures. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 465–479. Springer, Heidelberg (1997)

11. Camenisch, J., Groth, J.: Group signatures: Better efficiency and new theoretical aspects. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 120–133. Springer, Heidelberg (2005)
12. Camenisch, J., Lysyanskaya, A.: An identity escrow scheme with appointed verifiers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 388–407. Springer, Heidelberg (2001)
13. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 56–72. Springer, Heidelberg (2004)
14. Camenisch, J., Maurer, U.M., Stadler, M.: Digital payment systems with passive anonymity-revoking trustees. In: Bertino, E., Kurth, H., Martella, G., Montolivo, E. (eds.) ESORICS 1996. LNCS, vol. 1146, pp. 33–43. Springer, Heidelberg (1996)
15. Camenisch, J., Michels, M.: A group signature scheme with improved efficiency. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 160–174. Springer, Heidelberg (1998)
16. Camenisch, J., Michels, M.: Separability and efficiency for generic group signature schemes. In: Wiener, M.J. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 413–430. Springer, Heidelberg (1999)
17. Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups (extended abstract). In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 410–424. Springer, Heidelberg (1997)
18. Chaum, D., van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991)
19. Chen, L., Pedersen, T.P.: New group signature schemes (extended abstract). In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 171–181. Springer, Heidelberg (1995)
20. Dingledine, R., Mathewson, N., Syverson, P.F.: Tor: The second-generation onion router. In: USENIX Security Symposium 2004, USENIX, pp. 303–320 (2004)
21. Frankel, Y., Tsiounis, Y., Yung, M.: "Indirect Discourse Proof": Achieving efficient fair off-line e-cash. In: Kim, K.-c., Matsumoto, T. (eds.) ASIACRYPT 1996. LNCS, vol. 1163, pp. 286–300. Springer, Heidelberg (1996)
22. Furukawa, J., Imai, H.: An efficient group signature scheme from bilinear maps. In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 455–467. Springer, Heidelberg (2005)
23. Furukawa, J., Yonezawa, S.: Group signatures with separate and distributed authorities. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 77–90. Springer, Heidelberg (2005)
24. Goldschlag, D.M., Reed, M.G., Syverson, P.F.: Hiding routing information. In: Anderson, R.J. (ed.) Information Hiding. LNCS, vol. 1174, pp. 137–150. Springer, Heidelberg (1996)
25. Heller, J.L.: Catch-22. Simon & Schuster (1961)
26. Kiayias, A., Tsiounis, Y., Yung, M.: Traceable signatures. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 571–589. Springer, Heidelberg (2004)
27. Kiayias, A., Yung, M.: Efficient secure group signatures with dynamic joins and keeping anonymity against group managers. In: Dawson, E., Vaudenay, S. (eds.) Mycrypt 2005. LNCS, vol. 3715, pp. 151–170. Springer, Heidelberg (2005), http://eprint.iacr.org/2004/076/
28. Kiayias, A., Yung, M.: Group signatures with efficient concurrent join. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 198–214. Springer, Heidelberg (2005)

29. Kiayias, A., Zhou, H.-S.: Hidden identity-based signatures. Manuscript available from the authors (2006)
30. Nguyen, L., Safavi-Naini, R.: Efficient and provably secure trapdoor-free group signature schemes from bilinear pairings. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 372–386. Springer, Heidelberg (2004)
31. 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)
32. Tor, `http://tor.eff.org/`
33. von Solms, S.H., Naccache, D.: On blind signatures and perfect crimes. Computers & Security 11(6), 581–583 (1992)
34. Wei, V.K., Yuen, T.H., Zhang, F.: Group signature where group manager, members and open authority are identity-based. In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 468–480. Springer, Heidelberg (2005)
35. Wikipedia, `http://wikipedia.org/`
36. Wikipedia.: Advice to Tor users in China (May 2006), `http://en.wikipedia.org/wiki/Wikipedia:Tor`

Space-Efficient Private Search
with Applications to Rateless Codes

George Danezis and Claudia Diaz

K.U. Leuven, ESAT/COSIC,
Kasteelpark Arenberg 10,
B-3001 Leuven-Heverlee, Belgium
{george.danezis, claudia.diaz}@esat.kuleuven.be

Abstract. Private keyword search is a technique that allows for searching and retrieving documents matching certain keywords without revealing the search criteria. We improve the space efficiency of the Ostrovsky *et al.* Private Search [9] scheme, by describing methods that require considerably shorter buffers for returning the results of the search. Our basic decoding scheme *recursive extraction*, requires buffers of length less than twice the number of returned results and is still simple and highly efficient. Our extended decoding schemes rely on solving systems of simultaneous equations, and in special cases can uncover documents in buffers that are close to 95% full. Finally we note the similarity between our decoding techniques and the ones used to decode rateless codes, and show how such codes can be extracted from encrypted documents.

1 Introduction

Private search allows for keyword searching on a stream of documents (typical of online environments) without revealing the search criteria. Its applications include intelligence gathering, medical privacy, private information retrieval and financial applications. Financial applications that can benefit from this technique are, for example, corporate searches on a patents database, searches for financial transactions meeting specific but private criteria and periodic updates of filtered financial news or stock values.

Rafail Ostrovsky *et al.* presented in [9] a scheme that allows a server to filter a stream of documents, based on matching keywords, and only return the relevant documents without gaining any information about the query string. This allows searching to be outsourced, and only relevant results to be returned, economising on communications costs. The authors of [9] show that the communication cost is linear in the number of results expected. We extend their scheme to improve the space-efficiency of the returned results considerably by using more efficient coding and decoding techniques.

Our first key contribution is a method called *recursive extraction* for efficiently decoding encrypted buffers resulting from the Ostrovsky *et al.* scheme. The second method, based on solving systems of linear equations, is applied after recursive extraction and allows for the recovery of extra matching documents

from the encrypted buffers. Recursive extraction results in the full decoding of buffers of length twice the size of the expected number of matches, and has a linear time-complexity. Shorter buffers can also be decrypted with high probability. Solving the remaining equations at colliding buffer positions allows for even more documents to be retrieved from short buffers, and in the special case of documents only matching one keyword, we can decode buffers that are only 10% longer than the expected matches, with high probability. We present simulations to assess the decoding performance of our techniques, and estimate optimal parameters for our schemes.

In this work we also present some observations that may be of general interest beyond the context of private search. We show how arrays of small integers can be represented in a space efficient manner using Pailler ciphertexts, while maintaining the homomorphic properties of the scheme. These techniques can be used to make private search more space-efficient, but also implement other data structures like Bloom filters, or vectors in a compact way. Finally we show how rateless codes, block based erasure resistant multi-source codes, can be extracted from encrypted documents, while maintaining all their desirable properties.

This paper is structured as follows: We introduce the related work in Section 2; present more in detail in Section 3 the original Ostrovsky scheme whose efficiency we are trying to improve; and explain in Section 4 the required modifications. Sections 5 and 6 present the proposed efficient decoding techniques, which are evaluated in Section 7. In Section 8 we explain how our techniques can be applied to rateless codes; and we present our conclusions in Section 9.

2   Related Work

Our results can be applied to improve the decoding efficiency of the Private Search scheme proposed by Rafail Ostrovsky *et al.* in [9]. This scheme is described in detail in Section 3. Danezis and Diaz proposed in [5] some preliminary ideas on how to improve the decoding efficiency of the Ostrovsky Private Search scheme, which are elaborated in this paper.

Bethencourt *et al.* [1,2] have independently proposed several modifications to the Ostrovsky private search scheme which include solving a system of linear equations to recover the documents. As such, the time complexity of their approach is $\mathcal{O}(n^3)$, while our base technique, recursive extraction, is $\mathcal{O}(n)$. Their technique also requires some changes to the original scheme [9], such as the addition of an encrypted buffer that acts as a Bloom filter [3]. This buffer by itself increases by 50% the data returned. Some of our techniques presented in section 6.2, that allow for efficient space representation of concatenated data, are complementary to their work, and would greatly benefit the efficiency of their techniques.

The rateless codes for big downloads proposed by Maymounkov and Mazières in [8] use a technique similar to ours for efficient decoding, indicating that our ideas can be applied beyond private search applications. We explore further this relation in Section 8, where we show how homomorphic encryption can be used to create rateless codes for encrypted data.

Pfitzmann and Wainer [13] also notice that collisions in DC networks [4] do not destroy all information transmitted. They use this observation to allow n messages to be transmitted in n steps despite collisions.

3   Private Search

The Private Search scheme proposed by Ostrovsky *et al.* [9] is based on the properties of the homomorphic Paillier public key cryptosystem [10], in which the multiplication of two ciphertexts leads to the encryption of the sum of the corresponding plaintexts ($E(x) \cdot E(y) = E(x + y)$). Constructions with El-Gamal [6] are also possible but do not allow for full recovery of documents.

The searching party provides a dictionary of terms and a corresponding Paillier ciphertext, that is the encryption of one ($t_i = E(1)$), if the term is to be matched, or the encryption of zero ($t'_i = E(0)$) if the term is of no interest. Because of the semantic security properties of the Paillier cryptosystem this leaks no information about the matching criteria.

The dictionary ciphertexts corresponding to the terms in the document d_j are multiplied together to form $g_j = \prod_k t_k = E(m_j)$, where m_j is the number of matching words in document d_j. A tuple $(g_j, g_j^{E(d_j)})$ is then computed. The second term will be an encryption of zero ($E(0)$) if there has been no match, and the encryption $E(m_j d_j)$ otherwise. Note that repeated words in the document are not taken into account, meaning that each matching word is counted only once, and m_j represents the number of different matching words found in a document.

Each document tuple is then multiplied into a set of l random positions in a buffer of size b (smaller than the total number of searched documents, but bigger than the number of matching documents). All buffer positions are initialized with tuples ($E(0), E(0)$). The documents that do not match any of the keywords, do not contribute to changing the contents of these positions in the buffer (since zero is being added to the plaintexts), but the matched documents do.

Collisions will occur when two matching documents are inserted at the same position in the buffer. These collisions can be detected by adding some redundancy to the documents. The color survival theorem [9] can be used to show that the probability that all copies of a single document are overwritten becomes negligibly small as the number of l copies and the size of the buffer b increase (the suggested buffer length is $b = 2 \cdot l \cdot M$, where M is the expected number of matching documents). The searcher can decode all positions, ignoring the collisions, and dividing the second term of the tuples by the first term to retrieve the documents.

4   Modifications to the Original Scheme

A prerequisite for more efficient decoding schemes is to reduce the uncertainty of the party that performs the decoding. At the same time, the party performing

the search should gain no additional information with respect to the original scheme. In order to make sure of this, we note that the modifications to the original scheme involve only information flows from the searching (encoding) party back to the matching (decoding) party, and therefore cannot introduce any additional vulnerabilities in this respect.

Our basic decoding algorithm (presented in Section 5) only requires that the document copies are stored in buffer positions known to the decoder. In practice, the mapping of documents to buffer positions can be done using a good hash function $H(\cdot)$ that can be agreed by both parties or fixed by the protocol. We give an example of how this function can be constructed.

Notation:

- l is the total of copies stored per document;
- d_{ij} is the j-th copy of document d_i $(j = 1 \ldots l)$ – note that all copies of d_i are equal;
- b is the size of the buffer;
- q is the number of bits needed to represent b $(2^{q-1} < b \leq 2^q)$;
- p_{ij} is the position of document copy d_{ij} in the buffer $(0 \leq p_{ij} < b)$.

The hash function is applied to the sum of the the document d_i and the copy number j, $H(d_i + j)$. The position p_{ij} is then represented by the q most significant bits of the result of the hash. If there is index overflow (i.e., $b \leq p_{ij}$), then we apply the hash function again $(H(H(d_i + j)))$ and repeat the process, until we obtain a result $p_{ij} < b$. This is illustrated in Figure 1(a).

With this method, once the decoding party sees a copy of a matched document, d_i, it can compute the positions of the buffer where all l copies of d_i have been stored (and thus extract them from those positions) by applying the function to $d_i + j$, with $j = 1 \ldots l$.

We present in Section 6 an extension to our decoding algorithm that further improves its decoding efficiency. The extension requires that the total number N of searched documents is known to the decoder, and that the positions of *all* (not just matched) searched documents are known by the decoder. This can be achieved by adding a serial number s_i to the documents, and then deriving the position p_{ij} of the document copies as a function of the document serial number and the number of the copy $H(s_i||j)$, as shown in Figure 1(b). We then take the q most significant bits of the result and proceed as in the previous case.

With respect to the original Ostrovsky scheme, our basic algorithm only requires the substitution of the random function $U[0, b-1]$ used to select the buffer positions for the document copies by a pseudorandom function dependent on the document and the copy number, that can be computed by the decoder.

The extension requires that the encoder transmits to the decoder the total number N of documents searched. The encoder should also append a serial number to the documents (before encrypting them). We assume that the serial numbers take values between 1 and N (i.e., $s_i = i$).

(a) in the basic algorithm   (b) in the extended algorithm
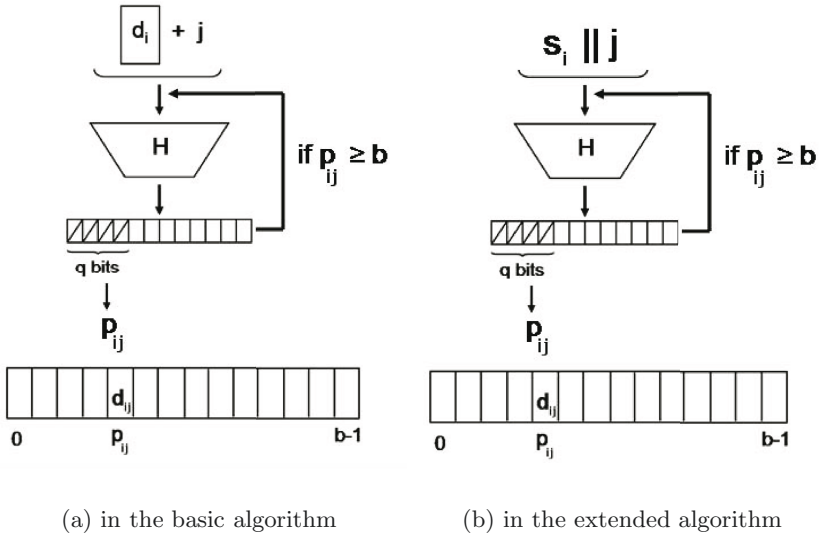
Fig. 1. Function to determine the position of a document copy d_{ij}

5   Basic Decoding Algorithm: Recursive Extraction

Given the minor modifications above, we note that much more efficient decoding algorithms can be used, that would allow the use of significantly smaller buffers for the same recovery probability.

While collisions are ignored in the original Ostrovsky scheme, our key intuition is that collisions are in fact not destroying all information, but merely adding together the encrypted plaintexts. This property can be used to recover a plaintext if the values of the other plaintexts with which it collides are known.

The decoder decrypts the buffer, and thanks to the redundancy included in the documents, it can discern three states of a particular buffer position: whether it is empty, contains a single document, or contains a collision.

In this basic scheme, the empty buffer positions are of no interest to the decoder (they do provide useful information in the extended algorithm, as we shall see in the next section). In the case of it containing a single document d_i, then the document can be recovered. By applying the hash function as described in Section 4 to $d_i + j$ with $j = 1 \ldots l$, the decoder can locate all the other copies of d_i and extract them from the buffer. This hopefully uncovers some new buffer positions containing only one document. This simple algorithm is repeated multiple times until all documents are recovered or no more progress can be made.

In the example shown in Figure 2(a), we match 9 documents and store 3 copies of each in a buffer of size 24. Documents '3', '5', '7' and '8' can be trivially recovered (note that these four documents would be the only ones recovered in the original scheme). All copies of these documents are located and extracted
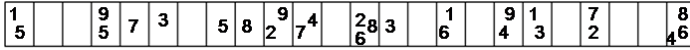
(a) Example of full buffer with 9 matched documents, 3 copies.
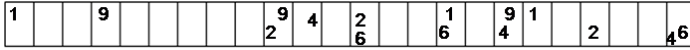
(b) Buffer after documents '3', '5', '7' and '8' have been removed.

(c) Buffer after documents '1', '2', '4' and '9' have been removed.

Fig. 2. Example of recursive extraction

from the buffer. At this point, documents '1', '2', '4' and '9' appear alone in at least one position, and can therefore be extracted. Once they are removed from the buffer, document '6' can be also retrieved.

This very simple algorithm already provides an improvement of a factor l (number of document copies) over the original Ostrovsky scheme, as we show in our evaluation in Section 7.

6   Extended Decoding Algorithm: Solving Equations

Our basic decoding algorithm may terminate without recovering all matching documents if we run into a situation where a group of documents is copied to the same set of buffer positions. Figure 3 gives an example of such a case: 3 matching documents (2 copies each) have been stored in 3 colliding buffer positions. Our key observation is that by expressing these buffer positions as linear equations, we can still retrieve the 3 colliding documents.

Our basic decoding based on recursive extraction algorithm takes advantage of the fact that, once one document copy is retrieved, all other copies can be extracted from the buffer. This does not require the buffer positions of the document
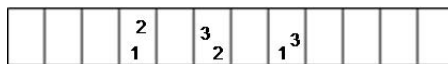
Fig. 3. Example of buffer with 2 copies of 3 documents colliding in 3 buffer positions

copies to be predictable for the decoder, and he needs to see at least one copy of the document before being able to compute the positions of the other copies.

Making predictable the positions of the document copies in the buffer, further reduces uncertainty and allows us to further improve the decoding efficiency. In order to achieve this we include, as pointed out in Section 4, a serial number in the document and tell the decoder the total number N of documents searched. Then we make the document copy position dependent on its serial and copy numbers (as shown in Figure 1(b)), so that the position of each searched document copy is known *a priori* to the decoder.

The end resulting buffer can be modeled as a system of simultaneous equations. Each equation represents a buffer position, which leads to at most b equations. Each document that has a copy in this buffer position is a variable in the equation, and the sum of the actual matched documents equals the value of the bucket. Note that non-matching documents are set to zero, and therefore they do not contribute to the sum.

The example shown in Figure 3 represents a mapping of three documents d_1, d_2 and d_3 into three buckets b_3, b_5 and b_7. The corresponding set of equations would be:

$$d_1 + d_2 = b_3 \tag{1}$$
$$d_2 + d_3 = b_5 \tag{2}$$
$$d_1 + d_3 = b_7 \tag{3}$$

We can solve the system of linear equations as long as the number of unknowns (i.e., documents) is lower or equal than the number of equations (i.e., buffer positions).

Note that, if a non-matching document d_4 is also allocated in two of the buffer positions (say, b_3 and b_7), we cannot retrieve any document because there are more unknowns that equations. Therefore the key to the success of this decoding technique is applying first the recursive extraction decoding to identify as many matching and non-matching documents possible; empty buffer positions give key information about non-matching documents. This aims to reduce the number of unknown documents to be less than the available equations, allowing us to solve the linear system. As such, solving equations is complementary to the first decoding technique, and it is always applied after recursive extraction.

6.1 Special Case: Searching for One Keyword

In some applications, the decoder may be interested in searching only one keyword in the documents. For example, when retrieving pseudonymous email [11], the decoder would provide his email address as the only keyword for searching in the documents.

In these cases, we can further improve the decoding efficiency to the extent that the buffer length needs to be less than 10% larger than the expected number of matches. As we show in Section 7, with this technique we can with high

probability retrieve more than 900 matched documents from a buffer with 1000 positions.

The technique works as follows: the serial number is appended to the lower end of the document, leaving enough space to accommodate the sum of serial numbers of documents present in the bucket. The modified documents d_i' are computed as: $d_i' = d_i \cdot 2^S + i$, where S is the number of bits needed to make sure that the serial numbers add up without overflowing the appended bit space. Given that the total number of searched documents is N, the worst case would be that all searched documents match and that there is a bucket that contains a copy of each document. In this case, the sum of all serial numbers is $\sum_{i=1}^{N} i = \frac{N(N+1)}{2}$. Therefore, it suffices to append $S = 2\log_2(N)$ bits to the document, where the serial number i is included. Note that a lower number of bits between $\log_2(N)$ and $2\log_2(N)$ may be sufficient, since the average number of matched documents in a buffer position is generally much lower than the worst case.

As we are considering the special case in which only one keyword is being searched, matching documents would be multiplied by a one, while non-matching documents are multiplied by a zero (as opposed to the general case of searching K keywords, where the document may be multiplied by up to K if all searched keywords are contained in it). The serial numbers, as part of the document, are also multiplied by either a one or a zero.

This scheme has the following property: given that 2 matching documents d_i and d_j are colliding in a buffer position b_k, the serial number bits of the collision will contain $i + j$. More generally, the sum of serial numbers in a given buffer position is given by: $R = \sum_{i=1}^{N}(i \cdot x_i \cdot y_i)$, where x_i is one if a document copy has been stored in that buffer position and zero otherwise (the values of x_i are known *a priori* to both the encoder and the decoder); and the variable y_i takes the value one if the document has been matched and zero otherwise (the value of y_i is unknown both to the encoder and the decoder). Note that in the general case of searching K keywords y_i takes values between zero and K.

In order to perform the decoding, we use the following information:

- M: number of matching documents in a buffer position (given by the decryption of the multiplication of first parts of the tuple $(g_i, g_i^{d_i})$ for each document d_i in the buffer position, $M = D(\prod_i(g_i)) = \sum_i(x_i \cdot y_i)$)
- R: result of summing all serial numbers of matching documents in the position, $R = \sum_{i=1}^{N}(i \cdot x_i \cdot y_i)$.

We then seek the subset of M documents stored in the position whose serial numbers sum R. We illustrate the method with the example shown in Figure 4. In this example, we can express the equations for the buffer positions as:

$$d_1 + d_2 + d_3 = b_3 \tag{4}$$
$$d_2 + d_4 + d_5 = b_5 \tag{5}$$
$$d_1 + d_3 + d_4 + d_5 = b_7 \tag{6}$$

Fig. 4. Example of buffer with 2 copies of 5 documents colliding in 3 buffer positions

If there are two matches in b_3, b_5 and three in b_7 (i.e., $M = 2$ for b_3 and b_5; and $M = 3$ for b_7); and the accumulative serial numbers R of b_3, b_5 and b_7 are 4, 7 and 9, respectively, then we know that d_2 and d_4 are zeros and we can solve the system of 3 linear equations with 3 unknowns.

This technique may still be applicable in cases where 2 or 3 keywords are searched, but given that the complexity grows exponentially with the number K of searched keywords, it quickly becomes infeasible to use it, even with a moderate K.

6.2   Tight Packing of Encrypted Lists and Bit Fields

In the previous section we described how we can use a single Paillier ciphertext to encode, space permitting, both the serial number of the document and the document itself. This encoding still allows for the homomorphic property; i.e., for the different plaintext parts of the message to be added together field-wise, when two ciphertexts are multiplied. This is a special case of a generic mechanism we can use to further improve the space-efficiency of the original Ostrovsky *et al.* scheme [9], the encoding of the Bloom filter buffer in Bethencourt *et al.* [1,2], and our decoding schemes.

A Paillier ciphertext can fully represent a plaintext of up to $\lceil \log(n) - 1 \rceil$ bits of length. In many cases (e.g., the first element of each buffer position in the Ostrovsky scheme, the representation of the serial number, or the Bloom filter entry) only a small plaintext is to be represented. We can do this by using only one Paillier ciphertext and packing as many elements as possible into it. Consider that we have two ciphertexts $E(i)$ and $E(j)$, representing two fields. We assume that the cryptographic protocols we shall perform will never result in a sum of those fields being greater than b bits long. We can then encode these two ciphertexts as:

$$E(i) \cdot E(j)^{2^b} = E(j \cdot 2^b + i) \tag{7}$$

It is possible to add a value to any element of the encrypted list. First, the ciphertext is shifted to the appropriate position and then multiplied to the tightly packed buffer. For example:

$$E(j \cdot 2^b + i) \cdot E(i') = E(j \cdot 2^b + (i + i')) \tag{8}$$

$$E(j \cdot 2^b + i) \cdot E(j')^{2^b} = E((j + j') \cdot 2^b + i) \tag{9}$$

As long as the sum of each element can be represented using only b bits (which is much smaller than the bits needed to represent the modulus n) we prevent the carry interfering with other fields in the ciphertext and achieve a significant gain in space efficiency.

7   Experimental Results

We present in Figures 5 and 6 simulation results[1] illustrating the performance of recursive extraction and solving equations for different sets of parameters.

Figure 5 illustrates how the recovery rate for recursive extraction changes, as the number of matched documents increases in a buffer of 100 places (figure 5(a)), and 1000 places (figure 5(c)). The recovery rate is defined as the fraction of documents retrieved from the buffer. When the number of matches is lower than half the size of the buffer, the recovery of all matching documents is virtually guaranteed. We plot the recovery rate for a number of documents that is greater than half the size of the buffer, in order to better illustrate the limits of our technique. We also plot the recovery rate for the original decoding scheme proposed by Ostrovsky *et al.*, in order to show the improvement offered by our techniques.

Figures 5(b) and 5(d) show the probability of success of our techniques based on solving equations, for buffers of length 100 and 1000, respectively. We show the probability of success for the general case (K searched keywords), and for the special case (one searched keyword). These "probabilities of success" have to be interpreted differently from the recovery rates of recursive extraction: whether a set of equations in a buffer can be solved is an all-or-nothing event – meaning that either all remaining unknown variables are extracted or none. Since solving equations can only be done *after* recursive extraction, these probabilities of success have to be seen as providing the possibility to get *all* documents (*in addition* to the documents already retrieved as shown in figures 5(a) and 5(c)).

Figure 6 illustrates the effect that different number of copies have on the recovery rate of the documents, for all techniques. We graph "measures of success" computed over numbers of matches that range from half the size of the buffer to slightly more than the buffer size (i.e., edge cases). For this reason, the graphs cannot be used to compare techniques, but rather the relative performance of the number of copies in each technique. As we expect, recursive extraction provides no advantage over the Ostrovsky scheme when only one copy of the document is used. Using three copies seems to be optimal for most cases (and hence we used it for our experiments shown in figure 5).

We observe that too few or too many copies, reduce the recovery rate of documents. In the first case not enough copies of the document are present to guarantee retrieval by ensuring [9] that at least one copy is alone in the buffer. In the latter case too many documents are inserted, which lowers the probability that any document is found on its own in the buffer. The original claim in [9], that recovery becomes better as the number of copies increases may also mislead implementers. This is only the case if one assumes a buffer of infinite size – and therefore an optimal parameter for the number of copies has to be calculated for each set of practical values of buffer size and expected matches.

---

[1] The full source code to produce all experiments and graphs is publicly available at http://homes.esat.kuleuven.be/~gdanezis/
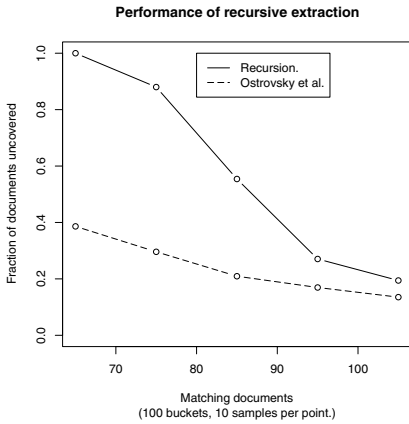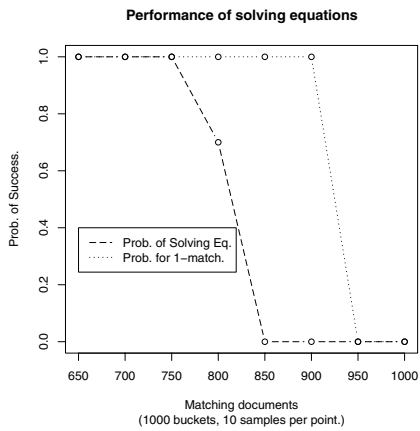
(a) Comparing Recursive Extraction vs. the original Ostrovsky *et al.* scheme for 100 buckets.
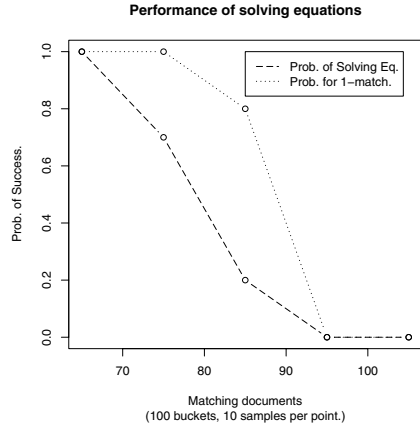
(b) The probability of success of solving simultaneous equations, and the special case of one keyword match, for 100 buckets.
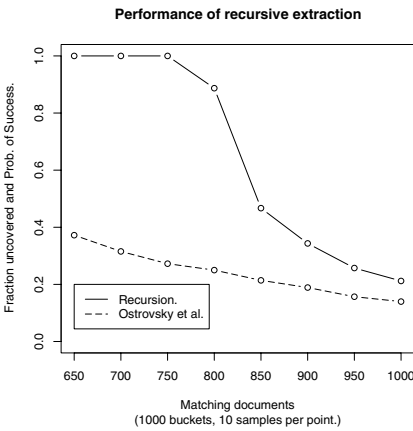
(c) Comparing Recursive Extraction vs. the original Ostrovsky *et al.* scheme for 1000 buckets.

(d) The probability of success of solving simultaneous equations, and the special case of one keyword match, for 1000 buckets.

Fig. 5. Performance evaluation of our techniques and comparison with the original scheme

8   Applications to Rateless Codes

Maymounkov and Mazires in [8] introduce "rateless codes", a method for erasure resistant, multi-source coding. These codes have been designed to be used in a

(a) Comparison for 100 buckets.   (b) Comparison for 1000 buckets.
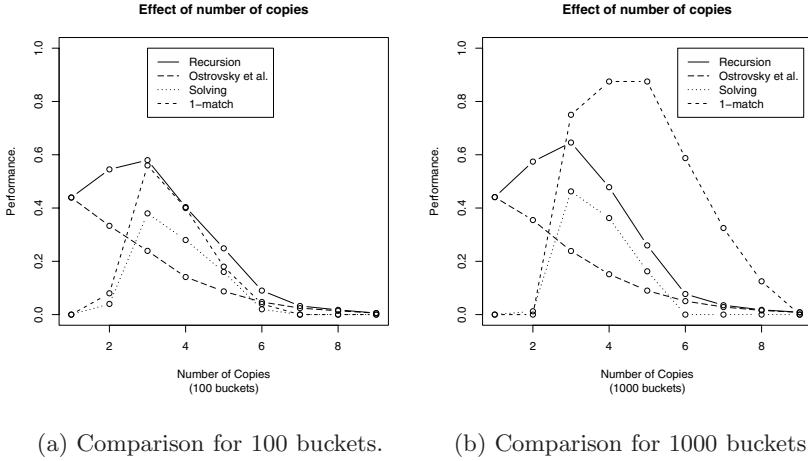
Fig. 6. The effect of the number of copies used on the performance of all techniques

peer-to-peer context, where Alice maybe downloading the same file from multiple sources, with no coordination between sources. The network is assumed to be unreliable and dropping packets transporting message blocks with some probability. We have already pointed out that the recursive decoding technique we use to decode the buffer resulting from private search is similar to the decoding strategy for such codes.

Many proposals for censorship resistant peer-to-peer systems [12] rely on peers storing encrypted files. We show that we can produce rateless codes from Paillier encrypted files, or using other homomorphic schemes like El-Gamal. Furthermore we can combine blocks received from multiple sources in order to retrieve the original file. Since Paillier encryption is probabilistic, the encrypted files on the different peers are unlinkable to each other for anyone not knowing the private decryption key.

We consider that Bob stores the file blocks F_i of a file, as Paillier encrypted ciphertexts, under the public key of Alice, i.e. $E_A(F_0), E_A(F_1), \ldots E_A(F_n)$. As in the original proposal, Bob generates the rateless code in two phases. First, the message is expanded into a composite message by appending auxiliary blocks. Each auxiliary block is the multiplication encrypted message blocks selected uniformly at random from all message blocks using a pseudo-random number generator (for more details see [7]). In the second phase, an "infinite" stream of check blocks can be generated. Those are the blocks to be transmitted to Alice, along with their serial numbers and all the seeds of the pseudo-random number generators used to generate them. Each check block is generated as follows: a random degree d is chosen using a pseudo-random number generator from a distribution ρ_d defined in [8]. A number d of blocks chosen uniformly at random using a pseudo-random number generator are then multiplied together to form

the check block. In both cases, due to the Paillier homomorphic properties, the product of the ciphertexts becomes the ciphertext corresponding to the sum of the plaintexts.

Bob sends check blocks to Alice, that decodes them. First of all, Alice decrypts the check block and recovers the sum of the auxiliary blocks from which it is composed. A similar algorithm as for the recursive decoding is then applied: check blocks that contain only one unknown auxiliary block are simplified by subtracting all the known blocks. As shown in [8], after a linear number of applications the message is recovered.

Allowing the computation of codes on encrypted data provides two key advantages:

– Alice can accept check blocks from two different sources, that are not in any way coordinating with each other, and use both streams to recover the original file. Yet, because of the randomized nature of Paillier, the different sources cannot know that the encrypted files correspond to the same source file.
– Alice knows when she has received sufficient blocks to recover the original message (from any number of sources), even if she is not able to decrypt the blocks and recover it. The mapping between auxiliary blocks and check blocks is determined only by the pseudo-random number sequences for which the seeds are known, and therefore she is able to tell when a decryption would be successful.

El-Gamal [6] encryption can also be used instead of Paillier, with certain advantages. In the El-Gamal variant ciphertexts are multiplied, which leads to the encryption of the product of the corresponding plaintexts. The decoder decrypts all blocks, as before, and divides (instead of subtracting) known blocks to simplify others.

A key observation is that the division necessary to reconstruct the original message can actually be performed on the encrypted check blocks, even without the knowledge of the secret key. As a result of this property of El-Gamal ciphertexts, not only messages encrypted blockwise using this cipher can be expanded into rateless codes served from multiple sources, but also the receiver of these blocks can perform the decoding and reconstruct a valid El-Gamal encrypted representation of the original message. The new representation of the message can in turn be rendered unlinkable to the two (or more) source representations by re-encrypting (also called self-blinding) the recovered ciphertexts.

9   Conclusions

We have presented in this paper efficient decoding mechanisms for private search. The very simple basic mechanism consists of recursively extracting documents from the returned buffer. The extended mechanism additionally solves equations in order to retrieve the remaining colliding documents in the buffer.

The size of the returned buffer with matched documents is the key to the success of private search schemes. If the size of this buffer is too long, any scheme can simply be reduced to transmitting back all the documents, which would save in complexity and cryptographic costs.

Our proposed decoding methods reduce by a significant constant factor the buffer sizes required by the Ostrovsky *et al.* Private Search [9] scheme. They require buffers less than twice the size of the matched documents, and a linear decoding complexity, meaning that the buffers at at least three times shorter than in the original scheme. Recursive extraction and solving equations are complementary and can be applied sequentially to extract a maximum number of documents from short buffers. Compression can be achieved by packing lists of values more efficiently, as presented in Section 6.2, further halving the base cost of the original scheme. In the important special case of matching documents that only contain one keyword, we achieve an overhead of less than 10%, making private search very practical.

We have presented simulation results to assess the decoding performance and estimate optimal parameters for our schemes.

Finally, we have shown how rateless codes can be constructed from encrypted data, while preserving the unlinkability of files across multiple sources.

Acknowledgements

References

1. Bethencourt, J., Song, D., Waters, B.: New constructions and practical applications for private stream searching (extended abstract). In: SP 2006. Proceedings of the 2006 IEEE Symposium on Security and Privacy, Washington, DC, USA, pp. 132–139. IEEE Computer Society Press, Los Alamitos (2006)
2. Bethencourt, J., Song, D., Waters, B.: New techniques for private stream searching. Technical report, Carnegie Mellon University (2006)
3. Bloom, B.H.: Space/time trade-offs in hash coding with allowable errors. Commun. ACM 13(7), 422–426 (1970)
4. Chaum, D.: The dining cryptographers problem: Unconditional sender and recipient untraceability. Journal of Cryptology 1, 65–75 (1988)
5. Danezis, G., Diaz, C.: Improving the decoding efficiency of private search. Dagstuhl Seminar on Anonymity and its Applications (October 2005)
6. Gamal, T.E.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (1985)
7. Maymounkov, P.: Online codes. Technical report, New York University (2003)

8. Maymounkov, P., Mazieres, D.: Rateless codes and big downloads. In: Kaashoek, M.F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735, pp. 247–255. Springer, Heidelberg (2003)
9. Ostrovsky, R., Skeith III, W.E.: Private searching on streaming data. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 223–240. Springer, Heidelberg (2005)
10. 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)
11. Sassaman, L., Cohen, B., Mathewson, N.: The pynchon gate: A secure method of pseudonymous mail retrieval. In: Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2005), Arlington, VA, USA (November 2005)
12. Serjantov, A.: Anonymizing censorship resistant systems. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, Springer, Heidelberg (2002)
13. Waidner, M., Pfitzmann, B.: The dining cryptographers in the disco: Unconditional sender and recipient untraceability with computationally secure servicability. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, Springer, Heidelberg (1990)

Cryptographic Securities Exchanges

Christopher Thorpe and David C. Parkes

School of Engineering and Applied Sciences
Harvard University
Cambridge MA 02138
{cat,parkes}@eecs.harvard.edu

Abstract. While transparency in financial markets should enhance liquidity, its exploitation by unethical and parasitic traders discourages others from fully embracing disclosure of their own information. Traders exploit both the private information in upstairs markets used to trade large orders outside traditional exchanges and the public information present in exchanges' quoted limit order books. Using homomorphic cryptographic protocols, market designers can create "partially transparent" markets in which every matched trade is provably correct and only beneficial information is revealed. In a cryptographic securities exchange, market operators can hide information to prevent its exploitation, and still prove facts about the hidden information such as bid/ask spread or market depth.

Keywords: Cryptography, market microstructure, securities exchanges.

1 Introduction

Market information plays a crucial role in modern securities exchanges. Published trades inform the public about the value of a particular security. Bid and ask quotations in limit books inform traders about other traders' interest in a security and at what prices orders are likely to be filled. Price change and trading volume information for equities track the (mis)fortunes and public awareness of corporations and equities markets. In theory, this market information should all benefit traders by forcing traders who have private information to disclose it via their trades. Unfortunately this information can also facilitate parasitic and unethical trading practices, and nondisclosure can itself lead to new exploits by market insiders who can then benefit their own accounts over investors' accounts. Balancing these forces is a significant challenge in market design, and homomorphic encryption techniques offer an attractive solution to this problem.

The application of homomorphic cryptography in other commercial protocols has been well studied in the academic literature (open and sealed-bid auctions [19,14,24], electronic cash [10], etc.) Yet, surprisingly little has been written about the contributions cryptography can make to securities markets, in particular the *open call auction* and *continuous double auction* protocols that underly most modern securities exchanges. Important prior work in this area includes Giovanni Di Crescenzo's pioneering work exploring privacy for stock markets [9], the secure double auction protocols Wang et

al. propose in [23], which employs homomorphic ElGamal encryption, and a "Secure Protocol to Construct Electronic Trading" described by Matsuo and Morita in [16].

The work of Bogetoft et al. in [5], based on secure multiparty integer computation, proposes an application to securities exchanges, although in their protocol, "all trade is executed at the same market clearing price" and orders that do not clear are rejected. In our case, we wish to support both market orders and the limit order book that is an integral component of modern financial markets. Less directly but nonetheless related, Szydlo [22] has proposed the application of homomorphic cryptographic commitments to the disclosure of stock portfolio holdings. Although some related work considers the privacy of trader identities, our work concerns only the revelation of quantitative information about trades not the anonymity of the traders, which we view as an orthogonal problem.

While the objective in most cryptographic work for auctions has been to hide information (secrecy), our objective is to enable a market designer to combine an appropriate level of *partial* transparency with provably correct behavior. We also do this in a setting informed by real-world demands, specifically, an exchange with both limit and market orders, and in which multi-party computation by all parties is infeasible.

Our design allows market designers to specify exactly what they wish to reveal, and reveal only that information while proving it, and the market operation, are correct. Immediate applications of our work can be seen in preventing unethical and parasitic trading practices in the major exchanges as well as providing for a means for trading large block orders without revealing information that can be exploited. Evidence for the need for information hiding in markets can be seen by recent SEC investigations and criminal convictions of unethical traders, and the development of new alternative trading systems (ATS's) that privately match large block trades. We detail how market information is misused and the securities industry's responses in Section 2.1.

In situating our work, we first discuss the role of information in securities markets from the perspective of *market microstructure*, a rich area of financial research that studies the role and exchange of information in markets and how market design principles serve to foster or inhibit information exchange. Market microstructure studies questions such as: What are the costs and benefits of transparency in financial markets? What determines the bid-ask spread for a particular stock? Do large orders really move the market? What is the effect of (not) publishing insider trades?

For simplicity, we will consider a single, *electronic clearing network* in which specialists, broker/dealers, or retail traders may place limit or market orders for shares of a particular equity (e.g. IBM stock) in a continuous double auction. We will explain the roles of each of these parties in the market, the types of transactions they may participate in, and why they do. We consider the various forms of information that these parties reveal through their actions (or inactions) and what information the markets reveal to them, and how they can profit from that information.

After considering the role of participants and information in our market, we construct a cryptographic framework that enables this information to be finely controlled and disseminated according to the specific rules established by a market operator. We observe that presently these types of information control are not achievable in financial markets because of a lack of trust: it is this transparency that proves *correctness* of the

market transactions. Yet requiring full *transparency* to achieve correctness is a blunt method that can be exploited. We decouple these considerations.

Our proposed system proves correctness and provides for any level of transparency; being able to prove facts without directly revealing the numbers behind them offers market designers a more expressive set of possibilities for reporting market status. Our construction and protocols use homomorphic cryptography [18,8,19,23,22] to prove the correct operation of the market according to its published rules and also to credibly reveal the required market information to the participants.

It is not our intention to advocate particular kinds of transparency but rather to offer a finer level of control to market designers. Indeed, this application of cryptography seems to us to open up interesting new questions for the field of finance. We conclude with worked examples and report the result of an initial analysis of the cost to support a realistic order flow on current hardware.

2   Introduction to Financial Markets

In this section we provide an overview of how equities are traded in order to motivate our contributions to those without a background in finance. The study of market microstructure in finance is most applicable to our work; Larry Harris' book *Trading & Exchanges* [12] is a well respected textbook on the field; we also found three recent survey papers of market microstructure [4,15,21] helpful in framing our contributions.

In many cases, we will simplify the complex workings of modern financial markets in order to illustrate the core principles that are relevant to our work. We clearly indicate these simplifying assumptions in our exposition. We use as our model a market for a single equity for a single company and assume that all trades in that market take place on an electronic clearing network (ECN) running a continuous double auction with an open limit order book. We assume that the market operates at fixed daily opening and closing times and trading does not take place anywhere else when the market is closed. For simplicity, we do not consider short sales or buying to cover, which are equivalent to selling and buying long positions for our purposes.

The market maintains an *order book* in which all outstanding limit orders are recorded. Depending on the transparency rules of the market, all, some, or none of the limit orders on the order book may be available to the public. Real-world exchanges (NYSE, NASDAQ, Chicago Board of Trade) offer various degrees of transparency for their order books.

For the purposes of the cryptographic properties of our exchange, there is no important difference between dealers, brokers, specialists, or investors. In our simplified model, everyone may post limit orders and has access to the same information.[1] Therefore we only consider two classes of participants: the (market) *operator*, i.e. the exchange or its agent, and *traders*, in which we include specialists, broker/dealers, and institutional and retail investors.

---

[1] A simplified way to look at it is that dealers, brokers and specialists provide liquidity to the market to support the trades investors want to make. Possibly the most important function of liquidity providers' use of limit orders is enabling investors to place market orders.

As we present our model, we introduce formal definitions that we use later in our protocol construction. We model the market state as the current state of the limit order book B and trade history H. The order book B is private, but the trade history H is public. (H can be public because any values logged therein are maintained in encrypted form.) The market operator also maintains a public, encrypted order book $\hat{B}$ that is equivalent to B except that all bids and quantities are encrypted. Each order placed receives a unique identifier i regardless of whether it is a bid or ask order which is associated with the order and its components. Ask and bid orders, a_i and b_i respectively, enter the market when placed and exit the market when withdrawn or executed. An order is the tuple $(p_i, q_i, t_i, s_i \in \{\mathtt{a}, \mathtt{b}\})$ representing the price, number of shares, the time the order was placed on the market, and the side of the market: whether the order is an ask (sell) or a bid (buy). When an order is taken off the order book, it is removed from the state of the order books B and $\hat{B}$ and the history H is updated with the execution or cancellation that resulted in its removal.

We will also refer to a function with access to a complete price ordering of the orders on the market $o(s \in \{\mathtt{a}, \mathtt{b}\}, rank)$ whose arguments are the side of the market (ask or bid) and the order's $rank$ where the most competitive price on either side has $rank = 0$. Its output is the unique identifier i for the ask or bid with the given $rank$. For example, we might write the current bid/ask spread as $p_{o(\mathtt{a},0)} - p_{o(\mathtt{b},0)}$, or the market depth (measured in shares) of the most competitive ten bid orders as $\sum_{r=0}^{9} q_{o(\mathtt{b},rank)}$. This ordering is maintained by the market operator; it is convenient for showing how the market operator proves correct operation of the market. Obviously, the ordering $o(s, r)$ changes whenever an order enters or exits the market. This function is also used to support the market invariant that all orders are maintained in strict priority order as described in Section 3.

In modern equities markets, orders fall into two basic categories: *market* orders are an instruction to buy or sell a specific quantity of a security, and are filled as soon as possible at the best available price on the market; *limit* orders are an instruction to buy or sell a specific quantity of a security at a specific price, and are filled only when another participant in the market is willing to make the opposite trade.

More complex orders that use real-time market information are possible, depending on broker support; for example, a *stop loss* order at a particular price instructs the broker to sell a position at the market when the market reports a trade at or below that price.[2] In practice, some traders also use orders based on real-time data as a substitute for limit orders because of the information revealed by limit orders or to get a better price; for example, an order such as "Buy 1,000 shares at the market if there are any trades below $20.00" might be used instead of a limit order to "buy 1,000 shares at $19.99" in order to keep the trader's intentions secret and potentially get a better price if the stock price dropped sharply. It might be that in a partially transparent market in which limit order prices are hidden, traders would be more inclined to use limit orders in these cases.

---

[2] Limit orders cannot substitute for stop orders. Limit orders are persistent, and less competitive than the current equilibrium price; stop orders react to market movements and are at *more* competitive prices. Our framework can be extended to support stop orders with concealed prices; the operator would maintain a side list of stop orders and prove when their target prices are reached by executed trades, all without revealing either the trade price or stop order price.

2.1   Market Information and Its Misuse

In this section we explore how transparency can be exploited, then examine at a high level the types of market information whose transparency may be regulated by cryptographic systems.

Misuse of Market Information.   The information provided by transparency can be exploited by unethical or creative traders. To illustrate this hidden cost of transparency, we detail two common practices, one unethical and the other "parasitic": front-running and penny-jumping, respectively. Larry Harris' chapter "Order Anticipators" in *Trading & Exchanges* explores these and other related practices in depth [12]. We speculate that these exploitations of transparency may be part of the cause for the conflict between published theoretical market microstructure results that show transparency should improve liquidity and other empirical results that are ambiguous with respect to this question [20].

"Front-running" is the unethical practice of a party with private information about an incoming large order to the market running in front of that order to take a position in the hope of making a quick profit when the large order arrives. For example, a trader knowing that a mutual fund is going to buy a \$10M position in IBM stock might buy a smaller position beforehand with the expectation that the mutual fund's purchase will drive the price higher. Front-running and allegations of it are widespread. In 2001, Dreyfus agreed to pay \$20.5 million to settle accusations that their fund manager Michael Schonberg engaged in front-running [1]. In 2003, the NYSE announced its Enforcement Division had investigated several specialist firms for rule violations including front-running and decided to bring disciplinary action against them. Seven specialist firms agreed to pay over \$200 million to settle charges brought as a result of these allegations [2]. In July 2006 a Manhattan jury convicted former specialist firm Van der Moolen managers Michael Stern and Michael Hayward of fraud for trading stocks on the firm's account before filling clients' orders in order to boost Van der Moolen's profits and their own compensation [7]. In early 2007, *The New York Times* reported other allegations of front-running: "The [SEC] has begun a broad examination into whether Wall Street bank employees are leaking information about big trades to favored clients. . . " [3].

"Penny-jumping" is not illegal, but often described as "parasitic". The practice extracts value from the market without contributing information. Specifically, a trader identifies a large limit order on the order book (e.g. 10,000 shares at \$25.00) and places a smaller limit order one tick above that order (e.g. 1,000 shares at \$25.01). The penny-jumper's order will be filled first, and he expects that his upside is greater than his downside, because his downside is protected by a free trading option created by the large limit order. If the market is random, it is likely that the stock will trade at a higher price before the large order is filled. If the price happens to decline before it increases, the large order will be filled, and the penny jumper exits his position via the large order for a one-tick loss (e.g. after 8,000 of 10,000 shares have been filled).

One response to concerns of unethical and parasitic practices is to construct a market in which only partial information is reported. But it is unclear whether investors would trust that information's correctness or trust the market operators not to benefit from any

private information. If prices are hidden, an unscrupulous market operator could simply fill a favored party's bid before higher bids. Indeed, regulators have begun to mandate transparency to protect investors, in light of specialists and broker/dealers exploiting private market information to their advantage [21,15,2,1,7].

There is also evidence that knowledge of large ("block") trades is similarly exploited. According to Stoll [21], large blocks of stock are not sent to the open market because "The risk of pre-trading portions of the block in this manner is that other traders will become aware of the block and will sell in anticipation, perhaps driving the price down..." and because other traders can exploit knowledge of large orders in other ways (as above). While Stoll further claims that "empirical evidence of block trades is quite mild," Keim and Madhavan [13] (as cited in [15]) find in an empirical study that the average (one-way) price impact for a seller-initiated transaction is -10.2% from a benchmark three weeks before a large block trade, after adjustment for market movement.

Historically, block trades are performed in "upstairs markets" where brokers shop around for the best deal. Keim and Madhavan "attribute this large price impact to information 'leakage' arising from the process by which large blocks are 'shopped' in the upstairs market." [15] The reason for hiding information in block trades is mainly to protect the traders before the large transaction occurs.[3]

In September 2006, a number of major banks announced a response to the block trading problem with two new so-called ATS's (Alternative Trading Systems). Citigroup, Goldman Sachs, Lehman Bros., Merrill Lynch, Morgan Stanley and UBS also announced a "Block Interest Discovery Service" (BIDS) for automatically matching large block orders without revealing them to the primary markets. Another ECN, *Liquidnet*, specializes in institutional large block trades and has captured a small but significant share of order flow: as of 30 June 2006 they handled over $175 billion (notional) of trades with an average value of $1.42 million on US equities, according to their website. This is clear evidence that institutional investors are dissatisfied with traditional market support for large block trades.

While these approaches help to limit the exploitation of information they do not provide any correctness guarantees and moreover any published quotations can be exploited as before. These approaches also require that the block trades be separated from the primary securities exchanges. This could have a significant impact on liquidity and overall market efficiency. With our solution, quotations from the primary market can be integrated into our order book and matched against standing block trades; all transactions can be matched by a single efficient marketplace.

2.2   Developing a Cryptographic Securities Exchange

Our model is of a simple securities exchange in which a market operator keeps a private order book B and publishes its public analog $\hat{B}$ with encrypted prices and quantities, and (optionally encrypted) history of its actions H. Incoming limit orders are placed on the book or matched with existing limit orders; incoming market orders are matched with limit orders; the operator proves its actions correct.

---

[3] Gemmill [11] offers an empirical analysis consistent with this view of the effects of post-trade reporting of block trades on the London Stock Exchange. He finds *ex post* disclosure of block trades does not have a dramatic effect on liquidity.

Our primary goal is to prevent various adversaries from exploiting information present in limit order books to the detriment of traders who wish to place limit orders. These adversaries primarily include other traders and market insiders (market makers, specialists, exchange employees) who attempt to (unethically or parasitically) profit by exploiting limit order information.

Rindi [20] uses the term "partial transparency" in her examination of three regimes of pre-trade transparency in a market for a risky asset based on an open limit-order book: "under full transparency agents can observe the order flow and traders' personal identifiers; under partial transparency they can observe the order sizes and under anonymity they can only observe the market price."

We consider each of these information classes in turn. For existing orders, the type of the order is implied; if it is in the book, it is a limit order. Incoming orders may be market or limit orders; we assume that is disclosed. In call auctions, the transaction type (buy or sell) can be kept secret until the auction closes, but it is not meaningful to hide whether an order is to buy or sell in continuous double auctions. As noted before, timed expiration of orders is unimportant.

The price per share p_i associated with an order a_i or b_i on the book may be fully transparent ($p_i =$ \$20.06), partially transparent (\$20.00 $\leq p_i \leq$ \$20.25), or kept completely private ($p_i =$?). Similarly, the quantity q_i and time posted t_i may be fully, partially, or not transparent.

The parameters of multiple orders may be related by inequalities. Two orders may be related by price (e.g. $p_i \geq p_j$), quantity (e.g. $q_i = q_j$) or time posted (e.g. $t_i < t_j$). Partial or complete orderings for price and time of all orders in a limit book can be constructed using these methods, as will become important for more expressive partially transparent revelations. Quantity becomes important when proving order flow and correct execution of trades.

Finally, one might wish to prove information about linear functions on the parameters of multiple orders, or compute linear functions without revealing unnecessary additional information about the orders themselves. Examples of these functions include:

- Bid/ask spread between the two most competitive orders
- Market depth within p cents of the mean between the outstanding bid and ask (measured in number of shares)
- Bid-ask spread between the two least competitive orders comprising a market depth of q shares
- Prices (if any) at a market depth of q shares
- Average number of hours outstanding orders above price p have been on the market

Using recent advances in homomorphic encryption, market designers can construct markets in which this kind of information can be revealed and proved correct without revealing additional information about underlying orders.

Information, and related proofs, need not be issued in real-time, and in fact in many cases market designers may prefer delayed revelation. In our system, market designers can decide exactly when to reveal market activity, and even construct different disclosure rules for different trade sizes. For example, the market might disclose small trades within 30 seconds and large trades within 1 day.

3   The Cryptographic Securities Exchange

We have described the model of our market as a limit order book with a history. We consider the state of the order book B, the encrypted public order book $\hat{B}$, and the history H to be the core state of our market. Various actions by the participants in the markets update this state. We formally define these actions, who may perform them, and how the update the state of the market depending on its state. The order book and history begin as empty states.

In our present model we maintain an important invariant in B and $\hat{B}$: all orders are maintained in a strict priority ordering as defined by the ordering function $o(s, rank)$. Despite regulations that prescribe order routing priority, the priority of trades within active markets is a complicated process beyond the scope of the present work. For example, smaller orders at slightly less competitive prices or more recently submitted might be filled instead of a large order that is the longest standing at the most competitive price.

We model these priority rules as follows, from highest to lowest:

1) Most competitive price (p_i is maximal)

2) Longest standing (t_i is minimal)

3) Best "fill", measured by the percentage of shares filled of the larger of the two orders ($\frac{|q_i - q_j|}{\max(q_i, q_j)}$ is maximal).

We do not consider a formal mechanism for proving the time priority of an order correct, in part because we see no benefit in encrypting the timestamp of an order: orders are posted when they arrive, and that reveals the time they were posted. Further, this information is not readily exploitable.

We assume a bulletin board that orders are posted to; the market operator is required to accept new orders by adding them to the history H as soon as they arrive. We also assume that at the beginning of each new trading session the public, encrypted order book $\hat{B}$ has been verified by tracing through the previous day's history in H.

3.1   Assumptions

Our protocol rests on certain realistic assumptions. The operator and all traders possess the means for generating secure digital signatures. A universal, tamper-resistant clock must be accessible by all parties, such as that maintained by the US NIST, to preserve the integrity of timestamps. To prevent the operator from improperly failing to disclose instructions, there is a universally accessible bulletin board—not maintained by the operator—that records all activities of all parties and publishes them for anyone to see.[4] (All private data remain secure by encryption.) We assume the hardness of the composite residuosity problem supporting Paillier's homomorphic encryption scheme [18]. We assume that a computer network may be monitored for activity, and that even large amounts of activity can be examined for any information "leakage".

---

[4] We assume a bulletin board strictly separate from the operator so that traders' orders may be presumed received and posted on time without respect to their content. Because the operator can decrypt incoming orders, it is important that all incoming orders be posted by a neutral third party to require the operator to prove its actions are correct; a corrupt operator could delay or ignore incoming orders to benefit favored traders.

3.2   Encryption Method

We employ the homomorphic encryption scheme described by Pascal Paillier [18] and extensions published by Damgård and Jurik [8], Parkes et al. [19], and a use of Boudot's efficient range proofs [6]. We write the encryption of a value m with the market operator's public key and random help value r as $E(m, r)$. The properties of this cryptosystem allow construction of mathematical proofs of certain facts over the ciphertexts. For example, given only $E(m_1, r_1)$ and $E(m_2, r_2)$, one can prove a value is within a constant range, e.g. $m_1 < n/2$; inequalities, e.g. $m_1 > m_2$; or generate new ciphertexts that are the sum of others, e.g. $E(m_1 + m_2, r_1 \cdot r_2) = E(m_1, r_1) \cdot E(m_2, r_2)$. We require these primitives for proving the correct operation of the market.

3.3   Processing Incoming Orders

Before orders arrive in any trading session, we recall that we assume the operator has proven the public, encrypted order book $\hat{B}$ correct by reference to the orders posted on the bulletin board in previous sessions. This means that all transactions may be performed with respect to existing orders in the order book without need for further proofs of their correctness or rank in the order book.

Limit Orders.   Any trader in our model may place a limit order according to the following protocol. Each limit ask order a_i is given a unique id i by the bulletin board and enters the market in the following manner. Note that the same method applies for bid orders b_i by interchanging "ask" and "bid" and reversing inequalities ($<$ becomes $>$).

Step 1. The trader encrypts the price p and quantity q and sends $(E(p, r_p), E(q, r_q), \mathtt{a})$ to the bulletin board. The bulletin board creates a unique identifier i, adds a timestamp t_i based on the current clock, publishes $\hat{a}_i = (E(p_i, r_{p_i}), E(q_i, r_{q_i}), t_i, \mathtt{a})$, computes the digital signature $SIGN_{BB}(\hat{a}_i)$ and both publishes it and sends it to the trader as a receipt. Only the operator can see what the p_i and q_i are.

Step 2. The trader privately sends the random help values r_{p_i}, r_{q_i} to the operator[5]

Step 3. The operator privately decrypts the values in $\hat{a}_i$ to compute $a_i = (p_i, q_i, t_i, \mathtt{a})$, and verifies that the random help values correspond to the ciphertexts provided.

Step 4. The operator logs in H that order a_i was received at time t_i.

Step 5. The operator compares p_i to the best ask price, $p_{o(\mathtt{a},0)}$ and the best bid price, $p_{o(\mathtt{b},0)}$ and proceeds in one of four ways:

-   If the incoming ask order is priced at less than or equal to the highest priority bid, i.e. $p_i \leq p_{o(\mathtt{b},0)}$, the operator matches a_i with all outstanding bid orders whose prices are $\geq p_i$ up to the quantity q_i in order of priority. If there are not enough to fill a_i, it becomes the most competitive ask order on the order book afterward.

---

[5] This is required to prevent other traders from exploiting the malleability of the homomorphic encryption scheme to submit bids based on a function of another trader's bid, e.g. "his bid plus 10 cents." Knowing the random help value implies knowing the decryption, so provided the cryptosystem is secure and the random help values are secret, no trader can submit a correct random help value for a ciphertext based on another trader's encrypted values.

- If the incoming ask order is priced between the highest bid and the lowest ask price, i.e. $p_{o(\mathtt{b},0)} < p_i < p_{o(\mathtt{a},0)}$, the operator adds it to the order book.
- If the incoming ask order is priced equal to the lowest ask price, i.e. $p_i = p_{o(\mathtt{a},0)}$, the operator adds it to the order book.
- If the incoming ask order is priced higher than the lowest ask price, i.e. $p_i > p_{o(\mathtt{a},0)}$, the operator adds it to the order book.

Step 6. The operator updates H on the bulletin board with the details of any trade that resulted from receiving a_i.

Step 7. The operator recomputes the ordering function $o(s, rank)$ such that the rank of all orders in B is defined and correct.

Step 8. The operator updates its private B and publishes $\hat{B}$ on the bulletin board with the new set of encrypted orders.

Step 9. The operator issues proofs of correctness of its actions on the bulletin board. Specifically, it proves the necessary inequalities to pigeonhole the incoming limit order a_i in its proper priority ordering, maintaining the invariant that the all outstanding orders in B and $\hat{B}$ are ordered according to priority.

Step 10. Anyone who wishes may verify the operator's public proofs.

Market Orders. A trader in our model may also place a market ask order a_i (or bid b_i). The protocol differs from the limit order protocol given above only in Step 5:

Step 6. The operator matches the incoming market ask order a_i with the k highest priority bid orders $b_{o(\mathtt{b},0,\ldots,k)}$ such that the $k-1$ highest bids do not fill a_i but k do, and executes the trade(s) on all matched orders.

Executing Trades on Matched Orders. The operator must prove that the quantity of the k multiple limit orders a large order is matched with is greater than or equal to the quantity of the market order, and that the sum of the quantities of the most competitive $k-1$ limit orders is strictly less than the quantity of the market order.

Two orders a_i and b_j are matched when the bid price meets or exceeds the ask price, i.e. $p_j \geq p_i$. If the quantities are equal, $q_i = q_j$, the trade is executed and both orders are removed from the order books B and $\hat{B}$ and the transaction is logged in the history H. Formally, to log the transaction the operator adds a journal entry to H $h_{i,j} = (\hat{a}_i, \hat{b}_j, t_{i,j})$ with its signature $SIGN_{MO}(h_{i,j})$. The time $t_{i,j}$ is the time reported by the universal clock at the time the order was executed. The operator also posts the following proofs on the bulletin board:

- A proof that $p_j \geq p_i$ given $E(p_i, r_{p_i})$ and $E(p_j, r_{p_j})$.
- A proof that $q_j = q_i$ given $E(q_i, r_{q_i})$ and $E(q_j, r_{q_j})$.

If the quantities differ, the order for fewer shares is fully filled and the order for more shares is partially filled. Then, the smaller order (w.l.o.g. a_i) is removed and the larger order (w.l.o.g.) b_j's quantity is updated in the order books B and $\hat{B}$. Formally, the entry b_j in B is replaced with $b_j = (p_j, (q_j - q_i), t_j, \mathtt{b})$, and in $\hat{B}$ with $\hat{b}_j = (E(p_j, r_{p_j}), E(q_j, r_{q_j})/E(q_i, r_{q_i}), t_j, \mathtt{b})$. Anyone can verify the correctness of the new published $\hat{b}_j$ by computing the quotient of the previously published encrypted values $E(q_j, r_{q_j})$ and $E(q_i, r_{q_i})$, which is known to be an encryption of their difference. The

transaction is logged in the history H as above with a similar journal entry $h_{i,j} = (\hat{a}_i, \hat{b}_j, t_{i,j})$ and signature $SIGN_{MO}(h_{i,j})$. The operator also posts the following proofs on the bulletin board:

- A proof that $p_j \geq p_i$ given $E(p_i, r_{p_i})$ and $E(p_j, r_{p_j})$.
- A proof that $q_j > q_i$ given $E(q_i, r_{q_i})$ and $E(q_j, r_{q_j})$. This is done by showing that $(E(q_j, r_{p_j})/E(p_i, r_{p_i})) \cdot E(-1, 1)$ is the encryption of a value $(q_j - q_i - 1) < n/2$. (This proves that no wraparound occurred; we subtract 1 from $q_j - q_i$ to prove a strict inequality.)

One minor issue in a market without transparent prices is that a limit order may be submitted to the market that is more competitive than it needs to be to clear. For example, a trader might post a new limit order to sell at $20.05 when there is a standing order to buy at $20.09. In transparent markets, this would obviously never happen except in cases of error. Choosing the clearing price for such situations is a matter of market design. With the primitives we have described, it is possible to prove correct a clearing price based on the standing order's price, the incoming order's price, the mean of the two (within one tick), or indeed any linear function of the two prices, without revealing the price itself or any information not implied.

Once two orders are matched and the proofs posted, a clearing agent will be responsible for transferring the ownership of the shares at the correct settlement price. The market operator will send the clearing agent the random help values necessary to verify the correctness of the execution price and number of shares from the history posted on the bulletin board. The agent then verifies the trade and settles it.

In addition to sending information to the clearing agent, any information published about the state of the market is proven at this point on the bulletin board. For example, the auctioneer might reveal the random help values associated with the determined clearing price and matched quantity to provide "last trade" tick data, or update proofs of market depth, bid/ask prices, etc. Typically the "market price" of a security for any period is the price at which it was last traded during that period; thus, publishing provably correct market prices is straightforward.

3.4   Post-Trade Reporting

The market operator can report clearing prices by revealing the random help values of the encrypted orders in the history H after any specified delay. Immediate revelation may be a problem in the event a partial fill is revealed and the remainder is still on the market: is price is now public. Facts similar to those provable for limit orders may be proven about trades after the fact, for example, volume, average price, closing price, etc. Post-trade transparency is as easily controlled by market designers as transparency during other phases of market activity, and we leave the question of appropriate reporting rules open for this reason.

3.5   Adversaries and Attacks

The adversary we are most concerned about in this work is the unethical or parasitic trader who exploits (presently public) market information for profit in a way that discourages placement of limit orders. A secondary class of adversary is a dishonest market

operator who may attempt to profit by exploiting the now private market information via trading or disclosure for compensation. We do not consider as adversaries parties with private information external to the market's operation, such as employees with proprietary information about traded companies.

Traders. We first consider attacks by parties who do not possess any insider access to the market operator or its systems. These traders may either attempt to circumvent the cryptographic security of the system or exploit the information provided in new ways. Provided cryptographic keys of adequate security are chosen to prevent a brute-force attack, cracking the encryption scheme itself is believed to be intractable under the Decisional Composite Residuosity Assumption described in Paillier's work [18].

The semantic security of the probabilistic Paillier cryptosystem protects the encrypted values against chosen plaintext attack. (For example, using a deterministic encryption of prices would be insecure, because an adversary could try all realistic prices and identify the values.) Paillier's scheme is not secure against an adaptive chosen ciphertext attack; indeed, the malleability of the scheme that enables the homomorphic properties we employ implies this insecurity. However, mounting a successful chosen ciphertext attack against our protocol does not seem a significant threat, as the only way a value can be decrypted is in the event someone is willing to trade it. Thus, any party attempting to gain information by submitting a chosen ciphertext as information must also be willing to execute any trades from that information.

We have not identified any additional parasitic trading practices that could be employed using a cryptographic securities exchange. Since we are not adding any information into the marketplace – only allowing designers to restrict information – we believe that there are no new exploits that would not be possible in an ordinary market with an open limit book.

This said, we reiterate that some parties may attempt to gain information from the marketplace by placing orders. For example, one could discover the price for the most competitive ask order by placing an order to buy one share at the market. Alternatively, a trader might place limit orders at various prices to see where they fit into the order book, in order to gain information about the price points, and then retract them. However, no trader may observe anything about the market without fundamentally changing the market: a "probe" share purchased revealed the price *for that share only*, and afterward, the number of shares at that price remains unknown and becomes smaller; probe limit orders enter the market and always bear the risk of being executed.

Several solutions to this problem come to mind. First, at a significant but tractable complexity cost, the marketplace could maintain not a strict ordering over all orders, but a partial ordering in which only the minimum information required to prove correctness is revealed. Thus incoming orders that were not competitive (and likely to be filled) would be proven only to be less competitive than the most competitive order. This would significantly limit the ability of a trader to count trades above a particular price by placing limit orders. Second, the market operator or market makers could place random numbers of zero-quantity limit orders on the marketplace so that there would be a large number of orders at every price point. Third, market designers could limit such exploitative practices by limiting order frequency, sizes, or specifying a minimum duration on the market.

The Market Operator. A more insidious attack is if a dishonest market operator, possibly in collusion with another trader, exploits its valuable private information or gives preference to particular traders. We recall our assumption of a bulletin board operated by a third party to prevent the market operator from discarding dispreferred orders, or delaying their publication until after preferred orders are listed. With this, an unscrupulous market operator cannot issue valid proofs of correctness of matched trades, but he could still selectively reveal information to preferred traders. We reiterate that despite this implied trust in the market operator, our architecture provides for two improvements over existing markets: information can be specifically controlled and is possessed by only one party (instead of the entire market), and the market operator may not manipulate the market by front-running or matching orders on any basis other than the published rules.

That said, the partial trust of the operator is a strong assumption, and solutions to enhance that trust merit discussion. One answer is to distribute the trust in the market operator among a group of parties, similar to the approach Bogetoft et al. describe [5]. This may be challenging from a business perspective but nonetheless possible. Another solution involves careful network, hardware and software security, employing special purpose hardware (e.g. that used in Trusted Computing architectures) that only runs software approved and signed by a third party, and monitoring all network traffic to detect any communications that might leak information.

4   Example Order Book and Transactions

This section describes incoming orders and how trades are identified and executed. Table 1 shows a sample order book B. The public, encrypted order book $\hat{B}$ is equivalent, except that the quantities and prices are encrypted. **R** indicates $rank$. Orders are always ranked in priority order. Each order's rank is defined according to the priority rules outlined above (best price, oldest) and randomly selected in the case of a tie.

We first consider an incoming market order to purchase 700 shares of the stock. The trader constructs $\hat{b} = (\_, E(700), \_, \mathtt{b})$ and posts it on the bulletin board. The bulletin board assigns ID $i = 25$ and timestamp $t_i = 09{:}44{:}32$ and publishes $\hat{b}_i = (\_, E(700), t_i, \mathtt{M})$. For clarity, we will use i for the ID of each incoming order in the following text to more clearly distinguish it from the limit orders.

The market operator sees the market order on the bulletin board, decrypts $\hat{b}_i$ to $b_i = (\_, 700, t_i, \mathtt{b})$, and matches two trades $(a_1 4, a_1 2)$ to fill the order. It adds journal entries to the history H and publishes proofs on the bulletin board:

- $H \leftarrow h_i = \hat{b}_i$
- $H \leftarrow h_{14,i} = (\hat{a}_{14}, \hat{b}_i, t_{14,i} = 09{:}44{:}33)$
- $H \leftarrow h_{12,i} = (\hat{a}_{12}, \hat{b}_i, t_{12,i} = 09{:}44{:}33)$
- Proofs of correct quantities: $q_{14} + q_{12} \geq q_i$ and $q_{14} < q_i$
- Sufficient proof of priority: $q_{12} < q_{13}$

Table 1. Order Book B_1

R	ID	Time	Qty	Ask
3	11	09:34:42	2500	$20.13
2	13	09:39:23	500	$20.10
1	12	09:39:23	300	$20.10
0	14	09:41:06	600	$20.09

R	ID	Time	Qty	Bid
0	22	09:37:14	1000	$20.05
1	24	09:43:42	500	$20.02
2	23	09:41:23	800	$20.00
3	21	09:30:06	1700	$19.96

The operator then updates B (and $\hat{B}$) by removing order a_{14} and updating $q'_{12} = 300 - (700 - 600) = 200$ (and $\hat{q}'_{12} = E(q_{12})/(E(q_i)/E(q_{14}))$). Anyone can verify that the updated encrypted quantity $\hat{q}'_{12}$ is correct by comparing it with functions of the quantities of the other orders.

In a second example, a trader posts a new limit ask order $\hat{a} = (E(\$20.03), E(1200), \_, \mathtt{a})$ to which the bulletin board assigns $i = 15, t_i = 09{:}46{:}02$. The market operator sees it, decrypts it, and concludes it is more competitive than the most competitive bid. He adds journal entries to H, removes

Table 2. Order Book B_4

R	ID	Time	Qty	Ask
3	11	09:34:42	2500	\$20.13
2	13	09:39:23	500	\$20.10
1	12	09:39:23	200	\$20.10
0	15	09:46:02	200	\$20.03
R	**ID**	**Time**	**Qty**	**Bid**
0	24	09:43:42	500	\$20.02
1	23	09:41:23	800	\$20.01
2	26	09:50:33	200	\$19.98
3	21	09:30:06	1700	\$19.96

$b_{o(\mathtt{b},0)}$, matches a_i with b_{22} and adds the remainder a'_i to B and $\hat{a}'_i$ to $\hat{B}$, preserving the priority order invariant, and publishes:

- $H \leftarrow h_i = \hat{a}_i$
- $H \leftarrow h_{i,22} = (\hat{a}_i, \hat{b}_{22}, t_{i,22} = 09{:}46{:}04)$
- Proof of correct quantities: $q_i > q_{22}$
- Proof of price position: $p_i \leq p_{22}, p_i > p_{24}$
- Proof of clearing price (as required)

In a final example, a trader posts a limit bid order $\hat{b} = (E(\$19.98), E(400), \_, \mathtt{b})$ to which the bulletin board assigns $i = 26, t_i = 09{:}50{:}33$. The market operator sees it, decrypts it, and places it in the order book in the appropriate position It adds a journal entry to H, adds the order b_i to B and $\hat{b}_i$ to $\hat{B}$, preserving the priority order invariant, and publishes $H \leftarrow h_i = \hat{b}_i$ and the proofs of priority $p_i < p_{23}$ and $p_i > p_{21}$. The order book is now as shown in Table 2.

5 Conclusions and Future Work

Clearly, providing controllable transparency of market information in securities exchanges together with proofs of correctness (both of information and of the market operation) is an important application of homomorphic cryptography. The protocol presented here is simple to understand, closely related to existing financial market protocols, and does not rely complex cryptographic primitives that might discourage its use among traders. Finance research has already started to study the implications of different levels of partial transparency, seeking to ensure liquidity and limit exploitation. Cryptography can be used to prove correct operation according to specified rules even under partial transparency.

We envision a broad range of future work based on the protocol we have presented and similar ideas. For instance, market designers might want support for more expressive order types, such as fill-or-kill, immediate-or-cancel, order-cancels-order, or stop orders maintained by the market. Our protocol could also easily be extended to open call auctions or periodic clearing models (such as POSIT). The market operator might wish to prove a less revealing ordering of the limit orders in the order book. Support for other specialists and liquidity providers' functions could be added by selective revelation.

Other more creative exchanges are possible in our setting. For example, integrating other ECN's with a cryptographic securities exchange may be of particular use in bridging the gap between block trades and ordinary securities trading. Cryptographic derivative markets for options and indices whose prices are tied to the activity in underlying securities' order books are another important possible extension of our work.

We have conducted an initial empirical analysis of the computation cost for running such a system, and arrived at a conservatively high estimate of 5 cents (US) to place and verify an order. Our experiments used a low end, dual Pentium IBM x-server with no special cryptographic hardware. This is inexpensive enough to be feasible in practice, although we leave a full efficiency analysis, perhaps in conjunction with a prototype, to future work.

Acknowledgments

References

1. Dreyfus will pay $20.5 million to settle lawsuit: The New York Times (June 22, 2001)
2. Settlement reached with five specialist firms for violating Federal securities laws and NYSE regulations: U.S. SEC Press Release (2004), http://www.sec.gov/news/press/2004-42.htm
3. Anderson, J.: S.E.C. is looking at stock trading. The New York Times (February 6, 2007)
4. Biais, B., Glosten, L., Spatt, C.: Market microstructure: a survey of microfoundations, empirical results and policy implications. Journal of Financial Markets 8(2), 217–264 (2005)
5. Bogetoft, P., Damgård, I., Jakobsen, T., Nielsen, K., Pagter, J., Toft, T.: A practical implementation of secure auctions based on multiparty integer computation. In: Di Crescenzo, G., Rubin, A. (eds.) FC 2006. LNCS, vol. 4107, Springer, Heidelberg (2006)
6. Boudot, F.: Efficient proofs that a committed number lies in an interval. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 431–444. Springer, Heidelberg (2000)
7. Bray, C.: Two ex-Van der Moolen specialists are convicted of securities fraud. The Wall Street Journal (July 15, 2006)
8. Damgård, I., Jurik, M.: A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In: Kim, K.-c. (ed.) PKC 2001. LNCS, vol. 1992, pp. 119–136. Springer, Heidelberg (2001)
9. Di Crescenzo, G.: Privacy for the stock market. In: Syverson, P.F. (ed.) FC 2001. LNCS, vol. 2339, p. 269. Springer, Heidelberg (2002)
10. Frankel, Y., Tsiounis, Y., Yung, M.: "Indirect Discourse Proofs": Achieving efficient fair off-line E-cash. In: Kim, K.-c., Matsumoto, T. (eds.) ASIACRYPT 1996. LNCS, vol. 1163, Springer, Heidelberg (1996)
11. Gemmill, G.: Transparency and liquidity: A study of block trades on the London Stock Exchange under different publication rules. Journal of Finance 51, 1765–1790 (1994)
12. Harris, L.: Trading and Exchanges. Oxford University Press, Oxford (2003)
13. Keim, D.B., Madhavan, A.: The upstairs market for large-block transactions: Analysis and measurement of price effects. Review of Finacial Studies 9, 1–36 (1996)

14. Lipmaa, H., Asokan, N., Niemi, V.: Secure Vickrey auctions without threshold trust. In: Blaze, M. (ed.) FC 2002. LNCS, vol. 2357, pp. 87–101. Springer, Heidelberg (2003)
15. Madhavan, A.: Market microstructure: A survey (March 8, 2000)
16. Matsuo, S., Morita, H.: Secure protocol to construct electronic trading. IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Sciences E84-A(1), 281–288 (2001)
17. Paillier, P.: Cryptographie à Clé Publique Basée sur la Résiduosité de Degré Composite. PhD thesis, École Nationale Supérieure des Télécommunications (1999)
18. Paillier, P.: Public-key cryptosystems based on composite residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–239. Springer, Heidelberg (1999)
19. Parkes, D.C., Rabin, M.O., Shieber, S.M., Thorpe, C.A.: Practical secrecy-preserving, verifiably correct and trustworthy auctions. In: ICEC 2006, pp. 70–81. ACM Press, New York (2006)
20. Rindi, B.: Transparency, liquidity and price formation. In: Proceedings of the 57th European Meeting of the Econometric Society (2002)
21. Stoll, H.R.: Market microstructure. In: Constantinides, G.M., Harris, M., Stulz, R. (eds.) Handbook of the Economics of Finance, Elsevier Science B.V., Amsterdam (2003)
22. Szydlo, M.: Risk assurance for hedge funds using zero knowledge proofs. In: Patrick, A.S., Yung, M. (eds.) FC 2005. LNCS, vol. 3570, Springer, Heidelberg (2005)
23. Wang, C., Leung, H., Wang, Y.: Secure double auction protocols with full privacy protection. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, Springer, Heidelberg (2004)
24. Yokoo, M., Suzuki, K.: Secure multi-agent dynamic programming based on homomorphic encryption and its application to combinatorial auctions. In: Proc. First Int. Conf. on Autonomous Agents and Multiagent Systems (2002)

Improved Multi-party Contract Signing

Aybek Mukhamedov and Mark Ryan

School of Computer Science
University of Birmingham, UK
{A.Mukhamedov, M.D.Ryan}@cs.bham.ac.uk

Abstract. A multi-party contract signing protocol allows a set of participants to exchange messages with each other with a view to arriving in a state in which each of them has a pre-agreed contract text signed by all the others. "Optimistic" such protocols allow parties to sign a contract initially without involving a trusted third party T. If all signers are honest and messages are not arbitrarily delayed, the protocol can conclude successfully without T's involvement. Signers can ask T to intervene if something goes amiss, for example, if an expected message is not received. Two multi-party contract signing protocols have been proposed so far.

One solution to this problem was proposed by Garay and MacKenzie (DISC'99) based on private contract signatures, but it was subsequently shown to be fundamentally flawed (it fails the fairness property). Another more efficient protocol was proposed by Baum-Waidner and Waidner (ICALP'00). It has not been compromised, but it is based on a non-standard notion of a signed contract.

In this paper, we propose a new optimistic multi-party contract signing protocol based on private contract signatures. It does not use a non-standard notion of a signed contract and has half the message complexity of the previous solution.

1 Introduction

A contract signing protocol allows a set of participants to exchange messages with each other with a view to arriving in a state in which each of them has a pre-agreed contract text signed by all the others. An important property of contract signing protocols is *fairness*: no participant should be left in the position of having sent another participant his signature on the contract, but not having received signatures from the other participants.

One way in which this can be achieved is by employing a trusted party T. All the signers of the contract send their signatures to T. When T has them all, he sends them out to each of the signers. It would be desirable to have a protocol which does not require a trusted party, but this is known to be impossible for deterministic protocols [7]. This has led to the invention of "optimistic protocols", which employ a trusted party only in the case that something goes wrong. If all the signers are honest and there are no adverse network delays which prevent the protocol from completing, the trusted party is not needed. But if a participant

of the protocol has sent messages which commit him to the contract and has not received corresponding commitment from the other participants, he can contact the trusted party who will intervene.

As well as fairness, there are other desired properties of contract signing protocols. *Timeliness* ensures that every signer has some recourse to prevent endless waiting. A third property called *abuse-freeness* guarantees that a signer is not able to prove to an external observer that she is in a position to choose between successfully completing the protocol and aborting it. This property is desirable because being in such a position would give the signer an unfair advantage.

Optimistic contract signing protocols have been first described for *synchronous* networks in [1,2,11,13]. 2-party protocols for *asynchronous* networks, have been proposed in [3,8,13], where all messages are eventually delivered, but without upper bounds on network delays. Later, two protocols for n-party case have been proposed: one by Garay and MacKenzie [9], and the other one by Baum-Waidner and Waidner [5]. Chadha, Kremer and Scedrov in [6] revealed and claimed to have fixed a flaw in the trusted party's protocol of Garay and MacKenzie's scheme. Later, we showed that Garay and MacKenzie's main protocol is flawed for $n > 4$ and fairness can not be restored whatever the trusted party does [12].

Baum-Waidner and Waidner's protocol requires $(n + 1)n(n − 1)$ messages in the "optimistic" execution, where n is the number signers and the number of dishonest signers can be up to $n − 1$. However, their protocol is based on a non-standard notion of a signed contract: a contract on a text m signed by an agent A is defined to be a tuple $(m, n + 1)$ digitally signed by A. Any other digitally signed (m, i) with $i < n+1$ is not considered to be a signed contract; it is merely A's promise to sign the contract. Such a notion has undesirable side-effects. The validity of the contract produced by Baum-Wainder and Waidners's protocol depends on the integer it is tupled with. Hence, when a party is presented with such contract it must be able to reliably establish $n+1$ (which could, for instance, be embedded in the body of the contract m) and compare with the integer that the contract is tupled with.

Baum-Waidner[4] further reduced the complexity of the previous scheme. This was achieved by adjusting trusted party T's protocol with an assumption that T knows in advance the number of dishonest signers (and sets the parameters of its protocol accordingly) and fairness is guaranteed provided *all* honest signers continue the protocol (i.e. if some honest signer decides to quit, when the protocol requires it to participate, fairness can not be guaranteed for other honest signers).

Our contribution. We propose a new optimistic multi-party contract signing protocol based on private contract signatures. It does not use a non-standard notion of a signed contract and achieves improvement in the message complexity of the optimisitc execution without assuming that T or any signer know the total number of dishonest signers. Our scheme requires $n(n − 1)\lceil n/2 \rceil + 1$ messages, which is about half the complexity of the previous protocol by Baum-Waidner and Waidner [5]. For example, if $n = 6$ our protocol requires 120 messages to "optimistically" sign a contract, whereas the previous scheme requires 210.

2   Model and Definitions

Let $P_1, \ldots, P_n$ denote signers, who want to sign a contract m and T a trusted third party. Signers may be honest, in which case they execute the protocol faithfully or *dishonest*, i.e. they deviate from the protocol. We assume that up to $n - 1$ of signers may be dishonest and are coordinated by a single party, the *adversary*. We assume that the ordered list $(P_1, P_2, \ldots, P_n)$ of signers is fixed in advance and included in the text m of the contract, and that all signers reliably know each others public key, and all contracts are distinct. $S_{P_i}(m)$ denotes P_i's universally-verifiable signature on m.

We shall say that P_i has a *valid* contract m, if it receives all signers' signatures on m. When P_i runs a contract signing protocol and acquires a valid contract m, we shall say "P_i decided signed". Otherwise, if it quits or receives an abort token from T we say "P_i decides failed".

We consider an asynchronous communication model with no global clocks, where messages can be arbitrarily delayed. However, the communication channels between signers and the trusted party T are assumed to be *resilient*, viz. the messages are guaranteed to be delivered eventually. The adversary is allowed to schedule and insert its own messages into the network. The protocol is expected not to fail, whatever such adversary does.

An optimistic contract signing protocol consists of two protocols, one executed by signer (*Main*), and another by trusted party T (*Abort* or *Resolve*). Usually signers try to achieve the exchange by executing Main. They contact T using Abort or Recovery only if something goes amiss in Main. Once a participant contacts T, it no longer takes part in Main. A request to T via Abort or Recovery can result in T sending back an abort token or a signed contract. The decision of whether to reply with an abort token or a signed contract is taken by T on the basis of the evidence included in the request, and also the previous requests that have been made by other participants. T has the property that if it decides to send back a signed contract, it sticks to that decision when answering further requests from other participants. However, if it issues an abort, it may later overturn that abort in order to maintain fairness. An honest participant (namely, one who adheres to the protocol) will not receive an abort and later have it overturned.

An optimistic contract signing protocol is expected to guarantee *fairness*. It is also desirable for the protocol to guarantee *abuse-freeness* and *timeliness*:

Definition 1. *An optimistic contract signing protocol is said to be* fair *for an honest signer P_i if whenever some signer P_j obtains $S_{P_i}(m)$ then P_i obtains $S_{P_k}(m)$ for all $1 \leq k \leq n$.*

Definition 2. *An optimistic contract signing protocol is said to be* abuse-free *if it is impossible for any set of signers at any point in the protocol to be able to prove to an outside party that they have the full power to terminate (abort) or successfully complete the contract signing.*

Definition 3. *An optimistic contract signing protocol is said to satisfy* timeliness *if each signer has recourse to stop endless waiting.*

3   The Protocol

The following is an optimistic multi-party contract signing protocol. The Main protocol, consists of $\lceil n/2 \rceil + 1$ rounds. In each round a signer P_i waits for promises from lower numbered signers (*below*), sends its promise to higher numbered signers (*above*), waits for promises from signers above and then send its promise to signers below. In the last round signers exchange actual signatures, together with their promises. If a signer does not receive some of the messages, it either quits the protocol or asks T to intervene.

PCS promises. Our protocol employs a cryptographic primitive known as *private contract signature* [8]. A private contract signature by P_i for P_j on text m with respect to trusted party T, denoted $PCS_{P_i}(m, P_j, T)$, is a cryptographic object with the following properties:

1. $PCS_{P_i}(m, P_j, T)$ can be created by P_i, and faked by P_j.
2. Each of P_i, P_j and T (but no-one else) can tell the difference between the versions created by P_i and faked by P_j.
3. $PCS_{P_i}(m, P_j, T)$ can be converted into a regular universally-verifiable signature by P_i, and by T; and by no-one else.

The idea is that $PCS_{P_i}(m, P_j, T)$ acts as a promise by P_i to P_j to sign m. But P_j cannot prove to anyone except T that he has this promise, since he can create it himself and only T can tell the difference between one created by P_i and one created by P_j.

In our protocol, agents exchange several such promises before issuing a signed contract. A promise issued by a signer at a later stage of the protocol signifies its stronger commitement to the contract as well possession of certain promises from other signers. Hence, τ-level promise of a signer P_i to P_j on m is a message $PCS_{P_i}((m, \tau), P_j, T)$, where $\tau \geq 0$ expresses the temporal ordering of P_i promises.

3.1   Main Protocol for Signer P_i

Each signer waits for 1-level promises from the signers below. On receipt of these, it sends its 1-level promises to the signers above it. Then it waits for 1-level promises from above, and on receipt, sends 1-level promises below. This sequence is repeated for r-level promises, for r ranging from 2 to $\lceil n/2 \rceil$, as shown in Figure 3.1. Finally, in the last round, $\lceil n/2 \rceil + 1$-level promises and signatures are exchanged. The protocol is defined formally in Table 1.

If expected messages are not received, a participant P_i may simply quit the protocol, or request abort or resolve from T, depending on where P_i is in the main protocol.

When P_i requests abort it sends to T the message:

$$S_{P_i}((m, P_i, (P_1, \ldots, P_n), abort))$$

For the resolve requests P_i sends

$$S_{P_i}(\{PCS_{P_j}((m, \tau_j), P_i, T)\}_{j \in \{1,\ldots,n\} \setminus \{i\}}, S_{P_i}(m, 0))$$

to T, where for $j > i$, τ_j is the maximum level of promises received from all signers P'_j with $j' > i$, and for $i > j$, τ_j is the maximum level of promises received from all signers P'_j with $i > j'$:

$$\tau_j = \begin{cases} \max\{\tau \mid \forall j' > i, & P_i \text{ has received } PCS_{P_{j'}}((m, \tau), P_i, T)\} & \text{if } j > i \\ \max\{\tau \mid \forall j' < i, & P_i \text{ has received } PCS_{P_{j'}}((m, \tau), P_i, T)\} & \text{if } j < i \end{cases}$$

(For example, if the maximum level promises P_4 receives from P_1 and P_2 is 3, and from P_3 it is 2, then P_4 would send 2-level promises for signers below.)

Table 1. Main protocol for signer P_i

Round 1

1. For each $j < i$, wait for promise $PCS_{P_j}((m, 1), P_i, T)$ from P_j.
   If any of them is not received in a timely manner, then quit.
2. For each $j > i$, send promise $PCS_{P_i}((m, 1), P_j, T)$ to P_j.
3. For each $j > i$, wait for promise $PCS_{P_j}((m, 1), P_i, T)$ from P_j.
   If any of them is not received in a timely manner, then request abort.
4. For each $j < i$, send promise $PCS_{P_i}((m, 1), P_j, T)$ to P_j.

For $r = 2$ to $\lceil n/2 \rceil$: Round r

5. For each $j < i$, wait for promise $PCS_{P_j}((m, r), P_i, T)$ from P_j.
   If any of them is not received in a timely manner, then request resolve.
6. For each $j > i$, send promise $PCS_{P_i}((m, r), P_j, T)$ to P_j.
7. For each $j > i$, wait for promise $PCS_{P_j}((m, r), P_i, T)$ from P_j.
   If any of them is not received in a timely manner, then request resolve.
8. For each $j < i$, send promise $PCS_{P_i}((m, r), P_j, T)$ to P_j.

Round $\lceil n/2 \rceil + 1$

9. For each $j < i$, wait for promise $PCS_{P_j}((m, \lceil n/2 \rceil + 1), P_i, T)$ and signature $S_{P_j}(m)$ from P_j.
   If any of them is not received in a timely manner, then request resolve.
10. For each $j \neq i$, send promise $PCS_{P_i}((m, \lceil n/2 \rceil + 1), P_j, T)$ and signature $S_{P_i}(m)$ to P_j.
11. For each $j > i$, wait for promise $PCS_{P_j}((m, \lceil n/2 \rceil + 1), P_i, T)$ and signature $S_{P_j}(m)$ from P_j.
    If any of them is not received in a timely manner, then request resolve.

3.2   Protocol for T

For each contract m with signers $P_1, \ldots, P_n$, when T learns about the contract (through abort or resolve request) it sets up a variable `validated`(m) initiated
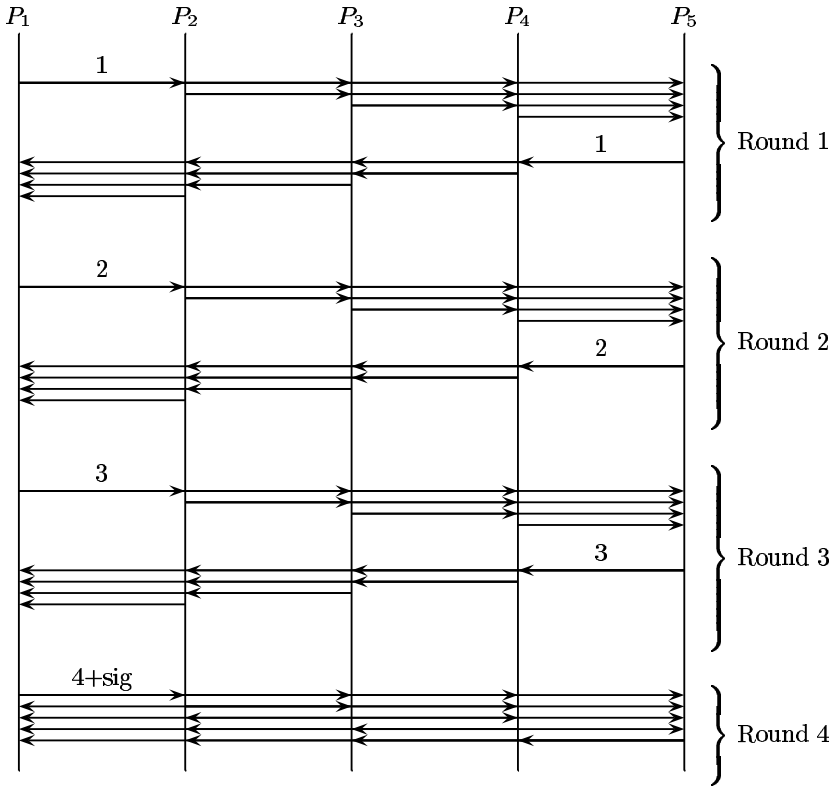
Fig. 1. Messages in the main protocol when $n = 5$

to false, which indicates if T decided to enforce the contract and has a full set of signatures (some converted by T from promises). T must reliably know the position of each signer in the run of the protocol, which can be deduced from the ordered list of signers included in the contract text m. T also maintains a set $S(m)$ of indexes of parties that contacted it in the past: signers are allowed to contact T only once. This set is also used when T considers whether to overturn its previous abort decision. For each signer P_i such that $i \in S(m)$, T also maintans two integer variables $h_i(m)$ and $l_i(m)$. Intuitively, $h_i(m)$ is the highest level promise P_i could have sent to any signer above, and similarly, $l_i(m)$ is the highest level of promise P_i could have sent to a signer below. This construction was inspired by the paper of Chadha, Kremer and Scedrov [6], even though it does not work for the protocol they consider [12].

Depending on the request T executes either Abort or Resolve protocol.

Abort protocol. When T receives an abort message from P_i, it adds i to the set $S(m)$. Then if the protocol has already been successfully recovered it sends back a signed contract; otherwise, it sends back an abort token (see table 2).

Table 2. Abort protocol for T

The first time T is contacted for contract m (either abort or recovery), T initialises $S(m)$ to $\emptyset$ and validated(m) to false.

If the abort message $S_{P_i}(m, P_i, (P_1, \ldots, P_n), abort)$ is received from P_i
> Check that the signature is valid
>
> if not validated(m) then
>> if $S(m) = \emptyset$ then store $S_T(S_{P_i}(m, P_i, (P_1, \ldots, P_n), abort))$
>> $S(m) = S(m) \cup \{i\}$
>> $h_i(m) := 1;\ l_i(m) = 0$
>> Send $S_T(S_{P_j}(m, P_j, (P_1, \ldots, P_n), abort))$ to P_i
>
> else
>> Send $\{S_{P_j}((m, \tau_j))\}_{j \in \{1, \ldots, n\} \setminus \{i\}}$ to P_i
>> where τ_j is the level of the promise from P_j that was converted to a universally-verifiable signature during the recovery protocol.

Resolve protocol. The recovery messages that T receives are designed so that T can infer what promises an honest signer could have sent and whether all the previous requests were made by dishonest signers. The protocol works is as follows:

1. T checks that all promises and signatures are valid, and promises from above and below are consistent (for details, see Table 3. If any of the checks fail, T ignores the request.
2. If there has been no previous query to T on m, i.e. validated(m) is false, it derives a signed contract by converting all the promises contained in the resolve request to universally-verifiable signatures. T puts the signed contract in its database, sends it back in reply to the request, and sets validated(m) to true.
3. If there has been a positive resolution before, i.e. validated(m) is true, T sends back the stored signed contract.
4. If there has been an abort, T replies with an abort token or overturns its previous abort decision if it deduces that all the previous requests were made by dishonest signers. T deduces that P_j is dishonest from P_i's resolve request if: P_i presents to T a promise made by P_j such which shows that P_j continued the protocol after making a request to T.

The protocol is defined formally in Table 3.

4   Properties of the Protocol

Our protocol respects *timeliness*, since all signers can choose to stop waiting (quit, request abort or resolve) at any time they are waiting to receive a message. In order to prove fairness, we need the following lemmas.

Table 3. Recovery protocol for T

The first time T is contacted for contract m (either abort or recovery), T initialises $S(m)$ to $\emptyset$ and `validated`(m) to false.

If the recovery message $S_{P_i}(\{PCS_{P_j}((m,\tau_j),P_i,T)\}_{j\in\{1,\ldots,n\}\setminus\{i\}}, S_{P_i}(m,0))$ is received
  Check that promises and signature are valid, and promises from above and below are consistent, i.e.:
    for all $j < i$, check that $\tau_j = \tau_{i-1}$
    for all $j > i$, check that $\tau_j = \tau_{i+1}$
    check that $\tau_{i-1} = \tau_{i+1}$ or $\tau_{i-1} = \tau_{i+1} + 1$

  if $i \in S(m)$ or one of the above checks failed then
    ignore the message

  else if $S(m) = \emptyset$ then
    `validated`$(m) :=$ true
    Send $\{S_{P_j}(m,\tau_j)\}_{j\in\{1,\ldots,n\}\setminus\{i\}}$ to P_i

  else if `validated`(m) then
    Send $\{S_{P_j}(m,\tau_j)\}_{j\in\{1,\ldots,n\}\setminus\{i\}}$ to P_i
      where τ_j is the level of the promise from P_j that was converted to a universally-verifiable signature.

  else // note that `validated`(m)=false $\wedge$ $S(m) \neq \emptyset$
    if $\exists p \in S(m) ((p < i \wedge \tau_p \leq h_p(m)) \vee (p > i \wedge \tau_p \leq l_p(m)))$ then
      Send the stored abort token $S_T(S_{P_j}(m,P_j,(P_1,\ldots,P_n),abort))$ to P_i
      $S(m) := S(m) \cup \{i\}$
      Compute $h_i(m)$ and $l_i(m)$ as follows:
      if $i = 1$
        // P_1 has contacted T in some step 7 of the main protocol
        $(h_i(m), l_i(m)) = (\tau_2 + 1, 0)$
      else if $i = n$
        // P_n has contacted T in some step 5 of the main protocol
        $(h_i(m), l_i(m)) = (0, \tau_{n-1})$
      else if $1 < i < n$ and $\tau_{i+1} = \tau_{i-1}$
        // P_i has contacted T in some step 5 of the main protocol
        $(h_i(m), l_i(m)) = (\tau_{i+1}, \tau_{i+1})$
      else if $1 < i < n$ and $\tau_{i-1} > \tau_{i+1}$
        // P_i has contacted T in some step 7 of the main protocol
        $(h_i(m), l_i(m)) = (\tau_{i+1} + 1, \tau_{i+1})$
    else
      Convert the promises into signatures $\{S_{P_j}(m,\tau_j)\}_{j\in\{1,\ldots,n\}\setminus\{i\}}$
      Store the signatures
      Send the signatures to P_i
      `validated`$(m) :=$ true

Lemma 1. *If a* resolve *request in round $r > 1$ results in an abort decision, then:*

1. *for all r' such that $1 < r' < r$ there are two* resolve *requests in round r' that resulted in an abort decision.*
2. *there is an abort request in round 1.*

Proof. 1. We define the following predicates:

$A(r)$: there exists a resolve request in round r from some signer P_i that results in an abort decision. P_i's request has $r - 1$ level promises from all other signers. We call such requests "type A".

$B(r)$: there exists a resolve request in round r from some signer P_i that results in an abort decision. P_i's request has r level promises from signers P_j, where $j < i$ and $r - 1$ level promises from P_j where $j > i$. We call such requests "type B".

Point 1 of the lemma states that if $r > 1$ then $A(r) \lor B(r) \rightarrow \forall r'.(1 < r' < r \rightarrow A(r') \land B(r'))$. We show this by proving the following: (a) $A(r) \land r > 2 \rightarrow B(r-1)$; (b) $B(r) \land r > 1 \rightarrow A(r)$.

To show (a): Suppose $A(r) \land r > 2$. Let P_i be the signer whose request results in abort. P_i's request has $r - 1$ level promises from all other signers. So, there has been a resolve request made by some signer P_k in round $r - 1$ (otherwise according to T's protocol any previous abort would be overturned). Moreover, k can be chosen to be less than i, since according to T's protocol, if all such k were greater than i, than P_i's request would have resulted in resolve. Therefore, P_k's resolve request contains $r - 1$ level promises from below and $r - 2$ level promises from above, since if it had only $r - 2$ level promises then P_i's request would overturn the abort received by P_k. Therefore, P_k's request shows $B(r - 1)$.

For (b): Suppose $B(r)$ and $r - 1$. Let P_i be the signer whose request results in abort. P_i's request has r-level promises from below and $r - 1$-level promises from above. Since P_i's request results in abort, there has been a resolve request made by some other signer in round $r' \leq r$. To see this, suppose that the highest r' for which there is a resolve request by a signer P_k other than P_i resulting in abort is less than r.

- if P_k's request is type B, then T sets $h_k(m) = r - 1$, $l_k(m) = r - 2$.
  - if $k < i$, then P_i's request has an r-level promise from P_k, contradicting $h_k(m) = r - 1$. So T overturns P_k's abort.
  - if $k > i$, then P_i's request has an $r - 1$-level promise from P_k, contradicting $l_k(m) = r - 2$. Again, T overturns P_k's abort.
- if P_k's request is type A, then T sets $h_k(m) = l_k(m) = r - 2$. P_i's request has an $r - 1$-level promise from P_k contradicting $h_k(m)$ or $l_k(m)$ as above. So T overturns P_k's abort.

Thus, in all cases, the assumption $r' < r$ leads to contradiction; and therefore $r' = r$. P_k's request proves $A(r)$.

2. If there is no abort in round 1, then according to T's protocol, any request by any participant in a later round will result in resolve.

$\square$

Lemma 2. *If T issues abort to P_i in a round $r > 1$ and then later resolve to P_j, then P_i is dishonest.*

Proof. Suppose P_i gets abort at round $r > 1$. The variables h_i and l_i are set according to T's Recovery protocol. We verify that h_i is the highest level promise P_i could have sent to any signer above, and similarly, $l_i(m)$ is the highest level of promise P_i could have sent to a signer below. There are four cases to consider:

- $i = 1$. Then $h_i = \tau_2 + 1 = \cdots = \tau_n + 1$ since P_1 sends out $\tau + 1$-level promises after receiving all τ-level promises, and $l_i = 0$ because P_1 doesn't send any promises to below.
- $i = n$. Then $h_i = 0$ since P_n doesn't send any promises to above, and $l_i = \tau_{n-1} = \cdots = \tau_1$ since P_n has received τ-level promises from everyone before he sends out any τ-level promises.
- $1 < i < n$ and all the τ_k's are equal. P_i has requested resolve while waiting for promises from below, and the evidence it sends are the promises it got in the previous round, which is now complete and it has sent out its promises in that round too. Therefore $h_i = l_i = \tau_k$ for all k.
- $1 < i < n$ and $\tau_1 = \cdots = \tau_{i-1} \neq \tau_{i+1} = \cdots = \tau_n$. Here, P_i's request for resolve is while waiting for promises from above, and its evidence consists of promises it received in two different rounds. The promises it has sent to signers above are $\tau_{i+1} + 1$-level promises, and to below they are τ_{i+1}-level promises, so h_i and l_i are set accordingly.

Now P_j asks for resolve with a request that contains $PCS_{P_i}((m, \tau_i'), P_j, T)$. Since this request does not result in abort, the conditions for abort (which begin "$\exists p$" in Table 3) must fail. Therefore, for all p, $(p < j \rightarrow \tau_p' > h_p) \vee (p > j \rightarrow \tau_p' > l_p)$. Take $p = i$ and we obtain $i < j \wedge \tau_i' > h_i$ or $i > j \wedge \tau_i' > l_i$; each case includes evidence that P_i continued the protocol since its request to T and is therefore dishonest. $\qquad\square$

Theorem 1. *The optimistic multi-party contract signing protocol above is fair.*

Proof. Assume P_i is an honest signer participating in the protocol to sign a contract m. Suppose P_i executed the protocol and decided failed, and some signer P_j decided signed. Then P_j has P_i's signature on m, because either: (1) P_i sent it in the last round of the main protocol; or, (2) T converted P_i's promise to P_j into a signed contract for P_j. We consider the two cases in turn.

1. Suppose P_i executed the last round of the protocol and sent out its signature on m. Then $i < n$ since P_n does not send out his signature until he has received everyone else's. Thus, P_i requested resolve from T in the last round with the request

$$S_{P_i}(\{PCS_{P_j}((m, \lceil n/2 \rceil + 1), P_i, T)\}_{j \in \{1,\dots,i-1\}},$$

$$\{PCS_{P_j}((m, \lceil n/2 \rceil), P_i, T)\}_{j \in \{i+1,\dots,n\}}, S_{P_i}(m))$$

and received abort. Since $i < n$ and P_i gets abort in the last round, T has evidence to overturn any abort issued in any previous round. Since T does not overturn all previous aborts, there is an abort given to P_k with $k > i$ in the last round. Thus P_i and P_k got abort in the final round ($\lceil n/2 \rceil + 1$). By lemma 1, rounds 2 to $\lceil n/2 \rceil$ have two failed resolve requests and round 1 has an abort request. The total number of requests is thus $2 + (\lceil n/2 \rceil - 1) \times 2 + 1 = 2\lceil n/2 \rceil + 1$. This is at least $n + 1$, but there are only n signers and each signer can make at most one request: a contradiction.

2. Suppose T returned a signed contract in response to a resolve request from P_j. There are three cases to consider:
   - If P_i quit the protocol in round 1, T could not have returned a signed contract, since P_i did not release any promises.
   - If P_i requested abort in round 1 from T, then it could have sent 1-level promises to signers above. Hence, T sets $h_i(m) = 1$ and, since P_i is honest, it does not release further promises. According to T's protocol, T could not have returned a signed contract, since any subsequent resolve request would only have $PCS_{P_i}((m, 1), P_k, T)$, where $k > i$.
   - If P_i received an abort decision for its resolve request in some round $1 < r \leq \lceil n/2 \rceil + 1$, and then P_i's promise to P_j got converted to a signature, then by lemma 2 P_i is dishonest.

In all three cases we reach a contradiction.   □

Abuse-freeness. Intuitively, the protocol is abuse-free, because of the use of private-contract signatures. No party has publicly verifiable information about P_i's commitment to the contract until a point from which P_i has the power to acquire a signed contract from all the other participants. (In future work, we intend to investigate our protocol in terms of formal definitions of abuse-freeness, such as that of [10]).

Timeliness. Our protocol also satisfies timeliness, since a participant can give up waiting for a message at any time and take recourse with the trusted party.

Remarks. Our protocol above works for up to $n - 1$ dishonest signers. It can be optimized in the same way as it was done by Baum-Waidner and Waidner [5]: if the number of dishonest signers t is less and is known advance to all honest signers, then we can reduce the number of messages for the Main protocol. For Baum-Waidner, it results in $(t + 2)n(n - 1)$ messages; in our case it is $(\lceil (t + 1)/2 \rceil + 1)n(n - 1)$.

The number of messages of the "optimistic" execution can also be reduced if we allow signers to forward other signers' messages. In particular, a signer P_i instead of broadcasting its promise to all signers above, can now send those messages to P_{i+1}, who will then send P_i's promises intended for other signers (together with his) to P_{i+2}, and so on. Similarly, the same changes are applied when P_i sends promises to signers below. As a result, the number of messages sent in the "optimistic" execution is now $(\lceil n/2 \rceil + 1)2(n - 1)$.

Garay and MacKenzie [9] state that any complete and fair optimistic contract-signing protocol with n participants requires at least n rounds in an optimistic

run. Our result appears to contradict that statement, but it is not clear since they did not define what a round is. Different protocols group messages into rounds in different ways, so the only meaningful comparison is by number of messages in the optimistic execution.

5   Conclusions

We have presented a new multi-party contract signing protocol which uses private contract signatures. The previous multi-party contract signing protocol based on private contract signatures by Garay and MacKenzie [9] has been shown to be incorrect [6,12].

Our scheme improves on the state-of-the-art protocol by Baum-Waidner and Waidner [5] in two important aspects. Firstly, our schemce requires only half the number of messages to complete "optimistic" execution . (In contrast with Baum-Waidner's improvement reported in [4], we do not require the unrealistic assumptions that the number of dishonest signers is known in advance to the trusted party, and that honest signers don't quit the protocol.) Secondly, our scheme does not use a non-standard notion of a signed contract.

References

1. Asokan, N., Schunter, M., Waidner, M.: Optimistic protocols for multi-party fair exchange. Research Report RZ 2892 (# 90840), IBM Research (December 1996)
2. Asokan, N., Schunter, M., Waidner, M.: Optimistic protocols for fair exchange. In: Matsumoto, T. (ed.) 4th ACM Conference on Computer and Communications Security, Zurich, Switzerland, pp. 8–17. ACM Press, New York (1997)
3. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 591–606. Springer, Heidelberg (1998)
4. Baum-Waidner, B.: Optimistic asynchronous multi-party contract signing with reduced number of rounds. In: Orejas, F., Spirakis, P.G., van Leeuwen, J. (eds.) ICALP 2001. LNCS, vol. 2076, pp. 898–911. Springer, Heidelberg (2001)
5. Baum-Waidner, B., Waidner, M.: Round-optimal and abuse free optimistic multi-party contract signing. In: Montanari, U., Rolim, J.D.P., Welzl, E. (eds.) ICALP 2000. LNCS, vol. 1853, pp. 524–535. Springer, Heidelberg (2000)
6. Chadha, R., Kremer, S., Scedrov, A.: Formal analysis of multi-party fair exchange protocols. In: Focardi, R. (ed.) 17th IEEE Computer Security Foundations Workshop, Asilomar, CA, USA, pp. 266–279. IEEE Computer Society Press, Los Alamitos (2004)
7. Even, S., Yacobi, Y.: Relations among public key signature systems. Technical report, Technion, Haifa (March 1980)
8. Garay, J.A., Jakobsson, M., MacKenzie, P.D.: Abuse-free optimistic contract signing. In: Wiener, M.J. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 449–466. Springer, Heidelberg (1999)
9. Garay, J.A., MacKenzie, P.D.: Abuse-free multi-party contract signing. In: Jayanti, P. (ed.) DISC 1999. LNCS, vol. 1693, pp. 151–165. Springer, Heidelberg (1999)

10. Kähler, D., Küsters, R., Wilke, T.: A Dolev-Yao-based Definition of Abuse-free Protocols. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 95–106. Springer, Heidelberg (2006)
11. Micali, S.: Certified E-mail with invisible post offices. Available from author; an invited presentation at the RSA 1997 conference (1997)
12. Mukhamedov, A., Ryan, M.D.: Resolve-impossibility for a contract-signing protocol. In: CSFW 2006. 19th Computer Security Foundations Workshop, IEEE Computer Society Press, Los Alamitos (2006)
13. Pfitzmann, B., Schunter, M., Waidner, M.: Optimal efficiency of optimistic contract signing. In: Seventeenth Annual ACM Symposium on Principles of Distributed Computing, pp. 113–122. ACM Press, New York (1998)

Informant: Detecting Sybils Using Incentives[⋆]

N. Boris Margolin and Brian N. Levine

Department of Computer Science, Univ. of Massachusetts, Amherst, MA, USA
{margolin,brian}@cs.umass.edu

Abstract. We propose an economic approach to Sybil attack detection. In our *Informant* protocol, a *detective* offers a reward for Sybils to reveal themselves. The detective accepts from one identity a security deposit and the name of target peer; the deposit and a reward are given to the target. We prove the optimal strategy for the informant is to play the game if and only if she is Sybil with a low opportunity cost, and the target will cooperate if and only if she is identical to the informant. Informant uses a Dutch auction to find the minimum possible reward that will reveal a Sybil attacker. Because our approach is economic, it is not limited to a specific application and does not rely on a physical device or token.

1 Introduction

Networked applications often assume or require that identities over network have a one-to-one relationship with individual entities in the external world. A single individual who controls many identities can disrupt, manipulate, or corrupt peer-to-peer applications and other applications that rely on redundancy; this is commonly called the Sybil attack [1].

While there has been quite a bit of research on deterring Sybil attacks using such techniques as identity certification, resource testing, and reputation systems [1,2,3,4,5,6], detection of Sybil attacks has received less attention. Existing detection methods are applicable only in very specific circumstances and applications, such as mobile sensor networks [7,8,9].

In this work, we take an economic approach in proposing a novel detection protocol called *Informant*. Our protocol provides an incentive to Sybil attackers to reveal two or more controlled identities in exchange for a payment. When the offered incentive exceeds the attackers opportunity cost for this admission, rational attackers will participate. Sybil identities by definition cannot be easily linked to any real-world identity — if they could, preventing the Sybil attack would be trivial. Thus, these revelations do not reveal the attacker herself, which removes an impediment towards participating.

One of the key challenges in designing our incentives scheme is avoiding false claims — a presentation of identities that are *not* controlled by a single entity. To solve this problem, we introduce a *trust game* that makes false claims financially
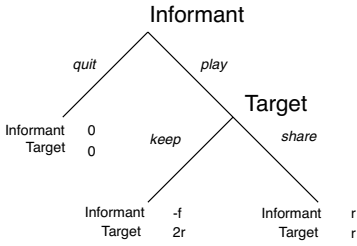
Fig. 1. Utilities in the Trust game when the Identities are independent
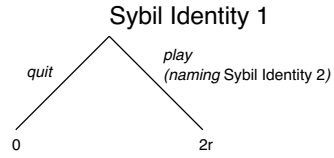
Fig. 2. Utilities in the Trust game when the Identities are part of a Sybil attack

risky for the claimant. Figure 1 outlines the game, which takes place between three identities: a *detective*, an *informant*, and a *target*. The informant and the target may be independent or they may be part of a Sybil controlled by a single entity; the detective offers a reward as incentive for the informant to reveal the truth. In a real peer-to-peer application, the informant and target are simply participants; the detective can be any entity. For an informant to claim to be part of a Sybil with a target, she must provide a security deposit f to the detective. The target then receives a payment of \$$(2r + f)$ from the detective. She is free to do whatever she wants with the reward.

If the identities are controlled by the same entity, that entity has made a profit of \$$2r$ by revealing its nature. If they are independent, the informant has lost \$f. She might ask the informant to give her back \$$(r + f)$ so that they share the reward of \$$2r$, but she has no way of imposing this desire, so a rational target will not cooperate. (We defer discussion of repeated games Section 4.) As we show more formally in this paper, when informants are rational, the detective can believe any claim.

This game does not distinguish between Sybil attackers and friends with a very high degree of trust in each other. But entities with high trust in each other, even if not actually malicious, are a matter for concern. They are capable of attacking or distorting the operation of an application, and break the assumption of redundancy required for availability, security, or privacy in many distributed applications [10,11,12]. While trust in the real world is complex, we are concerned only with a simple, monetary definition of trust and trustworthiness.

Contributions

- We define the *Trust Game*, which measures trust with a simple two-player economic protocol. We prove the optimal strategies for each participant: the informant will accept the game if and only if she has a high trust in the target, and the target will cooperate if and only if she has a high trustworthiness towards the informant.
- We define a more sophisticated game, called the *Sybil Game*, that includes the economic benefit to the detective of learning of Sybils and the economic cost to informant and target of revealing that Sybils are present. We prove

the optimal strategies for each participant. The detective will offer the game if and only if it will determine her choice about using the application in which these identities participate. The informant will accept the game if and only if she is Sybil with a low opportunity cost, and the target will cooperate if and only if she is identical to the informant. Both games are presented in Section 3.

– We propose the *Informant* protocol for detecting Sybils, based on our Sybil Game, in Sections 4 and 5. Informant uses a Dutch auction to find the minimum possible reward that will still reveal a Sybil attacker.

We present the background and related work in Section 2.

Our contributions are part of a growing set of results that apply economic games to security and p2p applications, including distributed hash table applications [13], multicast applications [14], file-sharing applications [15], anonymous communication systems [16], the Sybil attack [17], and digital rights management [18]. Because these emerging approaches rely on incentives and other economic mechanisms, they are not as exact as cryptographic mechanisms. However, they are able to address many problems that have resisted traditional cryptographic solutions.

2   Background

In this section, we discuss the prior work on incentive systems, define the Sybil attack, and discuss prior work on discouraging and detecting Sybil attacks.

Incentives. Many researchers have evaluated the behavior of rational participants in peer-to-peer networks, with a focus on the problems of free-riding. Shneidman and Parkes [19] give evidence that participants in p2p networks behave rationally. Our own work on incentive systems includes SPIES [20,18], an incentive system for digital rights management, and an analysis of the incentives for malicious Sybil attacks [17].

Sybils. Our formal definitions of the Sybil attack are adapted from Douceur [1]. *Entities* are economically rational agents that control one or more *identities*, which are the actors within a network protocol. Identities can send messages to each other through pipes connected to a *communications cloud* that obscures link-level information. Messages received by an identity are communicated to its controlling entity out-of-band. We assume that it is not possible to eavesdrop on these Identity to Entity communications.

Sybil Deterrence. The most popular approaches to Sybil deterrence are resource testing and trusted certification, both of which are discussed in Douceur [1]. Other methods include reputation systems, task verification, temporary identities, recurring payments, and per-resource payments.

A full discussion of these methods is beyond the scope of this paper, but none of them are completely effective at eliminating Sybil attacks. Resource testing

is ineffective for most systems, and trusted certification is usually too expensive [1,8,9]. Most reputation systems can be easily subverted by Sybil attackers, and those that cannot are less informative and little used in practice [21,4,22]. Task verification is only applicable for a small subset of applications [23]. Temporary identities, recurring payments, and per-resource payments require either the use of electronic cash or of significant human effort [24]. Since deterrence is not a completely solved problem, we believe that Sybil Detection is a critical part of a layered defense.

Sybil Detection. The Sybil detection research has focused on direct observation, Douceur's term for link-level or application-level observations dependent on a weakly anonymizing communications cloud. This type of detection is applicable to Sybil attacks on specific types of applications, such as sensor networks. Newsome et al [8] suggest active position verification as a method of direct observation in sensor networks. Of course, two independent sensors may be close together, but it is unlikely that a large number will be consistently close together. In our previous work [25], we suggest a passive approach to detecting sybil attacks in wireless networks.

In the more general case, Kohno, Broido, and Claffy [26] use clock skews to correlate messages originating from a single computer. This method has been used successfully against previous versions of *honeyd* (`http://www.honeyd.org`), a utility for making honeynets, a special class of Sybil identities. However, more recent versions of *honeyd* resist clock skew classification, and it appears that, in general, knowledgable and determined Sybil attackers can defeat clock skew analysis.

Direct observation focuses on devices rather than common control, so it is ineffective against attackers who, for example, are able to acquire a geographically diverse set of zombie machines. Our Sybil detection protocol does not rely on direct observation, but on the economic unity of Sybil identities, so its areas of application depend on the specific utilities of the participants rather than on the protocol details.

The work most similar to ours concerns attacks on auction systems. Yokoo et al [27] discuss the Sybil attack in combinatorial auctions, but the results are not generalizable to other applications. Rubin et al [28] use techniques similar to ours to find collusion in eBay auction records. Their work differs in that they do not address the general problem of Sybil detection, and they are focused on passive, rather than active, Sybil detection.

3   The *Trust* and *Sybil* Games

We introduce our solution to the problem of Sybil detection in three steps. First, in this section, we analyze the optimal strategies of the simple Trust game that we introduced in Section 1. Second, later in this section, we introduce the more realistic Sybil game, which extends the Trust game to take into account the self-interests of the detective and informant. Third, in the next section, we use the Sybil game as the core mechanism of a protocol for Sybil detection.

Trust Game

1. The informer chooses whether to *quit* or to *play*.
   - If *quit*, the game ends with a net profit of $0 to both players.
   - If *play*, she gives a security deposit of f to the detective, identifies the target, and the game proceeds.
2. The target receives a payment of $(2r + f)$. If the informant and target are part of a Sybil, the Sybil attacker has made $2r$.
   If they are independent, then the target chooses whether to *keep* or *share* the money.
   - If *keep*, she earns $(2r + f)$ (and the informant loses f)
   - If *share*, she and the informant both earn r.

Fig. 3. The Trust game

We define trust in terms of the conflict between self-interest and cooperation in an economic game involving two entities, an informer and a target. These entities may be identical, as in the case of a Sybil attack, or they may be independent. We discuss distinct, collaborating identities to Section 3.3.

3.1   The Trust Game

The Trust game, shown in Figures 1 and 2, measures the relationship between two identities with a simple economic protocol. The possible outcomes are *quit*, *(play,cooperate)*, and *(play,defect)*; the outcome that is reached depends on the the trust the informer has in the target and the trustworthiness of the target. A formal definition of the game is in Figure 3.

Optimal Strategies in the Trust Game. Rational agents in the Trust game will act to maximize their profit. We assume that both agents are rational and that this is common knowledge, and show that the informer will cooperate when her trust in the target is high, while the target will cooperate when her trustworthiness towards the informer is high. The outcomes of the Trust game given f and r therefore provide information about the relationship between the two entities involved. We now evaluate the players' strategies formally.

Theorem 1. *In the Trust Game, the behavior of rational identities is as follows:*

   (i) *If the target is independent from the informant, she will choose to keep the money. (If they are identical, she has no choice to make.)*
  (ii) *A Sybil identity will play the game and name another Sybil identity.*
 (iii) *An identity will not name an independent identity.*

Proof. We use the common game theoretic technique of backwards induction [29] to determine the strategies of the players in the Trust game. The target makes $(2r + f)$ if she keeps the money, and r if she shares it; so a rational target will keep it. This establishes *(i)*. A Sybil identity makes $2r$ if she names another Sybil identity, and $0 if she does not play. So a rational Sybil attacker will choose

---

Sybil Game

1. The detective chooses whether to *offer* the protocol or *quit*.
   - If the detective quits, the game ends. The net profit to each participant is \$0.
   - Otherwise, the game continues.
2. The informant chooses whether to *reject* the game or *play*.
   - If *rejects*, the game ends and the net profit to each participant is \$0.
   - If *play*, she gives a security deposit of \$$f$ to the detective, identifies the target. The target receives a payment of \$$(2r + f)$.
3. If the target is identical to the informant, the game ends with net profits of: \$$(b-2r)$ for the detective; \$$(2r - c)$ for the sybil.
   Otherwise, the target chooses whether to *keep* the money or *share* it.
   - If *keep*, the net profits are: \$$(b - 2r)$ for the detective; \$$(-f - c)$ for the informant; \$$(2r - c)$ for the target.
   - If *share*, the net profits are: \$$(b-2r)$ for the detective; \$$(r-c)$ for the informant; \$$(r - c)$ for the target.

Fig. 4. The Sybil game

to play. This establishes *(ii)*. To show *(iii)*, note that the informant is aware of the target's rationality, and so can predict that the target will keep the money. So if she names an independent target, she expects to lose \$$f$, versus a profit of \$0 for not playing, or \$$2r$ for naming part of a shared Sybil. Therefore, neither Sybil identities nor ordinary identities will name an independent identity.   □

The actions of the informant thus signal its type: Sybils will play the game, naming another Sybil identity, and ordinary identities will not (since they could only name independent identities. Collaborators may or may not play the game; they are discussed in Section 3.3.

3.2   The Sybil Game

The Sybil game is an extension of the Trust game that includes the economic benefit to the detective of learning of Sybils and the economic cost to Sybils of revealing their relationship.[1] The Sybil game is again between three players: the detective, who offers the game, an informer, and a target. As before, the informer and target may be identical. The game has four variables: f the security deposit; r the reward; b the detective's monetary benefit for learning about a Sybil relationship; c the attacker's cost for revealing a Sybil attack. The values r and f are known to all participants, while the others are private. The steps of the Sybil game are defined in Figure 4.

   In some cases b and c could be related, but this does not change the analysis of a particular instance of the game, since neither the detective or the attacker is able to affect b or c by their choices.

---

[1] A reasonably careful Sybil attacker can both reveal Sybil identities and continue to participate in the application, so the cost to Sybils is in additional precautions that application users may take, not in being excluded from the application or prosecuted.
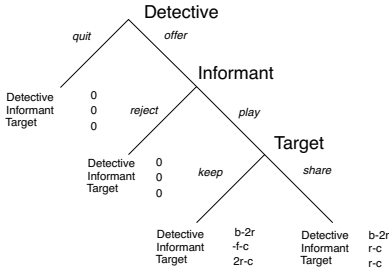
Fig. 5. Utilities in the Sybil game when the identities are independent
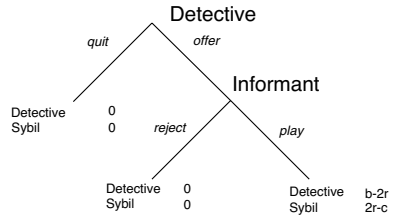
Fig. 6. Utilities in the Sybil game when the identities are part of a Sybil attack

The game has the four outcomes *quit, (offer,reject), (offer,play,keep)*, and *(offer,play,share)*. We now consider under which conditions each strategy is an equilibrium.

Strategies in the Sybil game. The strategies for the Sybil game are similar to those of the Trust game for the informer and target. The detective is best-off offering the game whenever her benefit exceeds the rewards she pays out.

Theorem 2. *In the Sybil Game, the optimal strategies for rational players are as follows:*

(i)   The detective will offer the game when $b \geq 2r$, and to quit otherwise.
(ii) The informant will accept the game if it is a Sybil with $c \leq 2r$, and reject it otherwise.
(iii) If the target is independent from the informant, it will choose to keep the money.

Proof. First note that parts *(ii)* and *(iii)* are exactly as in Theorem 1. For *(i)*, note that the detective expects she will receive \$0 if she quits, and somewhere between \$0 and \$$b - 2r$ if she offers the game. So she will offer the game as long as $b \geq 2r$.   □

3.3   Collaborators

Collaborators may act like independent entities or like Sybil identities in the Trust and Sybil games, depending on the level of trust between them. Suppose that the informant and the target are collaborators, and they have agreed to share the money received from the detective. When she is considering whether to name the target, the informant must consider how likely the target is to share the money received. Let p be her estimation of the probability of sharing, let r be the reward, and let f be the security deposit. Then naming the target is more profitable than not when $pr - (1-p)f > 0$, so the informant will play the game and name the target when $f < \frac{p}{1-p}r$. Collaborators also may have a much higher

opportunity cost than Sybils in revealing their identities, since these identities may be easier to link to the collaborators' real-world information.

Because of this additional complexity, we focus on Sybils and independent entities in this paper. However, the Informant protocol, which we define in the next section, will also detect collaborators if the security deposit is set low enough; it can be tuned to detect only Sybils by setting a high security deposit.

4   The Informant Protocol

In this section, we present Informant, a protocol to detect Sybils. The protocol is based on an extension of the Sybil Game. Rather than using a fixed reward r, it uses a *reverse Dutch auction* to determine the minimum possible reward that will still reveal a Sybil attacker. This benefits the detective financially without reducing the number of Sybils detected.

Informant has several positive characteristics. It reveals a Sybil attacker when her aversion to being detected is less than the detective's interest in detecting her. It benefits the detective when the possible presence of highly-averse Sybil attackers does not deter her from participating in the underlying peer-to-peer application. These properties are demonstrated in our analysis of the protocol in Section 5.

4.1   Assumptions

Informant is built on several assumptions: that entities are rational, that an application has a recurring join cost, and that the questioner is trusted. Each assumption is discussed in greater detail below.

Rational Entities. We assume that attackers and other entities participating in the application are rational in the economic sense. In other words, we assume that entities have goals and beliefs, and that they act according to their beliefs in order to achieve these goals. This assumption excludes "attacks" from malfunctioning client software or user error. Entities need not be specific human beings; they may also be corporate or criminal entities that are sufficiently organized to act in an economically rational way.

Following traditional game theory, we do not assume that an attacker's *goals* are reasonable. An attacker's goals could include disfiguring websites or feeding false information, based on personal conflict, basic malice, or capriciousness. Making money is a basic goal that we assume all entities participating share. Again following traditional game theory, we assume that some specific monetary value can be assigned to each of an attacker's goals.

Recurring per-identity join cost. We require a specific and recurring per-identity cost to participating in the distributed application. This cost can come in the form of a monetary entry fee, a CAPTCHA [30] solution requirement, proof of receipt of an SMS message, or some other form. Each entity with an identity in the protocol should be able to verify (at least manually) that another

identity has borne this entry cost. This can be done, for example, by having each identity participate in generating challenges such as CAPTCHAs in a challenge round, or by using a trusted "payment certification" authority.[2]

We do not require a specific method, and we do not further discuss entry fee verification. However, the authors have available an extensive analysis of using recurring per-identity join costs as a method of limiting Sybil attacks [17]. Unlike non-recurring fees, our results show that the difficulty of launching a Sybil attack increases with the number of other identities present in the application. In fact, many researchers rely on recurring per-identity costs (often in the form of limited resources) to limit the effectiveness of Sybil attacks [31,3,32,24,33,34,35].

Trusted Querier. Since our Sybil detection protocol allows the querier to cheat the respondents, the querier must be trusted to follow the protocol correctly. Since the querier does not need to be anonymous, and thus can be held accountable for her actions, this assumption is reasonable in many cases. Nevertheless, this does limit the scenarios in which Informant can be used.

4.2   Protocol Details

The variables and notation we use to describe our detection protocol are summarized below.

- i The ID number of a particular instance of the protocol.
- τ The time between rounds.
- r_0 The initial reward for established Sybil identity claims.
- r The current reward for established Sybil identity claims.
- n A nonce used to ensure freshness.
- f The security deposit required of the informant.

If possible, some discrete round of the distributed application where each identity receives some service should complete before Informant starts. This approach minimizes the opportunity cost to attackers in revealing some of their Sybil identities because any Sybil attack during the service phase will have already succeeded or failed. In the next round of the distributed application, the entity operating the Sybil attack can use all new identities, so knowledge of attacker identities in the current application incarnation will not prevent the attacker from introducing identities in the next incarnation. If the distributed application does not support discrete phases, Informant can still be used, but the opportunity costs to attackers will be higher.

In the protocol description below, $\$x$ represents an electronic payment of $\$x$ dollars usable by the message recipient[3]. $A \rightarrow * : m$ represents the broadcast of message m to all distributed application participants. This broadcast is assumed to be reliable.

---

[2] Such an authority requires much less trust than an identity certification authority that needs to gather non-electronic information and deal with key-distribution problems.

[3] We do not require any specific form of electronic payment; it does not need to be anonymous electronic cash, although this is an acceptable form.

Fig. 7. Steps in the Sybil Discovery protocol

Informant requires each participating identity A to have a public/private key pair, designated K_A^+ and K_A^-, respectively. In contrast to the key pairs used in many protocols, these are not part of any PKI and cannot be used to establish an entity's true identity; in particular, nothing prevents a single entity from computing or acquiring an arbitrarily large number of new pairs for each round of the application. We assume the application that Informant supports has a method of obtaining or exchanging each identity's public key.

Informant is broken into a number of steps. Figure 7 shows the control flow of the protocol.

1. $Q \rightarrow * : [i, \tau, f]_{K_Q^-}$
   Announce Auction. The detective Q announces a new auction and its parameters as a signed message.

2. $Q \rightarrow * : [i, f, r, n]_{K_Q^-}$
   Announce Auction Round. The detective sets the current price $r = r_0$. She announces to all application participants that she is willing to pay \$$r$ for knowledge of a single Sybil relationship, and includes nonce n.
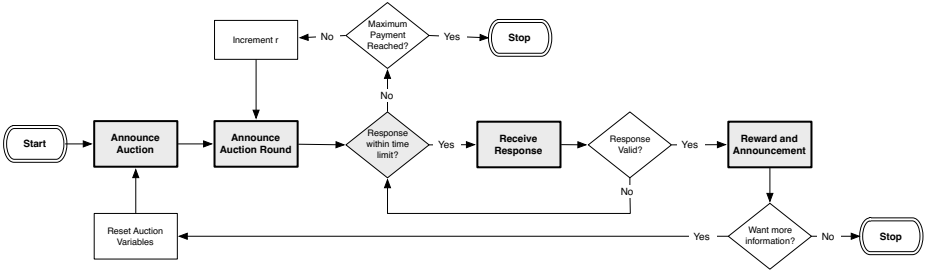
3. If no responses are received in τ secs, Q increments r by any desired amount, chooses a new nonce n, and reruns Step 2. If $r = b/2$, the protocol ends — nothing further can be learned.

4. $A \rightarrow Q : [n, \$f, A, B]_{K_A^-}$
   Receive Response. Otherwise, a claim of common control between A and B and a payment of \$$f$ has been received. Only the first valid message received is valid. The use of the nonce n prevents identities from sending messages before the round is announced.

5. $Q \rightarrow B : \$(f + r)$
   Reward Payment. The detective sets up an authenticated channel with the candidate Sybil identity B, and pays her \$$(f + r)$.

6. $Q \rightarrow * : [i, A, B]_{K_Q^-}$
   Announcement. The detective broadcasts to all application participants the claim of common control between B and C. This announcement prevents the claim from being sold to multiple detectives, as well as providing valuable information to the application participants. This broadcast includes the auction ID i.

At the end of Step 6, Q has good reason to believe in a common control (Sybil) relationship between the identities A and B. Optionally, the protocol could be amended to include in Step 4 a hashed nonce supplied by A; before the reward is paid, Q would require B to provide a signed copy of the nonce and n. This addition would prevent false claims if that was a concern.

If Q still wants to learn more information about Sybils present, she resets i, τ, and r_0, and restarts the protocol. Since common control is transitive, over multiple runs the initiator can learn about larger Sybil groups of identities.

Note that a Sybil attacker does not know whether other Sybils are present, nor what the questioner's maximum possible payment $b/2$ is. It is therefore in her interest to respond soon after r exceeds her opportunity cost t.

5   Sybil Detection Protocol Analysis

In this section, we analyze Informant in economic terms, giving the best strategies for the various participants and validating the claims of usefulness of Section 4. We also discuss the problem of opportunistic Sybils and how they can be avoided.

To facilitate analysis, we make several simplifying assumptions. First, we assume that there is either one entity attacking, with probability γ, or none, with probability $1 - \gamma$. We exclude situations with multiple Sybil attackers. Second, we assume that a Sybil attacker belongs to one of just two classes. Attackers in a *low-cost* class reveal themselves if offered a reward of $\$c < 2r$, where $2r$ is the maximum reward the detective is willing to pay. Attackers in the *high-cost* class reveal themselves if offered a reward of $\$C > 2r$. Attackers belong to the low-cost class with probability $1 - \delta$ and to the high-cost class with probability δ. This simplifies the analysis, but does not significantly alter the results. Third, we assume that the informant is always able to guess the maximum reward that will be offered, so that the target always receives the maximum payment of $\$(2r + f)$. This is a conservative assumption. Finally, we assume that the detective's interest in running the Informant protocol is to decide whether to stay in the underlying application or to leave it. If the detective instead makes some other choice based on the outcome of the tests, our analysis applies so long as the choice has the same incentive structure.

We represent the presence or absence of a Sybil in the underlying application, and the Sybil's type if present, as a choice of the class of the informant, made randomly by a player representing the role of nature[4]; non-sybil with probability $(1 - \gamma)$, low-profit Sybil with probability $\gamma(1 - \delta)$, and high-profit Sybil with probability $\gamma\delta$. This type of game is called a *signaling game* [29]; the informer, by accepting or rejecting the game, in effect signals her (otherwise unobservable) type to the detective.

The game tree, player utilities, and equilibrium strategies are shown in Fig. 8.

---

[4] The somewhat counterintuitive use of a "nature" player is a standard game-theoretic technique.

	No Sybil	Low-profit Sybil	High-profit Sybil	
leave	0 0	0 0	0 0	Detective Utility Informant Utility
stay	B 0	B-H c	B-H C	Detective Utility Informant Utility
leave	0 0	0 0	0 0	Detective Utility Informant Utility
stay	B 0	B-H c	B-H C	Detective Utility Informant Utility
leave	-2r -f	-2r 2r	-2r 2r	Detective Utility Informant Utility
stay	B-2r -f	B-H-2r c+2r	B-H-2r C+2r	Detective Utility Informant Utility
leave	-2r r	-2r 2r	-2r 2r	Detective Utility Informant Utility
stay	B-2r r	B-H-2r c+2r	B-H-2r C+2r	Detective Utility Informant Utility

Fig. 8. Equilibria in Informant

Theorem 3. *For rational Detectives, Informants, and Targets,*

(i) *If the detective runs Informant, she will stay if there are no Sybils reported, and leave if there is a Sybil reported.*

(ii) *Low-cost Sybils will play the game and announce themselves. Non-Sybil identities and high-cost Sybils will not play the game.*

(iii) *The detective will run Informant unless $(1 - \delta + \delta\gamma)B < \delta\gamma H + 2\delta(1 - \gamma)r$ where δ is the probability of a Sybil; γ the probability a Sybil is high-cost (and therefore highly averse to detection); B is the benefit of the protocol; and, H is the harm from the Sybil. Otherwise she will leave the application.*

In other words, Informant is useful so long as the expected harm from high-profit Sybils ($\delta\gamma H$) does not greatly exceed the benefit B provided by the application.

Proof. Parts *(i)* and *(ii)* are directly implied by the utilities in Figure 8: any change of strategy from the shaded Nash equilibria results in a lower utility for either the detective or the informant.

Given *(i)* and *(ii)*, the detective's expected utility from running Informant is B if there are no Sybils, $-2r$ if there is a low-cost Sybil, and $B - H$ if there is a high-cost Sybil that has a high aversion to detection. The probabilities of these outcomes are $(1 - \delta)$, $\delta(1 - \gamma)$, and $\delta\gamma$, respectively, so the detective's expected utility when running Informant is $(1 - \delta)B + \delta(1 - \gamma)(-2r) + \delta\gamma B$, which is positive when $(1 - \delta + \delta\gamma)B > \delta\gamma H + 2\delta(1 - \gamma)r$. This establishes *(iii)*. □

5.1   Opportunistic Sybils

We have evaluated Informant under the assumption that the Sybil attacker has a malicious interest in the underlying application. However, careless use of our Sybil detection protocol could induce *opportunistic Sybils*, which are attackers that have no interest in the application itself; instead, they participate and form a Sybil in the hopes of being paid to reveal their presence. To avoid this problem, detectives must run the protocol with a frequency and unpredictability such that the cost of maintaining a Sybil exceeds an opportunistic Sybil's profit.

We denote the cost to enter a single identity in the application e. We assume that a non-malicious entity's total utility after entering a single identity — the behavior of an honest user — is greater than zero. Denote this initial utility u. Since she is not malicious, she gains nothing from additional identities other than the chance of a reward.

First, we evaluate an application in which the Informant protocol is run every round. We assume conservatively that the attacker is certain that she will be the winning respondent in each auction round. For each identity the attacker adds, she pays e and expects to receive a reward of r after participating in the protocol. So if she enters two identities into the application she expects a total utility of $u - e + r$ versus u if she only enters one identity. Thus it is not profitable for non-malicious entities to form Sybils when $u - e + r \leq u$, that is when $r < e$, the offered reward is less then the per-identity participation cost.

The detective can avoid additional Sybils by keeping her rewards this low, but she will not be able to detect some Sybils in this case. She can increase her ability to detect the overall prevalence of Sybil attacks by participating only occasionally, but offering higher rewards when she does so. If the maximum reward the questioner is willing to offer is r_{max}, she can avoid introducing non-malicious Sybils by randomly running the protocol at most every r_{max}/e rounds, so that the expected per-round reward remains less that c. Even though the protocol is run less frequently, it is still worthwhile for legitimate Sybils to answer when it is run.

If a number of independent questioners are running Sybil detection protocols, a questioner who does not want to increase Sybil attacks should make sure that the *total* expected per round reward does not exceed e.

Note that in each case, running the Sybil detection protocol may increase the prevalence of Sybil attacks made by weakly malicious entities. If an attacker gains utility greater than zero, but less than e, from an additional identity introduced into the application, then a reward less than e may be enough to make the total per-round return positive. This tendency can be avoided by keeping the per-round reward as low as possible.

5.2   Discussion

Legal Concerns. If Sybil attackers are engaged in illegal activity, they may be reluctant to participate in Informant for fear that Sybil identities could be linked to their real-world identity. If this proves to be a concern, identities can be

registered using an anonymity system such as Tor [10] and paid with anonymous electronic cash.

Electronic Cash. Informant relies on some form of anonymous payment. The lack of practical anonymous electronic cash is the most significant obstacle to implementation.

Entry Fees versus Rewards. In general, the designer of a network application has two competing interests: to maximize participation in the protocol, which requires keeping entry fees low, and to maximize Sybil detection, which requires setting a high reward and thus a high entry fee as well (to avoid opportunistic Sybil attacks.) Where entry fees are low and rewards must be high, the detective can choose to offer Informant unpredictably and only very occasionally. Informant must only be announced after all identities have registered for a particular round of the application, so that opportunistic Sybil attackers must participate and pay the entry fees each round in the hopes of receiving the reward eventually.

6   Conclusion

We have designed and analyzed a novel, economic approach to Sybil attack detection protocol called *Informant*. We have proven the optimal strategies for each participant. The informant will accept the game if and only if she is Sybil with a low opportunity cost, and the target will cooperate if and only if she is identical to the informant. Our use of a Dutch auction ensures the minimum possible reward that will still reveal a Sybil attacker. While previous approaches have focused on physical tokens, such as radios [8,25] or clock skew [26], our approach is more general and not limited to a specific application. Given that the Sybil attack is not preventable without centralized verification of unique identity — all but impossible on a large scale — detection is crucial for protecting p2p applications.

References

1. Douceur, J.: The Sybil Attack. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, Springer, Heidelberg (2002)
2. Mathur, G., Padmanabhan, V.N., Simon, D.R.: Securing routing in open networks using secure traceroute. Tech Rep MSR-TR-2004-66, Microsoft Research (2004)
3. Castro, M., Druschel, P., Ganesh, A.J., Rowstron, A.I.T., Wallach, D.S.: Secure routing for structured peer-to-peer overlay networks. In: OSDI (2002)
4. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The Eigentrust algorithm for reputation management in P2P networks. In: Proc. WWW Conf., pp. 640–651 (2003)
5. Jelasity, M., Montresor, A., Babaoglu, O.: Towards Secure Epidemics: Detection and Removal of Malicious Peers in Epidemic-Style Protocols. Technical Report UBLCS-2003-14, University of Bologna (2003)
6. Levien, R.L.: Attack Resistant Trust Metrics. PhD thesis, UC Berkely (2004)

 7. Perrig, A., Stankovic, J., Wagner, D.: Security in wireless sensor networks. Commun. ACM 47(6), 53–57 (2004)
 8. Newsome, J., Shi, E., Song, D., Perrig, A.: The Sybil attack in sensor networks: Analysis & Defenses. In: Proc. IPSN Intl. Symp., pp. 259–268 (2004)
 9. Karlof, C., Wagner, D.: Secure routing in wireless sensor networks: Attacks and countermeasures. Ad hoc Networks Journal 1(2–3), 293–315 (2003)
10. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: Proc. USENIX Security Symposium (2004)
11. Cox, L., Noble, B.: Pastiche: Making backup cheap and easy. In: Proc. USENIX Symposium on Operating Systems Design and Implementation (2002)
12. Adar, E., Huberman, B.A.: Free riding on gnutella. First Monday 5(10) (2000)
13. Ntarmos, N., Triantafillou, P.: SeAl: Managing Accesses and Data in Peer-to-Peer Sharing Networks. In: Proc. P2P Computing, pp. 116–123 (August 2004)
14. Ngan, T.W.J., Wallach, D.S., Druschel, P.: Incentives-compatible peer-to-peer multicast. In: Proc. P2PEcon Workshop (2004)
15. Anagnostakis, K., Greenwald, M.: Exchange-Based Incentive Mechanisms for Peer-to-Peer File Sharing. In: Proc. ICDCS (2004)
16. Acquisti, A., Dingledine, R., Syverson, P.: On the Economics of Anonymity. In: Wright, R.N. (ed.) FC 2003. LNCS, vol. 2742, Springer, Heidelberg (2003)
17. Margolin, N.B., Levine, B.N.: Quantifying and discouraging sybil attacks. Tech Rep 2005-67, University of Massachusetts Amherst (2005)
18. Margolin, N.B., Wright, M., Levine, B.N.: Analysis of an incentives-based protection system. In: Proc. ACM Digital Rights Management Workshop (2004)
19. Shneidman, J., Parkes, D.C.: Rationality and self-interest in peer to peer networks. In: Kaashoek, M.F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735, Springer, Heidelberg (2003)
20. Margolin, N.B., Wright, M., Levine, B.N.: SPIES: Secret Protection Incentive-based Escrow System. In: Proc. P2PEcon Workshop (2004)
21. Cheng, A., Friedman, E.: Sybilproof reputation mechanisms. In: Proc. P2PEcon Workshop, pp. 128–132 (2005)
22. Čapkun, S., Hubaux, J.P.: BISS: Building secure routing out of an incomplete set of secure associations. In: Proc. ACM Wireless Security Conf., pp. 21–29 (2003)
23. Srivatsa, M., Liu, L.: Vulnerabilities and security threats in structured overlay networks: A quantitative analysis. In: Yew, P.-C., Xue, J. (eds.) ACSAC 2004. LNCS, vol. 3189, pp. 252–261. Springer, Heidelberg (2004)
24. Awerbuch, B., Scheideler, C.: Group Spreading: A Protocol for Provably Secure Distributed Name Service. In: Díaz, J., Karhumäki, J., Lepistö, A., Sannella, D. (eds.) ICALP 2004. LNCS, vol. 3142, pp. 183–195. Springer, Heidelberg (2004)
25. Piro, C., Shields, C., Levine, B.N.: Detecting the Sybil Attack in Ad hoc Networks. In: Proc. IEEE/ACM SecureComm. (2006)
26. Kohno, T., Broido, A., Claffy, K.C.: Remote physical device fingerprinting. IEEE Trans. Dependable Sec. Comput. 2(2), 93–108 (2005)
27. Yokoo, M., Sakurai, Y., Matsubara, S.: The effect of false-name bids in combinatorial auctions. Games and Economic Behavior 46(1), 174–188 (2004)
28. Rubin, S., Christodorescu, M., Ganapathy, V., Giffin, J.T., Kruger, L., Wang, H., Kidd, N.: An auctioning reputation system based on anomaly. In: Proc. ACM conference on Computer and Communications Security, pp. 270–279 (2005)
29. Osborne, M.J., Rubinstein, A.: A Course In Game Theory. MIT Press, Cambridge (1994)

30. von Ahn, L., Blum, M., Hopper, N., Langford, J.: CAPTCHA: Using hard AI problems for security. In: Biham, E. (ed.) Advances in Cryptology – EUROCRPYT 2003. LNCS, vol. 2656, pp. 294–311. Springer, Heidelberg (2003)
31. Nielson, S.J., Crosby, S.A., Wallach, D.S.: A taxonomy of rational attacks. In: Castro, M., van Renesse, R. (eds.) IPTPS 2005. LNCS, vol. 3640, Springer, Heidelberg (2005)
32. Cornelli, F., Damiani, E., Samarati, S.: Implementing a reputation-aware gnutella servent. In: Proc. of Intl. Workshop on Peer to Peer Computing (2002)
33. Marti, S., Garcia-Molina, H.: Limited reputation sharing in p2p systems. In: Proc. of the 5th ACM conference on Electronic commerce (2004)
34. Maniatis, P., et al.: Preserving peer replicas by rate-limited sampled voting. In: Proc. ACM SOSP, pp. 44–59 (2003)
35. Vishnumurthy, V., Chandrakumar, S., Sirer, E.G.: KARMA: A secure economic framework for p2p resource sharing. In: Proc. P2PEcon Workshop (2003)

Dynamic Virtual Credit Card Numbers

Ian Molloy[1], Jiangtao Li[2], and Ninghui Li[1]

[1] Purdue University
{imolloy,ninghui}@cs.purdue.edu
[2] Intel Corporation
jiangtao.li@intel.com

Abstract. Theft of stored credit card information is an increasing threat to e-commerce. We propose a dynamic virtual credit card number scheme that reduces the damage caused by stolen credit card numbers. A user can use an existing credit card account to generate multiple virtual credit card numbers that are either usable for a single transaction or are tied with a particular merchant. We call the scheme dynamic because the virtual credit card numbers can be generated without online contact with the credit card issuers. These numbers can be processed without changing any of the infrastructure currently in place; the only changes will be at the end points, namely, the card users and the card issuers. We analyze the security requirements for dynamic virtual credit card numbers, discuss the design space, propose a scheme using HMAC, and prove its security under the assumption the underlying function is a PRF.

Keywords: e-commerce, credit card theft.

1 Introduction

Credit cards are one of the most widely used payment mechanisms for both business-to-consumer and business-to-business commerce today. Credit card transactions account for billions of dollars in transactions daily [24], and these transaction records are often stored in various kinds of databases. Many e-commerce websites store credit card information for user convenience, as users will use these sites multiple times over a period time and would prefer not to enter the credit information for each transaction. Examples of such sites include PayPal, online shopping websites such as Amazon.com, and online travel sites such as Expedia. Online merchants may also keep records of credit card numbers for dealing with charge-backs and other disputes. Credit card processing centers will also store credit card numbers and transactions in an attempt to detect fraud. Anomalies in purchase characteristics such as amounts, retailers, frequencies, and locations can be an indication of fraud. Detecting these anomalies more quickly can be beneficial to both the cardholder and card issuer. Other organizations such as hotels, will store credit card numbers for liability from damages and incidentals.

The extensive databases kept by numerous parties quickly become highly desirable targets for those wishing to steal credit card numbers and commit fraud.

There have been several high-profile cases in recent years. For example, in 2001 attackers stole the customer records (including credit card information) of the online merchant Bibliofind, a subsidiary of Amazon.com [11]. In 2005 attackers broke into credit card processing center CardSystems Solutions Inc. and stole over 40 million credit card numbers [14]. Not all losses are the result of an online attack. Recently, stolen laptops have resulted in the loss of credit card numbers for 243,000 Hotels.com customers [1] and 80,000 Department of Justice employees [25].

In this paper, we propose a dynamic virtual credit card number scheme that reduces the damage caused by theft of stored credit card information. A user can use an existing credit card to generate a "virtual credit card (VCC) number" that is restricted in a number of ways. For example, it may be usable for a single transaction, or be linked with a particular merchant and have a lower credit limit and a shorter expiration date than the actual card. Such a VCC number can be generated using devices carried by the user, e.g., a cell phone or a PDA, without online contact with the card issuing bank. In our scheme, VCC numbers have the same format as normal credit card numbers. Merchants should be able to process a transaction with a VCC number in the same manner they use today; no change to their existing databases and applications is needed. Only the end points, i.e., the cardholders and the card issuers, need to be aware that a VCC is used. We also point out that a card holder can still use the actual card the old fashioned way. Our design aims at facilitating deployment. We have implemented a prototype for generating VCC numbers using Java 2 MicroEdition (J2ME) that runs on MIDP2.0 compliant cell phones. We have tested our MIDlet on Sony Ericsson z520a and Nokia 6102i model phones.

Several credit card issuers (CitiBank, Discover, and MBNA) already offer services similar to the concept of VCC. However, they all require users to install software onto a computer and communicate with the credit card issuer to get a new VCC number.

The rest of this paper is organized as follows. We review current attempts to secure credit card transactions online in Section 2. In Section 3, we analyze the necessary security properties for a VCC scheme and examine the solution space. We present our approach and discuss real-world considerations in Section 4, and give proofs of security in Section 5. We conclude with Section 6.

2   Related Work

There have been several attempts to reduce the usefulness of stolen card numbers. One widely adopted solution is security codes, such as the card verification value (CVV)[1] which is stored on the magnetic strip, and the CVV2, which is not. These are three- or four-digit cryptographic checksums that can validate the authenticity of a card for card-present and card-not-present transactions, such as online, mail-order, and telephone transactions. Merchants can ask for the CVV2

---

[1] The card verification value goes by different names to different credit card companies.

code during transactions, but are forbidden from storing them in their data-bases [23]. While CVV2 required a change to the card acceptor infrastructure, it is now ubiquitous.

CVV2 doesn't provide a perfect solution. Not all merchants or card issuers require CVV2 to approve a transaction. Flaws in an online processing center or merchant may allow attackers to gain CVV and CVV2 codes that are stored either inadvertently or temporarily while awaiting authorization. Finally, similar checksum codes used in Cartes Bancaires cards has been compromised [9], and other vulnerabilities have been found in ATM cards in the past [2].

A second approach requires the cardholder to enter an additional username and password for online transactions. Two examples are "Verified by Visa" for Visa credit cards [22] and "MasterCard SecureCode" for MasterCard credit cards [15]. These solutions require changes to the card acceptor infrastructure, which are not yet commonplace. These schemes do not work with telephone or mail-order transactions, hindering usage.

Another solution is to use proxy or virtual transaction numbers instead of the real credit card number [12]. This scheme has been developed by Orbiscom and is in use by MBNA, CitiBank and Discover [21,10,8]. When a cardholder wishes to make a transaction, she requests a temporary card number, and possibly binds some transaction parameters. The card issuer generates a new number not currently in use, links it the cardholder's account allowing reverse lookups, and returns the proxy number. A similar proposal is SecureClick [19] which requires a card issuer issued nonce for each transaction that acts as pre-approval.

Singh et al. developed a grammar-based method for generating offline credit card numbers [20]. The scheme is essentially a one-time password scheme where each new password is used as the next credit card number. Synchronization becomes challenging as credit card usage is asynchronous. Furthermore, the se-curity of the scheme in [20] is based on the difficulty of finding a string that is accepted by an unknown grammar. While this problem is intractable in the worst case, it is unclear whether it is computationally expensive in the average case, which is what is needed for cryptographic security.

Rubin and Wright [17] proposed an offline scheme that used arbitrary finite domain encryption methods [7] to encode a set of restrictions the cardholder wishes to place on their temporary card number. A cardholder first chooses sev-eral restrictions such as amount, expiration date, good or service type, merchant name, and timestamp and then encrypts the sequence of restriction parameters, using the result as the account number. Because the ciphertext space is very limited (around 29- to 39-bits), the scheme suffers from the following two prob-lems. One is that some parameters (e.g., merchant name) must be encoded in a very compact form, resulting in the same encoding being valid in many settings. The other problem is that the probability that a random ciphertext may be de-crypted into a valid combination of parameters may be quite high. Our proposed approach solves this problem by using MAC, rather than encryption, to generate the temporary card number. Further, it is noted in [7] that enciphering messages

of (around 29- to 39-bits) falls within a gap where there is no known solution that is both efficient and secure.

3   Problem Description

3.1   How Credit Cards Work

Because we would like a solution that does not require any changes to the current infrastructure and protocols, we first examine how credit card processing currently works.

Credit Card Number Format. A credit card number is a maximum of nineteen digits that can be broken into three pieces: issuer, account number, and checksum. Most numbers (including Visa, MasterCard, and Discover) are sixteen digits while American Express numbers are fifteen. The first six digits make up the issuing bank and the last single digit is the Luhn check digit. The Luhn code is a one-digit checksum of the credit card number which can be calculated and verified by anyone [26]. This yields a maximum of twelve digits for the account number. As most credit cards have 16 or 15 digits, the limit of the account number is 9 or 8, respectively. This is a limit we have to take into consideration as we want to generate virtual card numbers that work with the current infrastructure.

Parties in Credit Card Processing. Credit card transactions involve several parties. The three of interest are: *cardholder*, *card issuer*, and *merchant* (card acceptor).

Credit Card Processing Parameters Figure 1 shows the card processing steps. The cardholder sends *credit card information* including name, billing address, account number, expiration date, and CVV2 to the merchant. The merchant send this together with *merchant information*, (which is configured when a business gains merchant status, and includes the merchant's bank and account numbers, merchant name and number), and *transaction information* (including the date and time, the amount of the transaction, a merchant-specified order number, and often specific information such as the point of sale device used) to the issuer.
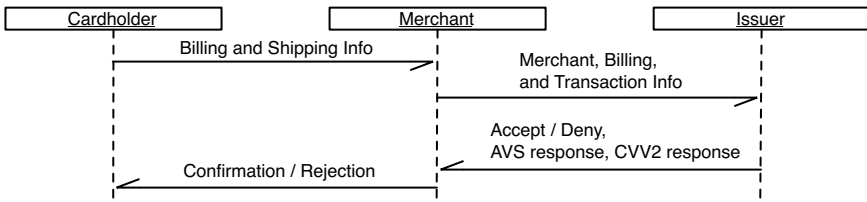
Fig. 1. Traditional Credit Card Processing

The issuer may return several pieces of information to the merchant (e.g. authorization or rejection, address verification service (AVS) and CVV2 match responses). AVS tells the merchant how well the billing address supplied by the cardholder matches the billing address on record. A rejection notice overrides any decision the merchant may make to accept the transaction, while the treatment of AVS and CVV2 responses are up to the discretion of the merchant [23].

3.2   Security Properties for VCC Schemes

In a VCC scheme, the cardholder is able to generate a VCC number that is bound to a single transaction, or with a single merchant and maximum transaction amount. We require such a scheme to have the following properties.

1. *Complete* - Any cardholder can generate a VCC number from her credit card account number, the transaction information and/or limitation on usage of the VCC, and any other information the cardholder may have.
2. *Sound* - Given the public transaction information and VCC number, the card issuer is able to uniquely identify the associated account.
3. *Account Hiding* - Knowing the public transaction information and the VCC number, no adversary has a non-negligible advantage in recovering the original credit card account number.
   This is motivated by the original motivation of having VCC numbers, that is, to hide the actual credit card numbers.
4. *Forgery Resistant* - Knowing an account number and some virtual credit card transaction information associated with the account, no adversary has a non-negligible advantage in forging a valid VCC associated with this account.
   Even with a VCC scheme, the original credit card number may still be stolen, because customers may choose to use the original card number in some transactions and because of card loss. We would like to ensure that a stolen card does not enable one to easily construct valid VCC numbers.

In other words, we are concerned with two types of threats: finding out the cardholder's original account number through transaction information involving VCCs, and the generation of valid VCC numbers if an attacker obtains account numbers. We are not concerned with malicious merchants who attempt to abuse a VCC (such as multiple submissions), as such threats are dealt with by existing dispute resolution procedures and laws.

Note that when VCC numbers are generated online by the card issuer, the sound and complete properties are no longer needed. One solution that would satisfy the other two properties is to randomly generate account numbers until encountering one that has not already been used. Such a solution would not work when we allow VCC numbers to be generated offline.

3.3   Examining the Solution Space

Our desired security properties impose limitations on the potential solution space. The sound property states that the card issuer *must* be able to recover

the account number from the VCC transaction information, whereas the account hiding property states that an attacker *must not* be able to recover the account number. This indicates that the card issuer must know something that the attacker doesn't know. Such a secret can be shared between the cardholder and the card issuer, or known only by the card issuer.

One potential solution is to use the account number to derive a secret key, and to use some keyed-MAC of the transaction information to generate the VCC number and the CVV2 code for the VCC. There are two problems with this solution. First, this violates the forgery resistant property, because knowing the account number enables one to forge virtual credit card numbers. Second, the account hiding property can also be broken easily, because an attacker can perform an exhaustive search over the space of valid account numbers. Credit card numbers are highly structured and have a small space, making exhaustive search attacks feasible.

One attempt to fix the above solution is to add additional secret information that is currently shared between a card issuer and a cardholder, such as social security number or mother's maiden name, to derive a secret key. Under this design, when an attacker obtains an account number, the attacker would still need to know additional information to be able to construct a virtual credit card number. However, this design suffers from another weakness. An attacker who somehow obtains one's account number can use an exhaustive search attack to try to recover other secrets that are used in the process of generating the key. Such a design protects account numbers at the cost of increasing danger of revealing this other information, which is arguably more sensitive than credit card numbers. We thus choose not to adopt this design.

Another potential approach is to use public key cryptography. This has the advantage of eliminating the need for a shared secret between each cardholder and the card issuer. The card issuer would have a public key, and the cardholders would encrypt their account numbers with the issuer's public key.This does not satisfy the forgery resistant property, as anyone knowing the account number and the issuer's public key can generate a valid VCC number. Also, most public key cryptography systems produce ciphertexts much larger than the credit card space, typically on the order of 160-1024 bits and above. Truncating the result would make decryption infeasible.

We thus decided to use a design where each cardholder shares a secret with the card issuer for each account, and this secret is beyond the long-term secrets (such as an SSN or mother's maiden name) already shared between a card issuer and a cardholder.

4   Our Proposed Scheme

4.1   A Dynamic Virtual Credit Card Scheme

We assume the cardholder already has an account with the card issuer. The card issuer knows the cardholder's name and address, which we shall call the billing information, B. The card issuer will have provided the cardholder with

an account number, C. Finally, they negotiate a shared secret, such as a password P. Note that the cardholder may already have a password through web access to their account information. A bank can choose to use this password or a different password for the VCC scheme. The advantage of using one password is ease of use. The disadvantage is that if an attacker gets access to the VCC number, then the attacker can use dictionary attack to try to recover the password.

In the description of our scheme below, we use two functions: H, a function that generates a key from a shared secret, and F, which can be thought as a keyed MAC function. This description is for generating a one-time VCC number, which can be used for a single transaction. We will describe how to generate a usage-limited VCC number in Section 4.2.

Generation. The cardholder will:

- Choose an expiration date, E, for the virtual card. This is usually the current month.
- Generate a string for the transaction, $\sigma = E||B||M||T$, where M is merchant information, and T is transaction amount.
- Generate the shared key $K = \mathsf{H}(C||P)$
- Calculate $V = \mathsf{F}_K(\sigma) \bmod 10^n$, in which n is the length of the account number plus the length of the CVV2 code.
- Divide V into V_1 and V_2. Prefix the card issuer code to V_1 and append a valid Luhn code to get the VCC number. V_2 is the CVV2 code.

Verification. To verify that a merchant submitted VCC number is valid for a given transaction, the card issuer will:

- Identify the original account C', using the billing information (name and address) supplied in the AVS.
- Find the password P' associated with the account C', and calculate the shared secret $K' = \mathsf{H}(C'||P')$.
- Calculate $V' = \mathsf{F}_{K'}(\sigma') \bmod 10^n$, using σ' from the merchant supplied values.
- If the submitted VCC number and CVV2 code match V', then process the transaction as usual, otherwise reject the transaction.

The above scheme is complete, as any cardholder can generate a VCC number. It is sound assuming that an account number can be uniquely identified given the name and address of a cardholder. In section 4.2 we discuss how to relax this restriction. In section 5.4 we show that using any pseudorandom function for F and H will satisfy the account hiding and forgery resistant properties.

We have written a prototype implementation of the card generation process in J2ME that is lightweight, fast, and capable of running on a wide variety of hardware, including cellular phones. In our current implementation, we use $\mathsf{SHA1}$ for the function H and $\mathsf{HMAC}\text{-}\mathsf{SHA1}$ for the function F.

4.2   Real World Considerations

Multi-use VCC Numbers In some situations, a cardholder may wish to generate a VCC number that can be used for multiple transactions with one merchant, for example PayPal or Amazon.com's 1-Click. In this case, the cardholder may want to set a credit limit lower than the limit of the account. As we would like to use the existing infrastructure, the credit limit chosen by the cardholder must be encoded in the VCC number.

Since we have a limited message space, we cannot allow all possible limits. Our design is to use one digit ℓ (we call this the VCC type) to encode whether this is a one-use card number, and if not, what is the credit limit. For example, $\ell = 0$ means single transaction, $\ell = 1$ means a limit of \$50, $\ell = 2$ means a limit of \$100, and so on. Using one digit, we can accommodate 9 different credit limit values. The VCC type digit can be in the VCC number (or the CVV2 code, if almost all merchants use it). Note that this digit must be appropriately encrypted, so that the credit limit cannot be learned by an attacker who gets the VCC number. To accommodate this, we change the design so that the VCC type digit is not generated from $V = F_K(\sigma) \bmod 10^n$. Instead, we use bits in $F_K(\sigma)$ that have not been used in generating V to randomly select a permutation π over $\mathbb{Z}_{10}$, and use $\pi(\ell)$ as the VCC type digit.

Collisions Between Actual and Virtual Credit Numbers. The sets of possible actual and virtual credit card numbers do not need to be disjoint, under the condition that the VCC scheme is used only when AVS information is provided to the card issuer. If a merchant does not provide AVS information, we must assume we are given an actual card number. If AVS information is provided, then we assume we can uniquely identify the cardholder's account information. There are now two possibilities: either the card number and CVV match the real card, or they do not. If they do not, we process the card as a VCC number. If they do match, then the card number given was either actual, or $C \equiv V$ for the given transaction; neither case violates the soundness or completeness properties, and a second account cannot be incorrectly charged. The attack in which an adversary provides the AVS information of someone else's account and tries to generate a valid VCC number is no easier than the attack of guessing someone else's credit card number and using it, and can be handled by current dispute resolution procedures.

Non-Unique Name-Address Pairs. Our scheme relies on the assumption that each name-address pair uniquely identify an account number. When this is not possible, then the probability that the VCC number generated using a second account also matches is about $1/10^n$, where n is the number of digits used in the VCC scheme. There are several approaches to enable us to relax this assumption. One approach is to reject a VCC when a collision occurs, in which case the client generates another VCC, taking a sequence number as an additional input. The probability that a collision occurs after a few rounds is extremely small. Another approach is to change the scheme so that the CVV2 code of the actual credit card is used to as the CVV2 code for the VCC. The bank thus only needs to

ensure that name, address, and the CVV2 code together uniquely identify an account. This allows one name-address pair to have multiple accounts.

5   Security

We now present formal definitions of security for a virtual credit card scheme and prove our proposed scheme is secure.

5.1   Security Model

We use the following notations. We say that $\mu(k)$ is a negligible function, if for every polynomial $p(k)$ and for all sufficiently large k, $\mu(k) < 1/p(k)$. We say $\nu(k)$ is overwhelming if $1 - \nu(k)$ is negligible. If S is a probability space, then the probability assignment $x \leftarrow S$ means that an element x is chosen at random according to S. If S is a finite set, then $x \leftarrow S$ denotes that x is chosen uniformly from S. Let A be an algorithm, we use $y \leftarrow A(x)$ to denote that y is obtained by running A on input x. In the case that A is deterministic, then y is unique; if A is probabilistic, then y is a random variable. Let p be a predicate and $A_1, A_2, \ldots, A_n$ be n algorithms then $\Pr\left[\{x_i \leftarrow A_i(y_i)\}_{1 \le i \le n} : p(x_1, \cdots, x_n)\right]$ denotes the probability that $p(x_1, \cdots, x_n)$ will be true after running sequentially algorithms $A_1, \ldots, A_n$ on inputs $y_1, \ldots, y_n$.

   We next describe our security model for VCC schemes. Let $C \in \{0,1\}^{\ell_c}$ be the original credit card number and $V \in \{0,1\}^{\ell_c}$ be the virtual credit card number, where ℓ_c is the bit-length of the credit card number. Let $A \in \{0,1\}^{\ell_a}$ be the account information, $B \in \{0,1\}^{\ell_b}$ be the customer billing information, $T \in \{0,1\}^{\ell_t}$ be the transaction information, and $S \in \{0,1\}^{\ell_s}$ be the secret that is known to the cardholder and the bank; where ℓ_a is the (maximum) length of the account information, ℓ_b is the length of the billing information, ℓ_t is the length of the transaction information, and ℓ_s is the length of the secret. The secret S has two parts: the first part is the original credit card C, and the second part is a password $P \in \{0,1\}^{\ell_p}$, where ℓ_p is the length of the password and $\ell_s = \ell_c + \ell_p$.

   There are three deterministic algorithms in the VCC scheme: Identify, VirGen, and Verify. The algorithm Identify : $\{0,1\}^{\ell_b} \times \{0,1\}^{\ell_c} \rightarrow \{0,1\}^{\ell_a}$ is the personal account identification algorithm, i.e., given B and V, Identify(B, V) outputs an account information A. The algorithm VirGen : $\{0,1\}^{\ell_t} \times \{0,1\}^{\ell_s} \rightarrow \{0,1\}^{\ell_c}$ is the virtual credit card generation algorithm, i.e., given T and S, VirGen(T, S) outputs a virtual credit card V. The algorithm Verify : $\{0,1\}^{\ell_t} \times \{0,1\}^{\ell_s} \times \{0,1\}^{\ell_c} \rightarrow \{$true, false$\}$ is the virtual credit card verification algorithm, i.e., given T, S, V, Verify(T, S, V) outputs either true or false.

   The virtual credit card scheme has the following phases:

- Customer-Bank initialization: In this phase, the customer first sends the billing information B to the bank. The bank then creates the original credit card number C and the account information A for the customer. The bank and customer jointly choose the password P and set the secret $S = C||P$.

- Customer-Merchant interaction: In this phase, the customer and merchant jointly determine the transaction information T. The customer then computes $V = \mathsf{VirGen}(T, S)$, and sends V and B to the merchant.
- Merchant-Bank interaction: In this phase, the merchant sends T, B, and V to the bank. The bank uses $\mathsf{Identify}(B, V)$ to identify the customer account A, then obtains the shared secret S based on A, and finally computes $\mathsf{Verify}(T, S, V)$. If the output of the Verify algorithm is false, the bank rejects the transaction.

5.2 Security Properties

The virtual credit card must satisfy the sound property, the complete property, the security against forgery property, and the security against account recovery property:

The *sound property* can be stated as:

$$\Pr\left[\begin{array}{c} B \leftarrow \{0,1\}^{\ell_b}, T \leftarrow \{0,1\}^{\ell_t}, S \leftarrow \{0,1\}^{\ell_s}, V \leftarrow \mathsf{VirGen}(T, S), \\ x \leftarrow \mathsf{Identify}(B, V) : x = \bot \end{array} \right] = 0$$

where $\bot$ is a symbol that represents empty output. In other words, given the billing information and the virtual credit card number, we can always identity the corresponding account number.

The *complete property* can be stated as:

$$\Pr\left[A \leftarrow \{0,1\}^{\ell_a}, T \leftarrow \{0,1\}^{\ell_t}, S \leftarrow \{0,1\}^{\ell_s}, x \leftarrow \mathsf{VirGen}(T, S) : x = \bot \right] = 0$$

In other words, we can always generate a VCC number given the transaction information and the secret.

The *secure against forgery* can be stated as follows. We consider forgery under adaptive chosen-message attacks. Our scheme is secure against forgery if an adversary cannot win the following game between a challenger and the adversary:

1. Setup. The challenger runs a setup algorithm to output an account information A and a secret S. The challenger sends the account information to the adversary.
2. Queries. Proceeding adaptively, the adversary requests VCC numbers for at most q messages (transactions) of her choice $T_1, \ldots, T_q \in \{0,1\}^{\ell_t}$. The challenger responds to each query with a virtual credit card number $V_i = \mathsf{VirGen}(T_i, S)$.
3. Outputs. Eventually, the adversary outputs a pair (T, V) and wins the game if T is not any of $T_1, \ldots, T_q$ and $\mathsf{Verify}(T, S, V) = \mathsf{true}$.

Since the space for the virtual credit card is rather small, the adversary can do random guessing. Our security definition (in the following equation) states that the adversary cannot do better than random guessing:

$$\Pr\left[\begin{array}{c} S \leftarrow \{0,1\}^{\ell_s}, (T, V) \leftarrow \mathcal{A}(T_1, V_1, \ldots, T_q, V_q) : \\ \mathsf{Verify}(T, S, V) = \mathsf{true} \end{array} \right] \leq 2^{-\ell_c} + \mu(t).$$

where $\mu(t)$ is a negligible function in time t.

The *secure against account recovery property* can be stated as follows. We consider account recovery under chosen-message attacks. Our scheme is secure against account forgery if an adversary cannot win the following game between a challenger and the adversary:

1. Setup. The challenger runs a setup algorithm to output an account information A and a secret S, which comprises of a credit card number C and a password P. The challenger sends the account information to the adversary.
2. Queries. Proceeding adaptively, the adversary requests VCC numbers for at most q messages (transactions) of her choice $T_1, \ldots, T_q \in \{0,1\}^{\ell_t}$. The challenger responds to each query with a VCC number $V_i = \mathsf{VirGen}(T_i, S)$.
3. Outputs. Eventually, the adversary outputs C, the original credit card number.

Since the space for the credit card number is rather small, the adversary can do random guessing. Our security definition (in the following) states that the adversary cannot do better than random guessing:

$$\Pr \left[\begin{array}{l} C \leftarrow \{0,1\}^{\ell_c}, P \leftarrow \{0,1\}^{\ell_p}, S = C||P, \\ C' \leftarrow \mathcal{A}(T_1, V_1, \ldots, T_q, V_q) : C' = C \end{array} \right] \leq 2^{-\ell_c} + \mu(t)$$

where $\mu(t)$ is a negligible function in time t.

5.3   Our Abstracted Scheme

We now give an abstract of our scheme that is presented in Section 4, then prove our scheme is secure in the next subsection. Let $H : \{0,1\}^{\ell_s} \rightarrow \{0,1\}^{\ell_k}$ be a collision-free hash function, and $F : \{0,1\}^{\ell_k} \times \{0,1\}^{\ell_t} \rightarrow \{0,1\}^{\ell_k}$ be a family of functions that can be modeled as a pseudorandom function (PRF), where ℓ_s is the length of the secret, ℓ_t is the length of the transaction information, and ℓ_k is the key length of F. Here we also use ℓ_k to denote the output length of the pseudorandom function F and the hash function H.

– $\mathsf{Identify}(B, V)$: This algorithm takes $B \in \{0,1\}^{\ell_b}$ and $V \in \{0,1\}^{\ell_c}$ as input, and outputs the account information A. Here, we assume the bank keeps a database of the clients' information, including billing information and account information. Given the billing information B, the bank can lookup the database to identify the corresponding account information A.
– $\mathsf{VirGen}(T, S)$: This algorithm takes the transaction information T and the secret S as input, and outputs the VCC number V. It performs the following steps:
   1. Computes $K = H(S)$.
   2. Sets V to be the last ℓ_c bits of $F_K(T)$.[2]

---

[2] Note that in our proposed scheme, $V = F_K(T) \bmod 10^n$ where n is the number of digits in the virtual credit card. In our security model, since we assume V to be a value of length ℓ_c, we set $V = F_K(T) \bmod 2^{\ell_c}$. This simplification will not affect our security proof.

– Verify(T, S, V): This algorithm takes the transaction information T, the secret S, and the virtual credit card number V as input, and outputs either true or false. It performs the following steps:
1. Computes $V' = \mathsf{VirGen}(T, S)$.
2. If $V = V'$ outputs true, otherwise outputs false.

In our proposed scheme, H is SHA1 and F is an HMAC or a CBC-MAC scheme. HMAC was originally shown to be a PRF under the assumption that the underlying hash function was collision resistant [4], and later if the underlying hash function was a PRF [3]. It should be noted that recent work has shown HMAC to not be a PRF when instantiated with certain hash functions, such as MD4, MD5, SHA0 and SHA1 [13]. While they were able to produce a distinguisher for HMAC-SHA1 using differentials discovered by Biham et a. and Wang et al., they were only able to do so when SHA1 was reduced to 43 rounds, and a probability of $2^{-73.4}$ (more than the general attack of 2^{-80}) and a data complexity of $2^{154.9}$ (more than the general attack of 2^{80}) [16]. We do not feel their results adversely affect our work.

5.4 Security Proofs

We now prove that our virtual credit card scheme is secure under the definition in Section 5.1. The sound property of our scheme is guaranteed if the bank maintains a proper customer database. Our scheme is also complete as the algorithm VirGen always returns an output. We now focus on the secure against forgery property and the secure against account recovery property.

Theorem 1. *Our virtual credit card scheme is secure against forgery.*

Proof. Let $F : \{0,1\}^{\ell_k} \times \{0,1\}^{\ell_t} \to \{0,1\}^{\ell_k}$ be a pseudorandom function, we now build a family of functions $F' : \{0,1\}^{\ell_k} \times \{0,1\}^{\ell_t} \to \{0,1\}^{\ell_c}$ as follows: Given $K \in \{0,1\}^{\ell_k}$ and $T \in \{0,1\}^{\ell_t}$,

$$F'(K, T) = F(K, T) \bmod 2^{\ell_c}.$$

It is clear that $V = F'(K, T) = F'(H(S), T)$. In the next two claims, we first show that if F is a pseudorandom function, then F' is a pseudorandom function as well. We then show that if F' is a pseudorandom function, then our scheme is secure against forgery.

Claim. If $F : \{0,1\}^{\ell_k} \times \{0,1\}^{\ell_t} \to \{0,1\}^{\ell_k}$ is a pseudorandom function, then $F' : \{0,1\}^{\ell_k} \times \{0,1\}^{\ell_t} \to \{0,1\}^{\ell_c}$, which is defined above, is also a pseudorandom function.

Proof. To prove this claim, we first review the definition of pseudorandom functions. Pseudorandomness of the function family F measures the ability of a distinguisher to tell whether its given oracle is a random instance of F or a random function of $\{0,1\}^{\ell_t}$ to $\{0,1\}^{\ell_k}$. For a distinguisher $\mathcal{A}$, let

$$\mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{A}) = \Pr\left[f \leftarrow F : \mathcal{A}^f = 1\right] - \Pr\left[f \leftarrow \mathrm{Rand}^{\ell_t \to \ell_k} : \mathcal{A}^f = 1\right].$$

For any integer $q, t \geq 0$, let

$$\mathbf{Adv}_F^{\mathrm{prf}}(q, t) = \max\left\{\mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{A})\right\},$$

where the maximum is over all distinguishers $\mathcal{A}$ that make at most q oracle queries and use at most t running time. Intuitively, if F is a pseudorandom function, then F', the last ℓ_c bits of F, should also be a pseudorandom function. We prove this by showing that

$$\mathbf{Adv}_{F'}^{\mathrm{prf}}(q, t) \leq \mathbf{Adv}_F^{\mathrm{prf}}(q, t).$$

Let $\mathcal{A}$ be a distinguisher that is given an oracle for a function $f' : \{0, 1\}^{\ell_t} \to \{0, 1\}^{\ell_c}$. Assume $\mathcal{A}$ invoked at most q queries and ran at most t time. We can design a distinguisher $\mathcal{B}$ for F versus $\mathrm{Rand}^{\ell_t \to \ell_k}$ such that

$$\mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{B}) = \mathbf{Adv}_{F'}^{\mathrm{prf}}(\mathcal{A}).$$

Given a distinguisher $\mathcal{A}$, we can build a distinguisher $\mathcal{B}$ as follows. Recall that, given an oracle for a function $f : \{0, 1\}^{\ell_t} \to \{0, 1\}^{\ell_k}$, $\mathcal{B}$ must determine whether f is chosen randomly from F or from $\mathrm{Rand}^{\ell_t \to \ell_k}$.

1. When $\mathcal{A}$ asks its oracle for query T_i, for $i \in \{1, \ldots, q\}$, $\mathcal{B}$ queries its own oracle using T_i and obtains $f(T_i)$. $\mathcal{B}$ computes $f'_i(T_i) = f(T_i) \bmod 2^{\ell_c}$ and return $f'_i(T_i)$ back to $\mathcal{A}$.
2. If $\mathcal{A}$ outputs 1 then $\mathcal{B}$ returns 1, otherwise $\mathcal{B}$ returns 0.

We show that

$$
\begin{aligned}
\mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{B}) &= \Pr\left[f \leftarrow F : \mathcal{B}^f = 1\right] - \Pr\left[f \leftarrow \mathrm{Rand}^{\ell_t \to \ell_k} : \mathcal{B}^f = 1\right] \\
&= \Pr\left[\begin{matrix} f' \leftarrow F' : \\ \mathcal{A}^{f'} = 1 \end{matrix}\right] - \Pr\left[\begin{matrix} f \leftarrow \mathrm{Rand}^{\ell_t \to \ell_c}, f'(T) = f(T) \bmod 2^{\ell_c} : \\ \mathcal{A}^{f'} = 1 \end{matrix}\right] \\
&= \Pr\left[f' \leftarrow F' : \mathcal{A}^{f'} = 1\right] - \Pr\left[f' \leftarrow \mathrm{Rand}^{\ell_t \to \ell_c} : \mathcal{A}^{f'} = 1\right] \\
&= \mathbf{Adv}_{F'}^{\mathrm{prf}}(\mathcal{A})
\end{aligned}
$$

In the above equations, it is clear that if a function f is randomly chosen from $\mathrm{Rand}^{\ell_t \to \ell_k}$, then f', which outputs the last ℓ_c bits of f, is a random function from $\mathrm{Rand}^{\ell_t \to \ell_c}$. We finish the proof by showing:

$$\mathbf{Adv}_{F'}^{\mathrm{prf}}(q, t) = \max\left\{\mathbf{Adv}_{F'}^{\mathrm{prf}}(\mathcal{A})\right\} \leq \max\left\{\mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{B})\right\} = \mathbf{Adv}_F^{\mathrm{prf}}(q, t)$$

Claim. If $F' : \{0, 1\}^{\ell_k} \times \{0, 1\}^{\ell_t} \to \{0, 1\}^{\ell_c}$ is a pseudorandom function, then our scheme is secure against forgery.

Proof. In this proof, we show that if there exists a forger who can forge a virtual credit card, then we can build a distinguisher to distinguish F' from random functions. Our proof is similar to the proof in [6]. More formally, let

$$\mathbf{Adv}_{F'}^{\mathrm{vc}}(q,t) = \max \left\{ \Pr \left[\begin{array}{c} S \leftarrow \{0,1\}^{\ell_s}, (T,V) \leftarrow \mathcal{A}(T_1, V_1, \ldots, T_q, V_q): \\ \mathsf{Verify}(T,S,V) = \mathsf{true} \end{array} \right] \right\}$$

$$= \max \left\{ \Pr \left[\begin{array}{c} S \leftarrow \{0,1\}^{\ell_s}, (T,V) \leftarrow \mathcal{A}(T_1, V_1, \ldots, T_q, V_q): \\ F'(H(S),T) = V \end{array} \right] \right\}$$

We want to show that

$$\mathbf{Adv}_{F'}^{\mathrm{vc}}(q,t) \ \leq \ \mathbf{Adv}_{F'}^{\mathrm{prf}}(q,t) + 2^{-\ell_c}.$$

Let $\mathcal{A}$ be a forger who tries to forge a virtual credit card. Assume $\mathcal{A}$ invoked at most q queries and ran at most t time. We can design a distinguisher $\mathcal{B}$ for F' versus $\mathrm{Rand}^{\ell_t \to \ell_c}$ such that

$$\mathbf{Adv}_{F'}^{\mathrm{prf}}(\mathcal{B}) \ \geq \ \mathbf{Adv}_{F'}^{\mathrm{vc}}(\mathcal{A}) - 2^{-\ell_c}.$$

Given a forger $\mathcal{A}$, we can build a distinguisher $\mathcal{B}$ as follows. Recall that, given an oracle for a function $f' : \{0,1\}^{\ell_t} \to \{0,1\}^{\ell_c}$, $\mathcal{B}$ must determine whether f' is chosen randomly from F' or from $\mathrm{Rand}^{\ell_t \to \ell_c}$.

1. When $\mathcal{A}$ asks its oracle for query T_i, for $i \in \{1, \ldots, q\}$, $\mathcal{B}$ answers with $V_i = f'(T_i)$.
2. $\mathcal{A}$ outputs a (T,V) pair such that $T \notin \{T_1, \ldots, T_q\}$.
3. If $V = f'(T)$ then return 1 else return 0.

It is easy to see that

$$\Pr\left[f' \leftarrow F' : \mathcal{B} = 1\right] = \mathbf{Adv}_{F'}^{\mathrm{vc}}(\mathcal{A})$$
$$\Pr\left[f' \leftarrow \mathrm{Rand}^{\ell_t \to \ell_c} : \mathcal{B} = 1\right] \geq 2^{-\ell_c}$$

Subtract the above two equations, we obtain $\mathbf{Adv}_{F'}^{\mathrm{prf}}(\mathcal{B}) \geq \mathbf{Adv}_{F'}^{\mathrm{vc}}(\mathcal{A}) - 2^{-\ell_c}$. The above reduction shows that the probability of a successful forgery is less than the probability of distinguishing F' from a random function plus $2^{-\ell_c}$. Therefore, our scheme is secure against forgery if F' is a pseudorandom function.

Theorem 2. *Our virtual credit card scheme is secure against account recovery.*

Proof. Recall that the algorithm VirGen takes $T \in \{0,1\}^{\ell_t}$ and $S \in \{0,1\}^{\ell_s}$ as input, computes $K = H(S)$ and $V = F'_K(T)$, and outputs V. Let $S = C||P$, where C is the original credit card and P is a password. We want to show that no adversary can compute C with a probability more than random guessing. Note that the adversary can query the oracle multiple times with T_i and obtain the corresponding V_i, so that she can try to learn K then learn C. What we

prove next is that, even if the adversary learns K, the adversary cannot guess C correctly, i.e.,

$$\Pr\left[C \leftarrow \{0,1\}^{\ell_c}, P \leftarrow \{0,1\}^{\ell_p}, K = H(C\|P), C' \leftarrow \mathcal{A}(K) : C' = C\right] \leq 2^{-\ell_c} + \mu(t)$$

for any polynomial-time adversary $\mathcal{A}$, where $\mu(t)$ is a negligible function in t. This is quite obvious given H is a one-way hash function, that is,

$$\Pr\left[M \leftarrow \{0,1\}^*, K = H(M), M' \leftarrow \mathcal{A}(K) : M' = M\right] \leq \mu(t)$$

Given $K = H(C\|P)$, the adversary can either do a random guessing, i.e., pick $C' \leftarrow \{0,1\}^{\ell_c}$ or find the pre-image of K. Therefore, the overall success probability for the adversary is bounded by $2^{-\ell_c} + \mu(t)$.

6   Closing Remarks

Theft of stored credit card information is an increasing threat to e-commerce. We propose the concept of dynamic virtual credit card (VCC) numbers to mitigate this threat. Dynamic VCC numbers can be generated by credit card holders without online contact with the issuing bank. Using VCC numbers requires no change to the merchant's credit card processing infrastructure and reduces the damage caused by stolen credit card numbers. We have identified the security requirements for VCC schemes and proposed a scheme that offers flexibility as well as security. We have also discussed how to address issues related to real-world deployment of the scheme, and proved that our scheme is secure under commonly used cryptographic assumptions. We believe this is a viable solution to the problem of credit card information theft.

References

1. Hotels.com credit-card numbers stolen: CNN Money (June 2, 2006)
2. Anderson, R.: Why cryptosystems fail. Communications of the ACM 37(11), 32–40 (1994)
3. Bellare, M.: New proofs for NMAC and HMAC: Security without collision-resistance. Cryptology ePrint Archive, Report 2006/043 (2006), http://eprint.iacr.org/
4. Bellare, M., Canetti, R., Krawczyk, H.: Keying hash functions for message authentication. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, Springer, Heidelberg (1996)

5. Bellare, M., Garay, J., Hauser, R., Herzberg, A., Krawczyk, H., Steiner, M., Tsudik, G., Herreweghen, E.V., Waidner, M.: Design, implementation and deployment of the ikp secure electronic payment system. IEEE Journal on Selected Areas in Communications 18, 611–627 (2000)

6. Bellare, M., Kilian, J., Rogaway, P.: The security of the cipher block chaining message authentication code. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, Springer, Heidelberg (1994)

7. Black, J., Rogaway, P.: Ciphers with arbitrary finite domains. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 114–130. Springer, Heidelberg (2002)

8. Citigroup.: Citi identify theft solutions: Virtual account numbers, http://www.citibank.com/us/cards/cardserv/advice/van.htm

9. Dennis, S.: French banks hacked (March 2000), http://www.computeruser.com/newstoday/00/03/11/news4.html

10. Discover Bank.: Discover card: Secure online account numbers, http://www.discovercard.com/discover/data/faq/soan.shtml

11. Evers, J.: Amazon unit loses credit card data to hackers. InfoWorld (March 6, 2001)

12. Franklin, D.C., Rosen, D.: Electronic online commerce card with transactionproxy number for online transactions. Patent 5883810 (1999)

13. Kim, J., Biryukov, A., Preneel, B., Hong, S.: On the security of HMAC and NMAC based on HAVAL, MD4, MD5, SHA-0 and SHA-1. Cryptology ePrint Archive, Report 2006/187 (2006), http://eprint.iacr.org/

14. Krim, J., Barbaro, M.: 40 Million Credit Card Numbers Hacked. Washington Post, p. A01 (June 18, 2005)

15. MasterCard: Mastercard securecode, http://www.mastercard.com/securecode/

16. Preneel, B., van Oorschot, P.C.: MDx-MAC and building fast MACs from hash functions. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 1–14. Springer, Heidelberg (1995)

17. Rubin, A.D., Wright, R.N.: Off-line generation of limited-use credit card numbers. In: Syverson, P.F. (ed.) FC 2001. LNCS, vol. 2339, pp. 196–209. Springer, Heidelberg (2002)

18. Secure Electronic Transaction LLC: Set secure electronic transaction specification – version 1.0 (1997)

19. Shamir, A.: Secureclick: A web payment system with disposable credit card numbers. In: Syverson, P.F. (ed.) FC 2001. LNCS, vol. 2339, pp. 232–242. Springer, Heidelberg (2002)

20. Singh, A., dos Santos, A.L.M.: Grammar based off line generation of disposable credit card numbers. In: SAC 2002, pp. 221–228. ACM Press, New York (2002)

21. D. Transactions. Discover redoubles its commitment to single-use card numbers, http://www.orbiscom.com/news9.php

22. Visa International Service Association: Visa security progam: Verified by visa, https://usa.visa.com/personal/security/vbv/index.html

23. Visa International Service Association: Rules for visa merchants - card acceptance and chargeback management guidelines. Technical report, Visa International Service Association (2005)

24. Visa International Service Association: Visanet fact sheets (2006), http://www.corporate.visa.com/md/fs/corporate/visanet.jsp

25. Weiss, T.: Laptop with credit card info for 80,000 DOJ workers stolen. ComputerWorld (March 31 2005), http://www.computerworld.com/governmenttopics/government/legalissues/story/0,10801,102146,00.html

26. Ziegler, J.: Everything you ever wanted to know about CC's, http://euro.ecom.cmu.edu/resources/elibrary/everycc.htm

The Unbearable Lightness of PIN Cracking

Omer Berkman[1] and Odelia Moshe Ostrovsky[2,3,*]

[1] The Academic College of Tel Aviv Yaffo, School of Computer Science
[2] Algorithmic Research Ltd.,
www.arx.com
[3] Tel Aviv University, School of Computer Science

Abstract. We describe new attacks on the financial PIN processing API. The attacks apply to switches as well as to verification facilities. The attacks are extremely severe allowing an attacker to expose customer PINs by executing only one or two API calls per exposed PIN. One of the attacks uses only the translate function which is a required function in every switch. The other attacks abuse functions that are used to allow customers to select their PINs online. Some of the attacks can be applied in switches even though the attacked functions require issuer's keys which do not exist in a switch. This is particularly disturbing as it was widely believed that functions requiring issuer's keys cannot do any harm if the respective keys are unavailable.

Keywords: Security API, API attack, Financial PIN Processing API, HSM, Insider attack, Phantom Withdrawal, VISA PVV, IBM 3624, EMV.

1 Introduction

Personal Identification Number (PIN) is the means used by a bank account holder to verify his/her identity to the issuing bank. When a PIN is entered by the card holder at a service point (e.g., an Automatic Teller Machine), the PIN and account number are sent to the verification facility (the issuing bank or other authorized entity) for verification. To protect the PIN on transit, it is formatted into a PIN block, the PIN block is encrypted under a transport key and the resulting Encrypted PIN Block (EPB) is sent for verification. As there usually isn't direct communication between the service point and the verification facility, the PIN goes through switches. Each switch decrypts the EPB, verifies the resulting PIN block format (so the format serves as some form of Message Authentication Code), re-formats the PIN block if necessary, and re-encrypts the PIN block with a transport key shared with the next switch (or the verification facility when arriving there). Switches may be part of other issuers' verification facilities or may be stand alone. There is generally no connection between a switch facility that handles an incoming EPB, and the issuer of the respective

---

account number. Additionally, switches may be physically far from the issuer (for example, when a customer withdraws money overseas).

To protect the PIN and the encryption keys both in switches and in the issuer's environment, all operations involving a clear PIN are handled within a Hardware Security Module (HSM). Such operations are controlled by an application at the site using a cryptographic API. The Financial PIN Processing API is a 30-years old standard which includes functions for, e.g., PIN issuing, PIN verification, PIN reformatting, and PIN change.

The issuer's environment is usually physically separated into an issuing facility and an online verification facility. The issuing facility where customer PINs are generated and printed for delivery is usually isolated logically and physically from the rest of the issuer's environments. The verification facility as well as switches on the other hand, are connected to the outside world and required to be online so they are much more prone to attack. Much of the required functionality in the issuing facility is sensitive, so HSMs implementing the Financial PIN processing API should separate (at least logically) the functionality required for the issuing facility from that of the verification facility. Switches are treated as verification facilities in this respect so HSMs in switches should contain (at least logically) only functions required for the verification facility.

In this paper we describe attacks on the Financial PIN Processing API, which result in discovering customers PINs. The attacks can be applied in switches as well as in verification facilities. The attacks require access (i) to the HSM in the attacked facility for executing API calls; (ii) to EPBs incoming to the attacked facility. Applying such attacks thus requires the help of an insider in the attacked facility. However, when the attacks are applied on a switch, one cannot relate to them as insider attacks. Since the switch, and the issuer whose EPBs are attacked on the switch, are unrelated, an insider of the switch facility is an outsider from the issuer's point of view. The issuer has no control, neither on the environment nor on the employees in the attacked facility. We stress that our attacks only require the use of API functions (and only the ones approved for the verification facility) and **do not** assume that the attacker can perform sensitive operations such as loading known keys into the attacked HSM.

Attack 1 uses a single API function denoted *translate*. The *translate* function allows to reformat an EPB in any PIN block format to an EPB in another PIN block format. It also allows to change the transport key which encrypts the PIN block. It is a required function in every switch, and exists also in verification facilities as part of the API. The attack executes a (one-time) preprocessing step of 20,000 HSM calls in which a (small) look-up table is built. This table allows revealing the PIN packed in each EPB arriving to the attacked switch using one or two HSM calls.

Attack 2 requires the use of one of two API functions - *calculate offset* or *calculate PVV* - which are used primarily for allowing customers to select their PINs online. The attack has four variants. These variants allow discovering a PIN given its respective account number (Attack 2.1), discovering a PIN from its EPB (Attack 2.2), setting a new value for a customer's PIN given the respective

account number (Attack 2.3), and partitioning all EPBs arriving to the attacked facility into groups having the same PIN (Attack 2.4). The attacks on account numbers (2.1 and 2.3) are applied in a verification facility. The attacks on EPBs (2.2 and 2.4) can be applied both in switches and in verification facilities. Each of the four variants can be performed in one or two HSM calls per attacked entity (account number or EPB), and requires no preprocessing.

Both *calculate offset* and *calculate PVV* functions require issuer keys so it is quite surprising that they can be attacked in switches as switches do not contain issuer keys. This is particularly disturbing as it is widely believed that functions requiring issuer's keys cannot do any harm if the respective keys are unavailable.

In some of the cases above, the attacked functions are not used by the application at the site. For example, the *calculate offset* and *calculate PVV* functions are generally not required in switches and *translate* is generally not required in a verification facility. It is important in such cases to irreversibly disable these functions (as well as other unused functions) if this capability is offered. Issuers certainly have the incentive to apply such measures in their verification (and issuing) facilities. However, it is not clear how to verify that switch facilities adhere to these measures.

The attacks abuse integrity and secrecy weaknesses in the financial PIN processing API, some of which are well known ([1,2,3,4,5,6], see also Section 3). For example, integrity in the financial PIN processing API is so weak that one can easily trick API functions into accepting a customer's EPB together with an account number which is not the customer's.

As the attacks target the standard itself, they apply to all common commercial HSMs implementing the API and affect all financial institutions. The attacks apply also to systems employing the EMV standard ([7]) when on-line verification takes place, as is the case in ATM transactions.

The attacks enable the discovery of several thousand customer PINs per attacked HSM per second enabling an attacker to apply serious attacks on issuing banks, such as simultaneous withdrawals of aggregate large sums of money. The attacks may also explain cases of phantom withdrawals where a cash withdrawal from an ATM has occurred, and neither the customer nor the bank admits liability.

To prevent the attacks described in this paper, changes in the standard must be introduced. Such changes require worldwide modifications in ATMs, HSMs and other components implementing the Financial PIN processing API.

The rest of the paper is organized as follows. Section 2 discusses the threat model. In Section 3 we describe known vulnerabilities in the standard. Sections 4 and 5 describe our attacks. A discussion of the attacks is given in Section 6 and is followed by concluding remarks in Section 7. Finally, an appendix contains information on the attacked functions for reference.

2   Threat Model

A potential attacker is an insider of the attacked facility - a switch or a verification facility. Such an insider should have logical access to the HSM in the

facility and should be able to generate API calls (the required API functions depend on the attack). In many cases this is easy as the HSM is connected to the organization's internal network. When this is not the case, the attacker can, for example, interfere with or masquerade as the legal application working with the HSM in the attacked facility.

In most of the attacks the attacker is required to generate EPBs which contain known PINs and which share a transport key with the attacked HSM. To do this, the attacker can use any banking card (genuine or fake) and enter a desired PIN at an ATM adjacent physically or logically to the attacked HSM. The attacker then needs to record the EPB when it arrives to the attacked facility. This can be done in various ways, e.g., by a program that reads the EPB on its way from the application to the HSM in the site. In the same way an attacker is able to record EPBs incoming to the switch, e.g., in order to expose the PINs they hide.

In order to prevent insider attacks on their HSMs, a few banks install a (hardware) mechanism by which their on-line application timeouts whenever the HSM is used by a different entity. However, it is hard to figure out whether a short timeout really signifies an attack. Furthermore, it is possible to attack a system employing such a mechanism by, for example, physically intervening with the low-level communication between the application and the HSM.

All API functions use cryptographic keys. The standard does not specify how keys should be input to an API function but most implementations either keep keys outside the HSM encrypted by a master key, or keep them inside the HSM. In the first case, HSMs accept encrypted keys in each API call. In this case, an attacker is only required to record the desired encrypted key buffer from a real transaction. The same encrypted key can then be used in the attacker's API calls to the HSM. In the second case where keys are stored and managed inside the HSM, the attacker only needs to know the required key ID. In this case, however, the HSM may also handle user access rights to the keys. To use the required keys in such cases the attacker can, as before, interfere with or masquerade as the legal application working with the HSM in the attacked facility. In any case, we never assume that the attacker has any knowledge of the value of cryptographic keys.

Transport keys sometimes change. However, parameters to the API functions that control the keys to be used in the API function come from the outside so the attacker can always direct the HSM to use the same key. Additionally, when required, the attacker can use the *translate* function to translate an EPB encrypted with one transport key to an EPB encrypted with another.

It should be noted that an attacker is not required to be an authorized user - a maintenance employee or after-hours cleaner can generally do the job on the attacker's behalf. Moreover, in all variants of Attack 2, the attacker can also be an insider programmer that applies the attack innocently, believing that the PVVs or offsets (see Section 5 and the appendix for definitions) he/she was asked to supply are required for legitimate purposes as they are output in clear from the relevant API functions, and treated as non-sensitive. In switches, insiders (perhaps even high-ranking) may conduct the attacks but target EPBs of foreign banks only, dramatically decreasing the chances of their being caught.

3   Basics and Previous Work

As we mentioned in the introduction, on its way for verification, the PIN is formatted into a PIN block and the result is encrypted using a transport key to generate an Encrypted PIN Block (EPB). Specifically, [8] describes four different PIN Block formats. ISO-0, ISO-1, ISO-2, and ISO-3, which differ in whether the customer's account number and/or random data is involved in the format in addition to the PIN itself. ISO-0 uses only account number, ISO-1 uses only random data, ISO-2 uses neither account number nor random data, and ISO-3 uses both account number and random data. [9] approves ISO-0, ISO-1, and ISO-3 for online PIN transactions. ISO-2 is **not** approved for online PIN transactions since an EPB based on ISO-2 (and on a given transport key) has only 10,000 possible values (assuming the PIN is of length 4) enabling the use of a look-up table.

Our attacks abuse the following known weaknesses:

1. The *translate* API function allows reformatting an EPB from any of the approved formats (ISO-0, ISO-1, or ISO-3) to another ([3,6,4]).
2. The ISO-1 format is independent of any account number ([6]).
3. A result of Weaknesses 1 and 2 is that an EPB in ISO-0 (or ISO-3) associated with a given account number can be converted (by going through ISO-1) to an EPB in ISO-0 associated with a different account number ([1], [3] and [6]). Note that doing this unties the link between the customer's account number and the customer's PIN and creates a fabricated link between a different account number and this customer's PIN.
4. An EPB based on ISO-0 and a particular account number has only 10,000 possible values enabling the use of a look-up table ([3,1,6]). We note that this weakness implies that for a particular account number, the ISO-0 format is as weak as ISO-2.
5. As mentioned in the introduction, the format of PIN block serves as a form of Message Authentication Code (MAC). The weakness is that in ISO-0 format, digits of the PIN are XORed with digits of the account number, making it impossible to correctly authenticate neither ([6,4,3]).

We are not aware of previous attacks abusing the *calculate offset* and *calculate PVV* functions but note that Attack 2.1 which abuses the *calculate offset* function is reminiscent of an attack (described in [10,11,6]) on a **non-API** function which was added temporarily to the implementation of the API in a certain bank in order to enable changing all customers' account numbers without re-issuing new PINs.

Previous API-level attacks appear in [12,13,14]. Previous attacks on the Financial PIN Processing Standard appear in the references above as well as in [11]. Among these, of particular interest is the decimalisation-table attack ([5]), which targets the *verify* function in verification facilities. It allows revealing a PIN from its account number or EPB by executing 15 HSM calls on the average. Contrary to our attacks, the decimalisation-table attack does not apply to switches. Additionally, it targets only one of the two verification methods in the standard (see Section 5 for details) while we attack both.

4   Attack 1 - Attacking the Translate Function

The attack we describe in this section, enables revealing for any EPB arriving to the attacked switch (or verification facility), the PIN that the EPB packs. The attack uses at most two API calls per EPB. The attack requires also a one-time preprocessing step consisting of 20,000 API calls (assuming the PIN is of length 4 as is normally the case). The attack uses the translate API function only.

We start by observing that Weakness 3 (in Section 3) degrades the security of the system to the strength of ISO-2 (recall that ISO-2 is the weak and thus non-approved PIN block format): Fixing an account number to some value A and translating all EPBs arriving to the attacked switch to ISO-0 with account number A (going through ISO-1) ensures that all resulting EPBs are based on account number A, thus degrading their strength to that of ISO-2. This observation has extremely serious implications on the security of the Financial PIN Processing API, as it implies that a single look-up table of size 10,000 is all that is required in order to discover the PIN packed in **every EPB arriving to the attacked switch**, regardless of its account number. Clulow ([6]) was evidently aware to this saying "it is noteworthy that regardless of format, key and pan, all encrypted pins are potentially vulnerable to a single codebook", but his words seem to have gone almost unnoticed probably because he and others had no efficient way of building the required table.

In this section we do exactly this. Specifically, we show how to generate a table of 10,000 EPBs, where the ith EPB, $1 \leq i \leq 10,000$ contains the PIN whose value is i, and such that each EPB in the table is formatted in ISO-0 using a fixed account number A.

One obvious way such a table can be generated is by brute force - generating 10,000 EPBs by ATMs: For each i, $1 \leq i \leq 10,000$ use a card with any account number and type PIN value i. When the respective EPB arrives at the attacked HSM, translate it to ISO-0 using account number A (by one or two calls to the *translate* function depending on the format of the incoming EPB). Using different account numbers when generating EPBs via ATMs would make it harder to discover the attack.

We now describe a much more practical method of building the table. Instead of generating an EPB per each possible PIN as in the brute force manner above, this method uses Weakness 5 to generate an EPB per 100 possible PINs. Thus, by generating 100 EPBs we can build the whole 10,000-entries look-up table (it is also possible to generate less than 100 EPBs and build a partial look-up table).

We start by describing the ISO-0 PIN block format. Denote the PIN $P_1 P_2 P_3 P_4$ and the respective account number $A_1 A_2 \ldots A_{12}$ (only 12 digits of the account number are used in the ISO formats). The PIN block is the XOR of two 16-hexadecimal digits blocks. An *original block* containing the PIN ("F" stands for the hexadecimal value F)

0	4	P_1	P_2	P_3	P_4	F	F	F	F	F	F	F	F	F	F

with an *account number block* containing the account number

0	0	0	0	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}

When an API function receives an EPB in ISO-0 as a parameter, it also receives its associated account number. To use the PIN packed in the EPB, the function decrypts the EPB, and XORs the result with the account number block to recreate the original block. It then authenticates the result by verifying that the values of the first two digits of the original block are 0 and 4, that the last 10 digits are hexadecimal F and that the PIN is composed of decimal digits.

Weakness 5 - the fact that two digits of the PIN are XORed with two digits of the account number - is used for generating the table. The attacker generates in ATMs 100 EPBs packing, respectively, PIN values $0000, 0100, \ldots, 9900$. Each of these EPBs is formatted in ISO-0 and associated with account number $00A_3 \ldots A_{12}$ where the values $A_3, \ldots, A_{12}$ are immaterial to the attack and can be different for each PIN value to make the attack more innocent. (It is also possible to generate the 100 EPBs in ATMs using completely arbitrary account numbers and then change the account number of each EPB to the desired one using the *translate* function.)

We complete our description by showing how the attacker generates an EPB containing PIN value $xyuv$ for any decimal values x, y, u, v:

To generate an EPB that packs PIN value $xyuv$, the attacker uses the EPB packing $xy00$ which was generated by ATM. Using the *translate* function this EPB is reformatted to ISO-1 but instead of using the original account number $00A_3 \ldots A_{12}$ the attacker provides the *translate* function with account number $uvA_3 \ldots A_{12}$.

The *translate* function decrypts the EPB and gets a block which is the XOR of the original block

0	4	x	y	0	0	F	F	F	F	F	F	F	F	F	F

and the original account number block

0	0	0	0	0	0	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}

Note that the function only sees the decrypted block - the XOR of these two blocks. It then XORs the decrypted block with the following:

0	0	0	0	u	v	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}

to get

0	4	x	y	u	v	F	F	F	F	F	F	F	F	F	F

This resulting block will be authenticated (its first two digits are 0 and 4, its 10 last digits are hexadecimal F, and the PIN consists of decimal digits). Consequently, PIN value $xyuv$ will be packed in an EPB in ISO-1 PIN block and returned. The attacker can now translate this EPB to ISO-0 with the desired account number A.

5   Attack 2 - Attacking Functions Allowing PIN Change

In the Financial PIN Processing API, the PIN is verified using one of two approved methods - the IBM 3624 or the VISA PIN verification value (PVV) methods. In both methods the input to the verify function is as follows:

- An EPB containing the PIN presented by the customer.
- The customer's account number.
- A four decimal digits *customer's verification value* (called *offset* in the first method and *PVV* in the second).

This customer's verification value is not secret. It is kept either in a database or on the customer's card.

Denote by P the PIN packed in the EPB, by A the customer's account number, and by V the customer's verification value.

The verify function decrypts the EPB, authenticates it by verifying the PIN block format, extracts P from the EPB, and verifies whether $V = f(P, A)$ where f is a function which depends on an issuer's secret key. The function f and the issuer's key are different between the two methods.

In order to allow customers to select their PINs online, the Financial PIN Processing API contains two functions (one for each method) that allow recalculating the customer's verification value when the customer's PIN changes. The functions are denoted *calculate offset* and *calculate PVV*. Both functions receive the following input:

- An EPB containing the customer's selected PIN.
- The customer's account number.

The functions return $V = f(P, A)$ where P, A, V and f are as before. We note that in both functions, the value V is pseudo random as a result of using the random issuer's key in f.

The main weakness in both functions regardless of f is that the new PIN supplied to the function (packed in an EPB) is not bound to the old PIN. Indeed, the main step in each of the four variants 2.1-2.4 abuses this weakness, so the variants have much in common.

Note that since an attacker can carry out the attack by directly using the API, it would not be enough to check the above binding by the application at the site. Note also that it would not be enough to change the API by adding each of these functions a parameter consisting of an EPB that packs the customer's old PIN (and a means to verify it, i.e., the respective verification value) since the attacker can record a customer's real EPB on its way for verification, and use it as the additional parameter.

To attack any of the two API functions, we are required to send as parameters an EPB and an account number. In all four attacks, the EPB would be generated by one customer and the account number would belong to another. To force the attacked function to accept the non-matching parameters, weaknesses 1 and 2 of Section 3 are utilized: We use the *translate* function to reformat each EPB to

ISO-1 before sending it to the respective API function. Because ISO-1 does not depend on account number, there would be no inconsistency between the EPB parameter and the account number parameter. When describing the attacks below, we do not mention this reformatting any more (but we count it in the number of HSM calls required). Note that restricting the *calculate offset* or *calculate PVV* functions to accept only EPBs with a certain format would not thwart the attacks, as we can reformat the EPB to that format. Note also that with the exception of Attack 2.4, all the attacks below can be applied (though in a more restricted form) even if the *translate* function is disabled. See Section 5.3.

In Section 5.1 we describe our attacks on the *calculate offset* function. In Section 5.2 we describe our attacks on the *calculate PVV* function. It is worth noting that except for assuming that the value V is pseudo random, the attacks on *calculate PVV* do not use any properties of the respective f (so, for example, they apply also to *calculate offset*).

We use a shorthand $O = offset(E, A)$ (respectively, $V = PVV(E, A)$) to denote calling the *calculate offset* function (respectively, the *calculate PVV* function) with an EPB E and an account number A.

5.1   Attacks on the Calculate Offset Function

The specific function f in *calculate offset* is $V = P - g(A)$ where P is the PIN packed in the EPB parameter, A is the account number parameter, V is the returned offset, g is a function that depends on an issuer's key and computes a 4 decimal digits number, and "$-$" is minus modulo 10 digit by digit.

Attack 2.1 - Attacking Account Numbers in a Verification Facility. This attack reveals for every customer account number associated with the attacked issuer, the respective customer's PIN. It requires one HSM call per attacked account number. In addition, it requires generating by ATM an EPB that packs a known PIN. This single EPB will be used to attack all account numbers associated with the issuer.

We start by generating an EPB in an ATM that packs an arbitrary known PIN (the account number associated with this EPB is immaterial as we reformat the EPB to ISO-1 prior to using it). This EPB, denoted E_a (for attacker's EPB) is used to attack the account numbers of all customers.

For each customer's account number A_c, compute $O = Offset(E_a, A_c)$. Denote by P_a the value of the known PIN packed in the attacker's EPB, by P_c the required customer PIN, and by O_c the customer's offset (stored in the issuer's database or on the magnetic stripe of the card). We thus have $O = P_a - g(A_c)$. Since P_a is known, $g(A_c)$ can be easily computed. We also know that $O_c = P_c - g(A_c)$. Since $g(A_c)$ is known and since the value of O_c is not secret (it can be recorded during a transaction or read from the database or from the card) the customer's PIN P_c can be trivially calculated. Note that the exact nature of g is immaterial, but the attack requires that the real value of $g(A_c)$ be used (since the value of O_c depends on it) so it needs to be applied in the verification facility where the required issuer's key exists.

Attack 2.2 - Attacking EPBs Incoming to a Switch. The attack reveals for each customer's EPB arriving to the attacked switch, the PIN it packs. It requires one or two HSM calls per attacked EPB. In addition, it requires generating by ATM an EPB that packs a known PIN. This single EPB will be used to attack all EPBs arriving to the attacked switch. We note that the attack can be applied also in verification facilities.

Generate an EPB in an ATM that packs a known PIN and denote it E_a. Fix an arbitrary account number B. Compute $O_1 = Offset(E_a, B)$. For each customer's EPB arriving to the attacked switch compute $O_2 = Offset(E_c, B)$ where E_c is the customer's EPB.

Denote by P_a and P_c the values of PINs packed in the attacker's and customer's EPBs, respectively. We thus have $O_1 = P_a - g(B)$ and $O_2 = P_c - g(B)$. Since the value of P_a is known, the value of P_c can be trivially calculated. Note that the value of $g(B)$ is immaterial, so the attack can be applied in a switch which does not contain the required issuer's key.

5.2 Attacks on the Calculate PVV Function

Attack 2.3 - Attacking Account Numbers in a Verification Facility. This attack reveals for any account number associated with the attacked issuer, the PVV that corresponds to this customer's account number and an attacker's chosen PIN. Replacing the verification value on the card or in the database (depending on the system) enables withdrawing money from the customer's account using the chosen PIN. The attack requires one HSM call per account number attacked. In addition, it requires generating by ATM an EPB that packs a known PIN. This single EPB will be used to attack the account numbers of all customers.

Generate an EPB in an ATM that packs an arbitrary known PIN and denote it E_a. For each customer's account number A_c, compute $V = PVV(E_a, A_c)$.

The computed PVV value V corresponds to the customer's account number and the chosen PIN. Since the attack takes place in the verification facility, the required issuer's key is used, and the PVV is valid.

It remains to explain how the attacker can replace the customer's original PVV used by the system by the PVV computed in the attack.

According to [9], the clear PVV can be stored on the card's magnetic stripe or in a PVV database. In case the PVV is stored on both, the PVV is taken from the database. In many implementations the PVV is stored only on the card as long as the customer uses the initial PIN generated by the issuer.

Setting the customer's PVV to the computed PVV can be done as follows:

Case 1: The PVV is stored only on the card. Generate a card containing the customer's details and set the PVV value on the magnetic stripe to the PVV that was calculated by the attacker. In this case the fabricated card (associated with the attacker's chosen PIN) and the customer's original card (associated with the customer's PIN) will both be valid at the same time. It is important to note that in this case, issuing a new PIN to a customer will not prevent the attack as the fabricated card with the false PVV will remain valid.

Case 2: The PVV entry of this customer exists in the PVV database. In this case the attacker needs write access to the PVV database. As both PVVs and offsets are not considered sensitive, access to this database is generally not restricted. In many banks, for example, HSM service personnel can access this database. As a result of the attacks published in this paper, the attitude towards PVVs and offsets is now being changed. Anyway, given write access, the attacker can do one of the following:

- Delete the PVV entry (and then apply the steps described in Case 1).
- Set the customer's entry in the PVV database to the PVV that was calculated by the attacker. If the entry does not exist - create it. In this case the fabricated card will be the only valid card.

Attack 2.4 - Attacking EPBs Incoming to a Switch. Consider all customer EPBs arriving to the attacked switch. The attack discovers for each such EPB (and its associated account number) a list of other EPBs having the same PIN (with high probability). It requires one or two HSM calls per attacked EPB. We note that the attack can be applied also in verification facilities.

We use a table of 10,000 entries. The table is indexed by values of computed *PVV*s. Each entry of the table contains customer EPBs (and their associated account numbers). Initially all entries are empty.

Fix an arbitrary account number B. We show how to attack any customer's EPB arriving to the switch. Denote by E_c the customer's EPB.

1. $V = PVV(E_c, B)$.
   The computed PVV value V equals $f(P_c, B)$ where P_c is the PIN packed in the customer's EPB.
2. Add the customer's EPB E_c to the table entry corresponding to the resulting PVV value V.
   For example, if V is 5678 then E_c will be added to table entry 5678.

The computed PVV value V depends only on P_c, B, and on the key k used by the function f. Since the attack is applied in a switch, k is not the issuer's key as required, but some other arbitrary value (not known to the attacker). The value of k is immaterial to the attack. All that we require is that the value V be a pseudo random function of P_c, B, and k. Since B and k are fixed, V can be regarded as a pseudo random function of P_c only.

Suppose we have performed the above with many EPBs. What actually happens in steps 1 and 2 above is that all EPBs that pack the same PIN value are thrown into the same table entry. Since the process is random, a table entry may be empty, may contain EPBs corresponding to a single value of PIN, or may contain EPBs corresponding to several PIN values. Combinatorically, the process is equivalent to throwing balls (PINs) to bins (table entries) and asking questions on the number of balls (distinct PINs) in each bin. It can be shown that when the number of balls and bins is the same (10,000 in our case) the average number of balls in a non-empty bin is less than 2. In other words, EPBs

that ended in the same table entry correspond to less than 2 distinct PINs on the average.

To decrease the probability that EPBs in the same table entry correspond to more than a single PIN, we repeat the procedure with respect to the EPBs in each table entry using a different fixed account number C. EPBs from a given table entry that again end together in the same table entry, have high probability of having the same PIN.

5.3   What If Reformatting Is Disabled

In each of the attacks 2.1, 2.2, and 2.3 above, the attacker uses the translate function to reformat EPBs to ISO-1. This enables using a single recorded EPB for attacking all account numbers in attacks 2.1 and 2.3 (and all EPBs in Attack 2.2). Could the attacks still work if the *translate* function (or its reformatting capability) is disabled and all EPBs are in either ISO-0 or ISO-3 format?

An easy solution is for the attacker to record for each account number attacked, an EPB associated with that account number. Although such an attack cannot be applied on very large scale, it can still be harmful (especially when the attacker is interested in attacking specific customers).

Moreover, using the overlapping between PIN digits and account number digits (Weakness 5 in Section 3), a single recorded EPB in ISO-0 format may be used to attack up to 100 account numbers. Using Weakness 5 together with overlapping between account number digits and random digits, an EPB in ISO-3 format may be used to attack up to $6 \cdot 10^9$ account numbers. The details appear in [15].

6   Discussion

Attack 1 is perhaps the most hard to handle as the *translate* function is a required function in every switch. Attacks 2.1 and 2.3 deserve special attention as they do not even require that the customer know his/her PIN. Moreover, if one is interested in attacking a specific account number, the translate function is not required for the attack (as discussed in Section 5.3). Attack 2.2 is surprising, as it implies that customer fake cards having PINs different from the customer's original PIN, can be valid together with the customer's genuine card. Attack 2.4 does not require generating an EPB in an ATM. On the negative side, Attack 1 requires generating in ATMs 100 EPBs, and all variants of Attack 2 require that *calculate PVV* or *Calculate offset* be available.

Our recommendation to issuers is as follows. In their facilities, issuers should disable the *calculate offset* and *Calculate PVV* functions as well as the reformatting capability of translate, even for the price of eliminating customer selected PINs or other capabilities. Warning mechanism (e.g., with respect to the number of times API functions are called) may also be useful. With respect to switches, issuers should ensure good control over their country's local switches and apply detection and other mechanisms with respect to overseas transactions.

7   Conclusions

We have shown in this paper that the Financial PIN processing API is exposed to severe attacks on the functions *translate*, *calculate PVV* and *calculate offset* inside and outside of the issuer environment.

The attacks we describe provide possible explanations to many Phantom Withdrawals. The attacks are so simple and practical that issuers may have to admit liability not only for future cases but even retroactively. The attacks can be applied on such a large scale (in some of the attacks up to 18,000,000 PINs can be discovered in an hour) that banks' liability can be enormous.

As some of the attacks apply to switches, which are not under the issuers control, countermeasures in the issuers environment do not suffice. To be protected from this attack, countermeasures in all verification paths to the issuer must be taken. As this is unrealistic, solutions outside the standard must be sought.

We have also shown that physical and/or logical separation of the issuing and verification facilities does not prevent severe attacks, as part of the API functionality intended for use in verification facilities is vulnerable.

We have demonstrated that reformatting capability between different PIN block formats, can go further than degrading the security of the system to the weakest format, as weaknesses of several formats may be abused. Our attacks also show that the ISO-1 format is extremely weak and thus should be immediately removed from the list of approved interchange transaction formats.

Another interesting insight from the attacks described is that the offset and the PVV values may reveal as much information as the PIN itself. One possible remedy is treating them as secret values.

In addition to all implementations of this API, systems applying the EMV standard ([7]) and using online (rather than off-line) PIN verification are also vulnerable to the attacks.

The vulnerabilities exposed in this paper require worldwide modifications in ATMs, HSMs and other components implementing the PIN processing API.

References

1. Anderson, R.J., Bond, M., Clulow, J., Skorobogatov, S.: Cryptographic processors - a survey. Proceedings of the IEEE 94(2), 357–369 (2006)
2. Bond, M.: Understanding Security APIs. PhD thesis, University of Cambridge (2004), http://www.cl.cam.ac.uk/mkb23/research.html
3. Bond, M., Clulow, J.: Encrypted? randomised? compromised? In: Workshop on Cryptographic Algorithms and their Uses (2004)

4. Bond, M., Clulow, J.: Extending security protocols analysis: New challenges. In: Automated Reasoning and Security Protocols Analysis (ARSPA), pp. 602–608 (2004)
5. Bond, M., Zielinski, P.: Decimalization table attacks for pin cracking. Technical Report UCAM-CL-TR-560, University of Cambridge, computer Laboratory (2003), http://www.cl.cam.ac.uk/TechReports/UCAM-CL-TR-560.pdf
6. Clulow, J.: The design and analysis of cryptographic APIs. Master's thesis, University of Natal, South Africa (2003), http://www.cl.cam.ac.uk/jc407
7. EMV: Integrated circuit card specifications for payment systems (2004), http://www.emvco.com
8. ISO: Banking – personal identification number (PIN) management and security – part 1: Basic principles and requirements for online PIN handling in ATM and POS systems (2002)
9. VISA: PIN security requirements (2004), http://partnernetwork.visa.com/st/pin/pdfs/PCI_PIN_Security_Requirements.pdf
10. Anderson, R.: The correctness of crypto transaction sets. In: Christianson, B., Crispo, B., Malcolm, J.A., Roe, M. (eds.) Security Protocols. LNCS, vol. 2133, pp. 128–141. Springer, Heidelberg (2001)
11. Andersson, R.J.: Why cryptosystems fail. Communications of the ACM 37(11), 32–40 (1994)
12. Longley, D.: Expert systems applied to the analysis of key management schemes. Computers and Security 6(1), 54–67 (1987)
13. Rigby, S.: Key management in secure data networks. Master's thesis, Queensland Institute of Technology, Australia (1987)
14. Steel, G., Bundy, G.: Deduction with XOR constraints in security API modelling. In: McAllester, D. (ed.) CADE-17. LNCS, vol. 1831, Springer, Heidelberg (2000)
15. Moshe-Ostrovsky, O.: Vulnerabilities in the financial PIN processing API. Master's thesis, Tel Aviv University (2006)

Appendix

We describe below, for reference, the attacked functions parameters, and the computation performed by each function.

Translate	Calculate PVV	Calculate offset
EPB	EPB	EPB
account number	account number	account number
input PIN block format	PIN block format	PIN block format
input transport key	transport key	transport key
output PIN block format	issuer's PVV key	issuer's PIN key
output transport key		

The *translate* function extracts the PIN from the EPB by decrypting the EPB using the input transport key and authenticating the result using the account number and input PIN block format (Section 4 describes the authentication process). It then re-formats the PIN into a PIN block using the output PIN block format and account number, and re-encrypts the result using the output transport key. The resulting EPB is the output of the function.

The *calculate PVV* function extracts the PIN from the EPB (as in *translate*). It then concatenates the PIN to the account number, encrypts the result using the issuer's PVV key, and extracts four decimal digits from the encrypted result. These four digits constitute a PIN Verification Value (PVV) which is the output of the function.

The *calculate offset* function extracts the PIN from the EPB. It then encrypts the account number using the issuer's PIN key and extracts four decimal digits denoted *natural PIN* from the encrypted result. The natural PIN is then subtracted (modulus 10) from the PIN. The result constitute an *offset* which is the output of the function.

Virtual Economies: Threats and Risks

Christopher Thorpe[1], Jessica Hammer[2], Jean Camp[3], Jon Callas[4], and Mike Bond[5]

[1] Harvard University
cat@seas.harvard.edu
[2] Columbia University
jh2354@columbia.edu
[3] Indiana University
ljcamp@indiana.edu
[4] PGP Corporation
jon@pgp.com
[5] Cryptomathic Ltd
Mike.Bond@cl.cam.ac.uk

These notes were prepared to explore ideas developed in a panel discussion at Financial Cryptography 2007. Moderator: Jean Camp. Panelists: Mike Bond, Jon Callas, Christopher Thorpe.

1 Introduction

In virtual economies, human and computer players produce goods and services, hold assets, and trade them with other in-game entities, in the same way that people and corporations participate in "real-world" economies. As the border between virtual worlds and the real world grows more and more permeable, privacy and security in virtual worlds matter more and more.

Virtual economies first appeared as early as the late 1970's in MUDs (Multi-User Dungeons), with the advent of dial-up bulletin board systems and research computer internetworking. The earliest and simplest in-game economies simply allowed players to obtain currency dropped by slain monsters or from in-game vendors who would purchase unwanted items (usually also dropped by slain monsters). This currency could be used to buy superior weapons, armor, or training to allow the player to more effectively kill (often more powerful) monsters and thus earn more money. MUDs and related games in the 1980's began to use the in-game currencies for other purposes, such as creating in-game assets. As the complexity of MUDs grew, so did their economies, but because most MUDs were small in scope and run without profit by enthusiasts, there was little implied value in their in-game currency: the players who ran the system would simply conjure up currency whenever they or their friends needed it.

This changed substantially with commercial development of large-scale, multi-user virtual worlds designed to earn a profit. In these worlds, game designers employed scarcity to establish value; in particular, the universal scarcity of time.

Thus, virtual economies began to develop organically, with killing monsters as what might correspond to their first "natural resource" in a real-world economy. The first major success of these, EverQuest, was released in 1999, but as early as 1996, players began to exchange in-game currency and goods for real-world cash [7]. They wanted

more powerful in-game characters, and were willing to spend real-world cash to effectively hire someone to do the virtual work for them. Suddenly, virtual assets had real-world value.

The past few years have seen explosive growth in participation in virtual worlds. World of Warcraft alone has over eight million subscribers worldwide [4]. Many of these "traditional" online games, including World of Warcraft, prohibit the resale of in-game assets, and Blizzard Entertainment has banned tens of thousands of accounts and removed over $1M worth of gold from its World of Warcraft economy from players who exchange gold or use third-party programs to "farm" in-game assets. Trade in these assets is fast becoming a billion-dollar industry [3].

However, some companies have recognized that the exchange of in-game assets for real-world cash is inevitable, and even profitable. Linden Labs, the creators of *Second Life*, facilitate an exchange between its currency, Linden Dollars, and US Dollars. Three Rings Design's *Puzzle Pirates* supports worlds in which players purchase "doubloons" from Three Rings, then trade them for goods and services, or in-game currency from other players, in official, market-driven in-game exchanges. MindArk's *Entropia Universe* pegs its currency, the Project Entropia Dollar (PED), at 10 PED to $1 USD.

Nonetheless, few game players actually "own" their virtual property; end-user license agreements generally make it clear that all in-game state is the property of the game developer. (*Second Life* is a notable exception in this area.) As player investment in virtual worlds continues to grow, "virtual property rights" and the security of virtual property will become important issues.

Still, most players' participation in these in-game economies is ultimately a choice, more akin to playing the stock market than to buying groceries. While players exchange their time or money for in-game goods and services, they can just as easily invest their time and money in activities of other sorts, effectively going "off the grid" in a way rarely possible in real life. Unlike participation in the economy of the real world, the choice to participate in a virtual world is entirely voluntary.

The risks and rewards of a virtual economic life often exist entirely within the "magic circle" of the game itself [10]. Even the most powerful potion of healing in a virtual world cannot heal a real-life patient! Threats to a player's in-game property or achievements are certainly serious, as these items represent an investment of time – the only truly scarce commodity and economic measure of real value in a world of infinite digital duplication. But the magic circle has leaks in it, leaks that let the risks of a virtual world penetrate the very real. Thanks to the rise of economic institutions that exchange real-world cash for the time investment necessary for play – such as eBay, Second Life's LindeX, and the official Everquest II Station Exchange – virtual assets are now very real.

Because of this real value, many security and privacy concerns have emerged that the creators of these worlds could not have anticipated. Even real-world crimes have taken place as a result of, or perhaps, as a means of perpetrating, virtual crimes. While many security threats and risks that exist in the real world do not exist in virtual economies, some of them have emerged in virtual worlds. Moreover, a new class of threats has emerged at the often blurry boundary between virtual and real economies, particulary with respect to privacy. These notes explore these classes of threats and risks.

2   Some Security Issues Don't Exist in Virtual Economies

Many important issues of security in our lives are artifacts of our real-world infrastructure. For example, because of the way money has developed, we constantly deal with security issues surrounding the possession and transfer of cash or its equivalent. In virtual economies, the creators engineer worlds in which certain problems simply can't happen — in fact, in many cases, special coding would have to be created to allow some of our nastiest problems to even exist in virtual worlds. Thus, the absence of a real-world security concern may be due either to a deliberate design choice to eliminate it, or to the simpler nature of a virtual financial existence. We illustrate this with examples.

In most virtual economies, misrepresentation of a good or service is impossible. The buyer can immediately see for herself whether the item magic spell, etc., is what it is claimed to be and reject a forgery. Fake items and goods simply can't exist unless they are specially encoded.

Players' personal assets are protected from other players in modern virtual economies. This means that a threat of in-game violence to obtain in-game benefit is meaningless. In older games, the ability of players to destroy or steal others' property or kill their characters was nearly universally accepted as a problem [14]. Solving this problem has become one of the fundamental design assumptions of today's virtual worlds.

3   Many Real-World Problems Happen Virtually, Too

The majority of real-world problems that happen in virtual worlds stem from deceptive communications. Probably the most famous is one group's infiltration of a powerful "corporation", Ubiqua Seraph, in the game *EVE Online* [2]. Over the course of a year, double agents infiltrated every level of Ubiqua Seraph, and in April 2005, murdered its CEO and took over many of its in-game assets, valued at over $16,000 USD at the time. This was completely within the rules of the game. No police investigations, lengthy trials, or prison sentences ensued; public outcries on game forums for developer redress were rebuffed. This is what distinguishes *EVE* from the real world—it's like the Wild West, and the developers want it that way.

As economies grow in complexity, so do possible exploits of them. Market manipulation is rampant; even players with modest resources can easily corner the market on certain important in-game goods, then sell the goods at a significant profit. Some players have even developed automated programs to exploit in-game markets. We know of no virtual economies that have developed anti-trust rules or price controls in their marketplaces.

Extortion occurs in various interesting new forms in virtual economies. Rather than hostage-taking or threats of violence, one *World of Warcraft* guild allegedly decided to hold in-game content hostage from the rest of the server. They were the only group on the server capable of opening a new dungeon, and posted a demand for 5,000 gold pieces from other guilds (worth approx. $300 USD at the time) before opening the gates of Ahn'Qiraj [9].

4   Virtual Economies Have Created Unique Risks

A more typical example of fraud is that a player promises to pay for a particular service, then either the buyer refuses to pay afterward, or pays first and then the seller refuses to perform the service to the buyer's satisfaction. Another example is to use a deceptive price for an item, such as charging 20 gold pieces for an item that should cost 20 silver pieces and hoping the buyer won't notice.

While this sort of fraud occurs in the real world as well, few virtual economies support written contracts or binding agreements. A player's only recourse after being defrauded is to hope the gamemasters will review server trade and chat logs and resolve the issue by "divine intervention." Some games provide such recourse; for example, in *Puzzle Pirates*, oceanmasters spend much of their time solving such problems, and because perpetrators do not benefit, such problems are rare. Conversely, in *World of Warcraft*, gamemasters generally do not have the authority to provide restitution, and the only recourse is to cry foul in public chat channels. In such games, because players who do it can get away with it, such petty thefts seem to be more common.

Bugs in the programs specifying virtual worlds create new opportunities for dishonest gamers. Sometimes, these are exploited by client players, but other times, it has been alleged that insider game developers make extra cash by selling in-game valuables. Players who discover exploits create or duplicate valuable items or in-game currency, then sell them to other players either in the game or through real-world channels to make money. Some exploits include careful timing attacks, where the player picks up a valuable item more than once or trades cash with another player, but the server doesn't properly log it. Other means have exploited bugs that take place at borders of "zones" in the game world where a player moves from one server to another. One extreme example of such an exploit resulted in a temporary 20% inflation in *EverQuest II* currency prices in August 2005 [13].

Other interesting risks that do not have real-world counterparts are still emerging. Virtual worlds at the moment seem to mimic the way people interact in real worlds, but as technology and familiarity with virtual environments improves, we may discover that there are goods and services in virtual worlds that have no real analogy in our own world, and which we do not know how to manage or regulate. This becomes especially concerning in the context of a virtual economy that "leaks" into the real world via the scarcity of time, and the consequent exchange of in-game and real-world assets. We do not yet understand the relationship of capital in establishing player-initiated goods and services; indeed, since everything is virtual once the world has been programmed, human labor — mouse and keyboard inputs — seem to be the only real input into the system.

5   The Imperfect Border: Risks Where Real-Life Meets the Virtual

In 2005, a Chinese man was stabbed to death after selling a powerful sword an acquaintance had lent him in the online game *Legends of Mir 3*. The attacker had first reported the "theft" to police, who claimed there was nothing they could do, and then took matters into his own real-life hands [1].

Gamers who do not protect their real-life identity information, including their IP address and gaming account information, have found that their in-game actions follow them home through harassing telephone calls, email, or even physical mail or personal visits.

Even when the physical world is not involved, some researchers have argued that harassment within a virtual world itself can be almost as emotionally traumatic as real-life abuse, and many claim that "virtual rape" now exists [11]. Others go further, and argue that crimes in virtual worlds are as real as crimes in our own world [5].

One can also cross the border in the other direction. For example, virtual economies can provide interesting new benefits to those of us in the real world. Brown and Thomas write that a World of Warcraft guildmaster's avocation gave him an edge in landing a senior management position at Yahoo! [6]. Some players make respectable amounts of money playing online games: Mike Everest, a high school student in Colorado, and his mother, earned over $35,000 USD in *Entropia Universe*, some of which was spent sending two siblings to college [12].

But there are risks in a permeable border between the virtual and real. Microsoft warned in a presentation at Gamesfest 2006, "Those of you who are working on massively multiplayer online games, organized crime is already looking at you." A transparent, difficult-to-trace exchange of real-world capital for virtual assets already could provide for cheap and effective international moneylaundering operations – and there is enough money flowing through these games to make such activities feasible. (MindArk, the creator of *Entropia*, reported a 2006 in-game turnover of $350M USD in trade.) Organized crime may also develop new ways of exploiting virtual economies we have yet to contemplate.

6   Conclusions

Despite these risks, virtual worlds will only increase in popularity and richness, and they contribute to our understanding of our own world in important ways. Not only do we get the chance to experiment with alternate forms of economy and governance [8], but we get to do so on a massively shortened timescale. Changes that would be impossible, or at least generations-long, in the real world, can be implemented in virtual environments in a matter of months or years. Because a developer's code allows us to constrain player actions in much the same way the laws of physics do in the real world, we can even make changes that would be impossible in reality.

While there are serious security risks in virtual worlds to both real and virtual assets, virtual worlds are also uniquely equipped to respond to these threats. Working together with game designers will be crucial here because they are not trained to consider privacy and security – they make decisions because of what is good game design. This is a good thing; security and privacy experts have important contributions to offer, and a potential role in shaping what virtual worlds might, someday, be.

References

1. 'Game theft' led to fatal attack: BBC News (online) (March 31, 2005)
2. Murder incorporated. PC Gamer, p. 129 (September 2005)

3. Virtual economies. The Economist (January 20, 2005)
4. Blizzard Entertainment Press Release. World of Warcraft surpasses 8 million subscribers worldwide (January 11, 2007)
5. Brenner, S.W.: Is there such a thing as "virtual crime"? California Criminal Law Review 4(1) (2001)
6. Brown, J.S., Thomas, D.: You play World of Warcraft? You're hired! WIRED 14.04 (April 2006)
7. Castronova, E.: On Virtual Economies. SSRN eLibrary (2002)
8. Grimmelman, J.: Virtual Power Politics. New York University Press (2006)
9. Harper, E.: Cheaters slam 'Everquest II' economy (February 20, 2006)
10. Huizinga, J.: Homo Ludens. Beacon Press (1971)
11. MacKinnon, R.: Virtual rape. Journal of Computer-Mediated Communication 2(4) (March 1997)
12. Silverstein, J.: Are some video games gambling? ABC News (online) (September 8, 2006)
13. Terdiman, D.: Cheaters slam 'Everquest II' economy. CNET News.com (August 11, 2005)
14. Wikipedia (English). Player killer (2007)

Usable SPACE: Security, Privacy, and Context for the Mobile User

Dawn Jutla

Department of Finance, Information Systems, and Management Science
Sobey School of Business
Saint Marys University
Halifax, NS, B3H 3C3
Canada
`dawn.jutla@smu.ca`

Abstract. Users breach the security of data within many financial applications daily as human and/or business expediency to access and use information wins over corporate security policy guidelines. Recognizing that changing user context often requires different security mechanisms, we discuss end-to-end solutions combining several security and context mechanisms for relevant security control and information presentation in various mobile user situations. We illustrate key concepts using Dimitri Kanevskys (IBM Research) early 2000s patented inventions for voice security and classification.

Personal Digital Rights Management for Mobile Cellular Devices

Siddharth Bhatt[1], Bogdan Carbunar[2], Radu Sion[1], and Venu Vasudevan[2]

[1] Network Security and Applied Cryptography Lab
Computer Science, Stony Brook University
{sbhatt,sion}@cs.stonybrook.edu
[2] Pervasive Platforms and Architectures
Motorola Labs
{carbunar,vvenu}@motorola.com

Abstract. Driven by an unparalleled advance in network infrastructure support as well as a boom in the number of interconnected personal communication, computation and storage devices, the modern mobile customer experience has become increasingly compelling. Traditional barriers between the roles of information consumer and producer have disappeared. Users increasingly produce and distribute valuable and often personal content such as pictures and free or purchased copyrighted media. It becomes essential to enable user-level DRM controls for content access, data integrity and rights management.

In this presentation we will overview the design and implementation of a a personal digital rights management system for mobile devices. The Personal DRM Manager enables user-defined ORCON-type controls for personal content originating in a cell phone or other mobile device. Users can transparently define, generate, package and migrate content licenses between mobile devices on-demand. Networked cellular devices cooperate in the enforcement mechanisms.

Design. Main user-land components include: license generation, content packaging, secure networking over both blue-tooth and 802.11, and compliant content rendering. A device can naturally act as both sender and recipient. Additionally, we developed a public key infrastructure to allow devices to be both uniquely identified and to allow for session key exchanges with forward security.

We demonstrate the main features of our system, involving a set of live E680i GSM devices. A subset of the devices will generate content and transparently associate both state-full and state-less use licenses with the newly generated content. We will illustrate scenarios ranging from simple use-counters (this content cannot be played more than N times) to more complex conditional licenses such as (this content can be accessed only between October 7th and October 9th at full resolution and down-sampled to 56KBps otherwise). Additionally, we will discuss a few of the design choices for both the enforcement and the secure network hand-shake protocols.

E680i Platform. The Motorola E680i is a multi-feature palm-size embedded-linux based cell phone with direct MPEG4 video capture and playback, a real-time 3D sound engine and 3D stereo speakers, an integrated MP3 player, a large capacity internal memory of up to 2GBytes, a removable SD memory card slot, a 240 x 320 color screen, and an integrated VGA camera with 8x zoom.

Certificate Revocation Using Fine Grained Certificate Space Partitioning

Vipul Goyal⋆

Department of Computer Science
University of California, Los Angeles
`vipul@cs.ucla.edu`

Abstract. A new certificate revocation system is presented. The basic idea is to divide the certificate space into several partitions, the number of partitions being dependent on the PKI environment. Each partition contains the status of a set of certificates. A partition may either expire or be renewed at the end of a time slot. This is done efficiently using hash chains.

We evaluate the performance of our scheme following the framework and numbers used in previous papers. We show that for many practical values of the system parameters, our scheme is more efficient than the three well known certificate revocation techniques: CRL, CRS and CRT. Our scheme strikes the right balance between CA to directory communication costs and query costs by carefully selecting the number of partitions.

1 Introduction

A certificate is a digitally signed statement binding the key holder's (principal's) name to a public key and various other attributes. The signer (or the issuer) is commonly called a certificate authority (CA). Certificates act as a mean to provide trusted information about the CA's declaration w. r. t. the principal. The declaration may be of the form:

"We, the Certificate Authority, declare that we know Alice. The public key of Alice is ..."

"We further declare that we trust Alice for ..." (optional part)

Certificates are tamper-evident (modifying the data makes the signature invalid) and unforgeable (only the holder of the secret, signing key can produce the signature). Certificates are the building blocks of a Public Key Infrastructure (PKI).

When a certificate is issued, the CA declares the period of time for which the certificate is valid. However, there may be situations when the certificate must abnormally be declared invalid prior to its expiration date. This is called certificate revocation. This can be viewed as "blacklisting" the certificate. This means that the existence of a certificate is a necessary but not sufficient evidence

for its validity. A method for revoking certificates and distributing this revocation information to all the involved parties is thus a requirement in PKI. The reasons for revoking a certificate may be: suspected or detected key compromise, change of principal name, change of relationship between a principal and the CA (e.g., Alice may leave or be fired from the company) or end of CA's trust into the principle due to any possible reason.

The revocation mechanism should have an acceptable degree of timeliness, i.e., the interval between when the CA made a record of revocation and when this information became available to the relying parties should be small enough to be acceptable. Further, it is very important for the revocation mechanism to be efficient as the *running* expenses of a PKI derives mainly from administering revocation [Stu95].

Existing Techniques for Certificate Revocation. Certificate Revocation List (CRL) is the first and the simplest method of certificate revocation. A CRL is a periodically issued and digitally signed list containing the serial number of all the revoked certificates issued by a particular CA. However, it is widely recognized [Mic97, Goy04, Riv98] that CRLs are too costly and cannot provide a good degree of timeliness. Certificate Revocation System (CRS) [Mic96, Mic97, Mic02] was introduced by Micali and could answer the user queries with exceptional efficiency. The main problem with CRS is that it is not suitable in case of a distributed query answering system. The CA to directory communication[1] is too high shooting up the overall cost of the system [NN98, ALO98]. Aiello et al [ALO98] proposed an improvement to CRS aimed at reducing this communication but their approach had problems as we discuss in section 2.2. Certificate Revocation Tree (CRT) [Koc98] is the third well known technique for certificate revocation. Though the CA to directory communication is very low, the query cost is too high, again shooting up the overall cost of the system.

Another technique for certificate revocation is the Online Certificate Status Protocol (OCSP) [OCSrg] designed by IETF. In OCSP, the CA simply digitally signs the response to a certificate status query. Thus, OCSP may provide very high degree of timeliness but is recognized to be non-scalable since the CA is required to compute a signature for answering every query. Further, OCSP has no distributed implementation, i.e., it cannot be used in the settings where there should be a number of un-trusted directories answering the user queries. Using techniques from Identity based encryption [BF01], Gentry [Gen03] proposed a new cryptosystem having attractive properties in terms of revocation. However it was not a generic revocation solution and could not be used with existing cryptosystems (such as RSA). Other revocation techniques include [DBW01, BLL00, GGM00, MJ00]. See [Zhe03] for an analysis of these techniques.

Our Contribution. Motivated by the CA to directory communication cost (CDCC) and the query cost (QC) imbalances in both CRS and CRT, we propose a new system aimed at balancing the two. CRS and CRT are the two extremes,

---

[1] The terms "CA to directory communication cost" and "directory update cost" are used interchangeably in this paper.

CDCC being very high and QC being extremely low in the former, and CDCC being extremely low and QC being too high in the latter. Our technique aims at striking the right balance between CDCC and QC to minimize the system cost. The basic idea is to divide the total certificate space into several partitions. The number of partitions is a key parameter which can be optimized to reduce the overall communication cost. Each partition has a unique serial number, is digitally signed and contains the status of a set of certificate. At the end of a time slot, a partition may either expire or be renewed depending upon whether there was a status change for any of the certificates covered by it or not. Renewing a partition is done by exposing a link of the hash chain whose tip is embedded in that partition. Our system is named CSPR (certificate space partitioning with renewals).

As we show in section 4, the overall communication cost of our system is less than the three widely used revocation techniques, i.e., CRL, CRS and CRT for many practical values of the system parameters.

Rest of the paper is organized as follows: section 2 gives a background on hash chains and common certificate revocation techniques, section 3 introduces the proposed system called CSPR, section 4 evaluates the CSPR costs and compares it with other techniques in use, section 5 concludes the paper.

2   Background

2.1   Hash Chains

A hash chain of length L is constructed by applying a one-way hash function $H(.)$ recursively to an initial seed value s.

$$H^L(s) = \underbrace{H\left(H\left(...H\left(s\right)\right)\right)}_{\text{L times}}$$

The last element $H^L(s)$, also called the tip T of the hash chain, has the property that using $H^L(s)$, $H^{L-1}(s)$ can not be computed but its correctness can be verified.

2.2   Certificate Revocation Techniques

Certificate Revocation List (CRL) is the first and the simplest method of certificate revocation. A CRL is simply a periodically issued, time-stamped and digitally signed list containing the serial number of all the revoked certificates issued by a particular CA.

Certificate revocation status (CRS) was introduced by Micali [Mic96, Mic97, Mic02]. It was also patented and commercialized. The basic idea is as follows. For certificate creation, the CA chooses two random numbers Y_0 and N_0 and computes $Y = H^{365}(Y_0)$ and $N = H(N_0)$. These two fields are included in the certificate and are signed along with the other usual fields. The number 365 denotes the number of days in the year. On the ith day,

1. if the certificate is revoked, the CA releases N_0, which can be verified by hashing and comparing with N specified in the certificate.
2. if the certificate is still valid, the CA releases $H^{365-i}(Y_0)$ which can be verified by hashing i times and comparing with Y specified in the certificate.

Aiello et al [ALO98] extended CRS by reducing the overall CA to directory communication while still maintaining the same tiny query communication. This is done by including $\log_2(N)$ hash chain tips in each certificate, N being the number of certificates in the system. Although the reduction in CDCC was high for low revocation rates, the same cannot be said for system with higher revocation rates. Further, this improvement comes at the price of a significant increase in the certificate transmission costs due to increase in the certificate size.

Certificate Revocation Tree (CRT) was introduced by Kochar [Koc98, NN98]. A CRT is based on a Merkle hash tree [Mer89] containing certificate serial number ranges as the tree leaves. The root of the hash tree is signed by the CA. Now, the certificate status proof for a certificate with serial number s consists of the path node siblings from the root to the appropriate leaf (having s in its range), in addition to the signature on the root of the tree.

3   The Proposed Technique

We start by explaining a few notations to be used in the rest of the paper.

N The total number of certificates handled by the CA

R The estimated number of certificates (out of N) that will eventually be revoked prior to expiration

T The number of time slots for which a certificate is issued. One time slot is the duration between two certificate status information releases by the CA (e.g., for a weekly CRL, time slot is one week). It represents the maximum amount of time between when a certificate gets revoked by the CA and when the new status is made available to the relying parties.

q Estimated average number of queries per day handled by the system

T_i Representation of the ith 'absolute' time slot, i.e., the ith time slot after the CA has started operation

P The number of partitions in which the total certificate space of N certificates is divided. P is the key parameter in our scheme. We discover later the technique to find out the optimal value of P for a given set of system parameters.

S_i The serial number of the ith partition

n The number of certificates whose status is contained in one partition. We have $n \times P = N$

P_i^j The version of the ith partition created and released at the beginning of jth time slot T_j. We talk more about versions of a partition later.

D Number of directories in the system

U	Number of updates to the directories per day. Hence, U is equal to the number of time slots in one day.
$S_{CA}(M)$	Signature on the message M with the private key of the CA.
L_{SR}	The number of bits needed to hold a serial number (of a certificate or partition)
L_H	The length of the hash function output in bits
L_S	The length of a digital signature in bits
L_P	The length of a partition in bits
L_T	The number of bits needed to hold a time slot number

3.1   Creating Partitions

Unlike CRS and like CRT and CRL, our solution works for non custom built certificates. While CRS requires the certificates to have two additional fields Y and N, our technique can be used with any set of certificates having serial numbers. This may be especially important while migrating an existing PKI from one certificate revocation solution to another.

As discussed before, the basic idea is to divide the certificate space into a number of partitions, each partition containing revocation status of the certificates with serial numbers in a particular range. Note that a partition may have several 'versions'. When a certificate contained in a partition changes status (i.e., gets revoked), the current version of that partition is said to have expired and a new version, reflecting the new certificate status, is created and released by the CA at the beginning of the next time slot. If none of the certificates in a particular partition gets revoked during a time slot, the same version is renewed by the CA by exposing a hash chain link at the beginning of the next time slot. The details follow.

The CA divides the whole certificate space into P partitions[2], each partition containing the status information of n certificates having consecutive serial numbers. Each partition is given a unique serial number which is one less than the serial number of the first certificate in that partition. Hence, a partition having serial number S_i contains the revocation status of certificates with serial numbers from $S_i + 1$ to $S_i + n$. Each partition contains a field called "Certificate Status Data" (CSD). This field contains the revocation status of all the certificates in that partition. The jth bit of the CSD of the ith partition is 0 if the certificate with serial number $S_i + j$ is revoked and is 1 otherwise. Clearly, the size of this field is n bits, one bit each for holding the status of one certificate. We represent the CSD field of the ith partition as CSD_i.

For creating a version P_i^j of the ith partition to be released at the beginning of T_j, a hash chain of length L with seed k_i^j is constructed and its tip is specified in the partition. We talk more about the choice of L later. The seed k_i^j is chosen randomly by the CA. We have:

$$P_i^j = S_{CA}\left(T_j, S_i, H^L\left(k_i^j\right), CSD_i\right) \tag{1}$$

---

[2] We discuss how to select P optimally later on.

Where CSD_i is the certificate status data at the beginning of T_j.

It is also possible to later add more certificates to the already existing set of certificates by adding new partitions. Since certificates may not always be added in chunks of n, we allow the new partition being added to have some non-existent certificates also. More precisely, a new partition will be created with all of its n CSD bits as 1. It is possible that there may not yet exist certificates with some of the serial numbers lying in the serial number range of that partition. Consequently, as more and more certificates are later added, there will be no need to add more partitions until there are no non-existent certificates in that partition.

3.2   System Operation

At the beginning of the time slot T_k, the CA does the following:

1. For all the partitions for which there was no change in the certificate status data (i.e., none of the certificates in that partition were revoked) during time slot T_{k-1}, the CA reveals the next link of the hash chain. For partition P_i^j, the link $H^{L-(k-j)}(k_i^j)$ is revealed. All the directories in the system are updated with this new hash chain link. Hash chain traversal techniques [Jak02, CJ02, Sel03] may be used by the CA to efficiently compute the next link to be revealed. Note that CSD need not be changed for a certificate which expired (but was not revoked) during the time slot T_{k-1}. Hence, it is perfectly possible that the status of a certificate is 1 even after it has expired.

2. For all the partitions for which the certificate status data changed during time slot T_{k-1}, a new version is released by the CA. The new version P_i^k for P_i^j is created as follows:

$$P_i^k = S_{CA}\left(T_k, S_i, H^L\left(k_i^k\right), CSD_i\right)$$

Where CSD_i is the new certificate status data for this partition.

As an optimization, the CA need not send the whole new partition version to the directories. Instead, only the information which enables the directories to create the new partition version using the older version is sent. We call this information the partition update information (PUI). For a partition, the serial number of all the certificates revoked from it along with the new hash chain tip and the new signature suffice as PUI.

Now, during time slot T_k, a directory answers a user certificate status query for a serial number s by first locating the appropriate partition and then sending that partition and the latest hash chain link revealed for that partition to the user. More precisely, the directory finds an S_i s.t. $(S_i + 1) \leq s \leq (S_i + n)$, locates its current (un-expired) version P_i^j, and then sends back P_i^j along with $H^{L-(k-j)}(k_i^j)$. Note that there is no need to trust the directory for sending the un-expired versions only. A directory will be unable to produce the hash chain link $H^{L-(k-j)}(k_i^j)$ for a version P_i^j which expired prior to the beginning of time slot T_k.

The verifier can verify the status of the certificate from the response by:

1. making sure that $(S_i + 1) \le s \le (S_i + n)$ and the $(s - S_i)$ th bit of the CSD contained in the sent P_i^j is 1,
2. verifying the CA signature on P_i^j,
3. verifying that hashing the sent hash chain link (*current time slot – time slot number T_j specified in P_i^j*) times matches with the hash chain tip specified in P_i^j.

Now we comment on the choice of L, the length of the hash chain. If a partition does not expire due to changes in its CSD, it will automatically expire when its hash chain links are exhausted. At this point, a new version will have to be created by the CA. Thus, simply, L should be large enough to ensure that the probability of the hash chain links being exhausted before the partition expires is reasonably low. A good choice for L seems to be T, the total number of time slots for which a certificate is valid. Assuming $L = T$, by the time the hash chain gets exhausted, all the certificates in the partition would have already expired.

3.3   System Costs

Now we determine the average daily cost of the proposed system in terms of bits. We have the following.

CA to Directory Communication Cost per Update (CDCCPU). CA to directory communication per update is comprised of the hash chain links for the unexpired partitions and the PUI for the expired partitions. The aggregate of all PUIs consists of the serial numbers of all the certificates revoked during the previous time slot plus the new hash chain tip and signature for all the expired partitions. Clearly, the average number of certificates revoked per time slot is R/T. Assuming E to be the average number of partitions expiring per time slot, we have the following:

$$CDCCPU = (P - E) L_H + \frac{R}{T}.L_{SR} + E (L_H + L_S)$$

Or,

$$CDCCPU = P.L_H + \frac{R}{T}.L_{SR} + EL_S \qquad (2)$$

E is clearly upper bounded by the number of certificates being revoked per time slot, i.e., R/T. However, since multiple revoked certificates may be from the same partition, E may actually be lesser than this value. E is equal to the number of bins having at least one ball when R/T balls are thrown into P bins. Hence, E may be computed as follows:

Probability of the ith bin not having the ball when a ball is thrown into P bins $= (1 - \frac{1}{P})$

Probability of the ith bin not having a ball when R/T balls are thrown into P bins $= (1 - \frac{1}{P})^{\frac{R}{T}}$

Expected number of bins having at least one ball = Total number of bins – Expected number of bins having no balls

Or,

$$E = P - P\left(1 - \frac{1}{P}\right)^{R\!/T} \tag{3}$$

Directory Query Cost per Day (QC). A query response consists of the appropriate partition plus a hash chain link indicating the proof of renewals. Hence, the daily query cost QC is

$$QC = q.\left(L_P + L_H\right)$$

Total System Cost per Day (TC). Total daily cost TC of the system consists of updating each of the D directories U times plus the query costs. Hence, we have:

$$TC = U.D.\left(P.L_H + \frac{R}{T}.L_{SR} + EL_S\right) + q.\left(L_P + L_H\right)$$

Further, from (1), we have:

$$L_P = L_T + L_{SR} + L_H + \frac{N}{P} + L_S$$

Thus,

$$TC = U.D.\left(P.L_H + \frac{R}{T}.L_{SR} + EL_S\right) + q.\left(L_T + L_{SR} + 2.L_H + \frac{N}{P} + L_S\right) \tag{4}$$

3.4   Optimal Number of Partitions

In this section, we determine the optimal value of P to minimize the total daily system cost. As will be clear in the next section, the total number of partitions P would usually be much higher than the number of certificates expiring per time slot. Hence, to simplify our analysis, we approximate E by R/T. We put this approximation of E in equation (4) and compute the minima of its R.H.S. using differentiation. Differentiating R.H.S. of (4) w.r.t. P and putting the result equal to zero, we get:

$$U.D.L_H + q\left(-\frac{N}{P^2}\right) = 0$$

or,

$$P = \sqrt{\frac{N.q}{U.D.L_H}} \tag{5}$$

The above equation also gives useful insights on the issue of PKI expansion. As more and more certificates are added to the PKI to increase N, the number

of queries per day, i.e. q, is also expected to increase by the same factor. This is because the number of daily verifications of a certificate (and hence the queries pertaining to it) is independent of N. q is also expected to increase linearly with N in the Rivest's model [Riv98] of "Certificate holder supplies all validity evidences to the verifier" in which, instead of the verifier, the holder sends periodic queries to obtain recent validity evidence for its certificate. Thus from (5), along with q, P also increases linearly with N. This means that the optimal size of (or the number of certificates in) a partition, i.e., n $(=N/P)$ remains unaltered as the PKI expands. Hence, no extra efforts are needed to maintain optimality as the size of the PKI changes.

The above argument may not hold in some specialized PKIs. It is still possible to maintain optimality in such systems by periodically re-computing the optimal value of n and setting it as the partition size for new partitions being created. As certificates expire, older partitions will continue to get removed from the system[3]. Hence, the system forever keeps 'migrating' to the (currently) optimal value of the partition size.

4   Evaluation and Comparison

Following the framework of CRS [Mic96] and CRT [NN98], we evaluate the cost of the proposed system called CSPR from hereon. The following values of the parameters are assumed.

$N = 3 \times 10^6$, $R = 3 \times 10^5$, $q = 3 \times 10^6$, $T = 365 \times U$, $L_H = 160$, $L_S = 1000$, $L_{SR} = 30$, $L_T = 20$

The above values are mostly taken from [NN98]. The evaluations are done for the values of D between 0 and 10,000 and U between 1 and 100. Observe the value of T, the number of time slots. Since we assume that a certificate is issued for one year; T is equal to the number of days in one year (365) multiplied by the number of time slots in one day (U).

The comparison is done with CRL, CRS and CRT. Before going further, we compute the daily communication cost of each of these techniques.

Certificate Revocation Lists Daily Cost. Average CRL Size $= \frac{R}{2}.L_{SR} + L_S$

$$\text{Total Daily Cost} = D.U \left(\frac{R}{2}.L_{SR} + L_S \right) + q \left(\frac{R}{2}.L_{SR} + L_S \right) \qquad (6)$$

Certificate Revocation System Daily Cost.

$$\text{Total Daily Cost} = D.U \left(N.L_H \right) + q.L_H \qquad (7)$$

---

[3] A partition will be removed from the system when all the certificates in it get expired.

Certificate Revocation Tree Daily Cost. CA to Directory Communication per update consists of the serial numbers of the certificates revoked during the previous time slot and the new signature on the root of the Merkle tree. Query response consists of tree path node siblings from the appropriate leaf to the root along with the root signature. Hence, we have:

$$\text{Total Daily Cost} = D.U\left(\frac{R}{T}.L_{SR} + L_S\right) + q\left(L_H \log \frac{R}{2} + L_S\right) \qquad (8)$$

Table 1 summarizes the total daily system costs in bits for CRL, CRS and CRT using (6), (7) and (8) respectively and for CSPR using (4) with optimal number of partitions found using (5). The values are computed for various choices of D and U.

Table 1. Daily System Costs in Bits for Common Revocation Techniques

	CRL	CRS		CRT	CSPR
$D = 0$	1.3×10^{13}	4.8×10^8	(4.8×10^8)	1.1×10^{10}	4.1×10^9
$D = 10, U = 1$	1.3×10^{13}	5.3×10^9	(2.3×10^9)	1.1×10^{10}	4.3×10^9
$D = 100, U = 1$	1.3×10^{13}	4.8×10^{10}	(1.8×10^{10})	1.1×10^{10}	4.9×10^9
$D = 100, U = 10$	1.3×10^{13}	4.8×10^{11}	(1.8×10^{11})	1.1×10^{10}	6.6×10^9
$D = 100, U = 100$	1.4×10^{13}	4.8×10^{12}	(1.8×10^{12})	1.1×10^{10}	1.2×10^{10}
$D = 1000, U = 100$	1.4×10^{13}	4.8×10^{13}	(1.8×10^{13})	1.1×10^{10}	2.9×10^{10}
$D = 10,000, U = 100$	1.8×10^{13}	4.8×10^{14}	(1.8×10^{14})	1.3×10^{10}	8.8×10^{10}

The improvement of CRS due to [ALO98] appears in parenthesis. It should be stated here that this improvement comes at the cost of increasing the certificate size by 3.5 KB. Assuming one certificate transmission per revocation status query, the increase in certificate transmission costs comes out to be 1.1×10^{10} bits per day (not added in the table) .

Table 2 lists the optimal values of P and the corresponding n found using (5).

Remark 1. As demonstrated by the above values, CRS and CRT are the two extremes, CRS having unbeatable query costs but having CA to Directory Communication as bottleneck and CRT having unbeatable CA to Directory Communication

Table 2. Optimal Values of P and the corresponding n for various values of D and U

	Number of Partitions P	Certificates per partition n
$D = 0$	3.0×10^6	1×10^0
$D = 10, U = 1$	7.5×10^4	4.0×10^1
$D = 100, U = 1$	2.4×10^4	1.2×10^2
$D = 100, U = 10$	7.5×10^3	4.0×10^2
$D = 100, U = 100$	2.4×10^3	1.2×10^3
$D = 1000, U = 100$	7.5×10^2	4.0×10^3
$D = 10000, U = 100$	2.4×10^2	1.2×10^4

Table 3. Revocation Technique Selection

Distribution Degree and System Timeliness	CRS	CRT	CSPR
Centralized system or a system having very low distribution degree	*Suitable*	*QC High*	*QC High*
Moderately low to moderately high timeliness or distribution degree	*CDCC High*	*QC High*	*Suitable*
Very high timeliness or distribution degree	*CDCC High*	*Suitable*	*TC High*

but having query costs as the bottleneck. The proposed technique is able to balance the two costs by optimally choosing the number of partitions P and thus minimizing the overall cost of the system. Hence, P is the key parameter for CSPR.

Remark 2. The formula for the total cost of the system may be modified by assigning suitable weight to CDCC and QC. For example, in some low budget systems, CDCC may be assigned more weight to prevent CA from becoming the communication bottleneck, while in others, QC may be assigned more weight to improve the user experience. Accordingly, the formula for P is modified (by taking the weight during differentiation) and P is still able to play the role of cost balancer between CDCC and QC.

Remark 3. The computation required to validate a certificate status proof is similar in CRL, CRT and CSPR (dominated by a signature verification). For lower update rates, CRS has an advantage as it does not require a signature verification. However, as update rate increases, the computation starts becoming comparable to others. This is because the average number of hash function evaluations required for validating one certificate status proof in CRS is $365 \times U/2$.

Future Work. An interesting observation in the proposed system is that the partitions (without hash chains) may actually be treated as ordinary certificates and our scheme may be recursively applied. The notion of partition expiry may be replaced with the notion of partition revocation. Level 2 partitions may be created which will contain the status of a number of level 1 (ordinary) partitions having consecutive serial numbers. A Level 2 partition will have hash chains as the renewal mechanism and will expire as soon as any of the level 1 partitions in it expires (gets revoked). Similarly, level 3 partitions and so on are possible, the extreme being a tree of partitions of different levels. The above approach may be worth exploring in environments where the number of directories or updates per day is high. This is because it may reduce the CA to directory communication costs which are quite high in such environments, though at the price of increasing the query costs.

5   Conclusions

We conclude by summarizing the suitability of various revocation schemes under different values of D and U in Table 3. D is an indicator of the distribution degree of the system while U is an indicator of system timeliness.

References

[ALO98]   Aiello, W., Lodha, S., Ostrovsky, R.: Fast digital identity revocation (extended abstract). In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 137–152. Springer, Heidelberg (1998)

[BF01]   Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)

[BLL00]   Buldas, A., Laud, P., Lipmaa, H.: Accountable certificate management using undeniable attestations. In: ACM Conference on Computer and Communications Security, pp. 9–17 (2000)

[CJ02]   Coppersmith, D., Jakobsson, M.: Almost optimal hash sequence traversal. In: Blaze, M. (ed.) FC 2002. LNCS, vol. 2357, pp. 102–119. Springer, Heidelberg (2003)

[DBW01]   Dan Boneh, G.T., Ding, X., Wong, C.M.: A method for fast revocation of public key certificates and security capabilities. In: The 10th USENIX Security Symposium, pp. 297–308 (2001)

[Gen03]   Gentry, C.: Certificate-based encryption and the certificate revocation problem. In: Biham, E. (ed.) EUROCRPYT 2003. LNCS, vol. 2656, pp. 272–293. Springer, Heidelberg (2003)

[GGM00]   Gassko, I., Gemmell, P., MacKenzie, P.D.: Efficient and fresh certification. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 342–353. Springer, Heidelberg (2000)

[Goy04]   Goyal, V.: Certificate revocation lists or online mechanisms. In: Fernández-Medina, E., César Hernández Castro, J., Javier García-Villalba, L.(eds.) WOSIS, pp. 261–268, INSTICC Press (2004)

[Jak02]   Jakobsson, M.: Fractal hash sequence representation and traversal. In: ISIT 2002, http://eprint.iacr.org/2002/001andwww.markus-jakobsson.com

[Koc98]   Kocher, P.C.: On certificate revocation and validation. In: Hirschfeld, R. (ed.) FC 1998. LNCS, vol. 1465, pp. 172–177. Springer, Heidelberg (1998)

[Mer89]   Merkle, R.C.: A certified digital signature. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 218–238. Springer, Heidelberg (1990)

[Mic96]   Micali, S.: Efficient certificate revocation. Technical Report MIT/LCS/TM-542b (1996)

[Mic97]   Micali, S.: Efficient certificate revocation. In: Proceedings 1997 RSA Data Security Conference (1997)

[Mic02]   Micali, S.: Novomodo: Scalable certificate validation and simplified pki management. In: 1st Annual PKI Research Workshop - Proceeding (2002)

[MJ00]   McDaniel, P.D., Jamin, S.: Windowed certificate revocation. In: INFOCOM, pp. 1406–1414 (2000)

[NN98]   Naor, M., Nissim, K.:Certificate revocation and certificate update. In: Proceedings 7th USENIX Security Symposium (San Antonio, Texas) (January 1998)

[OCSrg]   Online certificate status protocol: version 2. In: Working document of the Internet Engineering Task Force (IETF), RFC 2560, http://www.ietf.org

[Riv98]   Rivest, R.L.: Can we eliminate certificate revocations lists? In: Hirschfeld, R. (ed.) FC 1998. LNCS, vol. 1465, pp. 178–183. Springer, Heidelberg (1998)

[Sel03]   Sella, Y.: On the computation-storage trade-offs of hash chain traversal. In: Wright, R.N. (ed.) FC 2003. LNCS, vol. 2742, pp. 270–285. Springer, Heidelberg (2003)

[Stu95]   Stubblebine, S.: Recent-secure authentication: Enforcing revocation in distributed systems. In: Proceedings 1995 IEEE Symposium on Research in Security and Privacy, pp. 224–234 (May 1995)

[Zhe03]   Zheng, P.: Tradeoffs in certificate revocation schemes. Computer Communication Review 33(2), 103–112 (2003)

An Efficient Aggregate Shuffle Argument Scheme

Jun Furukawa[1] and Hideki Imai[2]

[1] NEC Corporation, 1753, Shimonumabe, Nakahara, Kawasaki 211-8666, Japan
j-furukawa@ay.jp.nec.com
[2] Research Center for Information Security,
National Institute of Advanced Industrial Science and Technology, Japan
h-imai@aist.go.jp

Abstract. In this paper, we propose a novel scheme to prove the correctness of mix-net that is composed of multiple shufflings, in such a way that the computational complexity of its verifier does not depend on the number of its composite shufflings. We call this scheme an *aggregate shuffle argument scheme*. Although a similar scheme proposed by Abe in Eurocrypt 1998 exists, our scheme is much more efficient. In fact, the computational cost required for the verifier in our scheme is less than 1/60 of that in Abe's scheme. This is mainly because our scheme exploits the efficient shuffle arguments proposed of Furukawa et al. in Crypto 2001 while Abe's scheme exploits the shuffle proof proposed by Sako et al. in Eurocrypt 1995. We also proposed a formal model and security requirements of aggregate shuffle argument schemes.

Keywords: Voting, Shuffle, Aggregate, Mix-net, Efficient.

1 Introduction

The notion of mix-net, first introduced by Chaum in [4], is useful in applications which require anonymity, such as voting. The core ingredient in mix-net is the execution of multiple rounds of shuffling and decryption by multiple, independent mixers, so that none of the output decryptions can be linked to any of the input encryptions. To ensure the correctness of output, it is desirable to achieve the property of universal verifiability. Early studies, such as those in [22] and [2], required vast amounts of computation to prove the correctness of a shuffling in a mix-net without sacrificing unlinkability. The inefficiency of these schemes had been the one of the serious problems that are preventing mix-nets from applying them to very large-scale voting.

There are roughly two types of previous works that reduce the amount of computation for verification of mix-net. The straight forward way is to increase the efficiency of each component. In [7,8,9,10,12,16,13,5,21], efficient and practical schemes for proving the correctness of each component shuffling or of each pair of shuffling and decryption are proposed. On the other hand, in [1], Abe proposed a scheme to prove the correctness of total mix-net shufflings in such

a way that the amount of computation of its verifier does not depend on the number of shufflings executed in the mix-net. We call such a scheme an aggregate shuffle argument scheme. The essence of the latter scheme is to consider the whole mix-net as a single shuffle and let all mixers collaboratively play the role of its single prover. Then, the verification cost for the mix-net will be reduced to that for a single shuffle. Here, the proof of shuffle collaboratively generated is the one proposed by Sako et al. in [22].

Since the computational cost for the verifier in the Sako's scheme in [22] is huge, so is that for the verifier in Abe's scheme. Hence, Abe's mix-net is advantageous over the mix-net composed of efficient shuffles only when a large number of shufflings is executed in a single mix-net. However, in such a situation, neither the Abe's nor efficient-shuffle-based mix-nets are efficient enough to be applied to huge scale voting.

In this paper, we propose a novel aggregate shuffle argument scheme that is much more efficient than the one proposed in [1] by exploiting schemes proposed in [8,10] instead of the scheme proposed in [22]. We also propose a model and security requirements of aggregate shuffle argument schemes.

The basic idea of our scheme is to consider the whole mix-net as a single shuffle and let all mixers collaboratively play the role of a prover of the zero-knowledge argument for shuffling proposed in [8]. This is certainly possible by the technique of general multi-party computation. However, such a general solution requires of each mixer an unrealistic amount of computation, an amount on the order of the square of the number of mixers. Hence, our goal is to propose an efficient way for multiple mixers to collaboratively play the role of a single prover in the shuffle argument in [8] as Abe has proposed for the shuffle proof in [22]. Although we could not achieve it directly, we solved the problem in two steps. We first developed a zero-knowledge argument for shuffling, a new variant of [8], that is specifically designed for our purpose. Then, we proposed an efficient protocol for multiple mixers to collaboratively play the role of its single prover. Here, we leverage the property that the permutation matrices form a group. As a result, our scheme requires of each mixer an amount of computation that is linear to the number of all the mixers.

Suppose that the number of ciphertexts to be mixed is k and that of mixers is λ. Then, the verifier in our scheme is required to compute $10k$ exponentiations to verify the correctness of total mixing, and each mixer in our scheme is required to compute $28k$ and $13k(\lambda - 1)$ exponentiations, respectively, to prove the correctness of its shuffling and to verify the correctness of shuffling of all other mixers. On the other hand, each mixer and verifier in the scheme proposed in [1] are required to compute, respectively, $640k$ and $320k + 640k\lambda$ exponentiations. If we use the scheme proposed in [12] for multiple times, each mixer and verifier are required to compute, respectively, $6k\lambda$ and $6k + 6k\lambda$ exponentiations.

It should be mentioned here that, although it is theoretically possible for the verifiers to collaboratively and interactively play the role of the single verifier in our mix-net, we do not consider it as a realistic solution for very large-scale voting such as one for a national election. Rather, as is the case considered in

[9], we consider the case that all mixers collaboratively generate a Fiat-Shamir transformation of their proof instead of interactively proving their correctness. Then, each of the verifiers is able to independently verify their correctness.

Note that, although Abe's scheme can be fixed trivially, it does not satisfy our security requirements. Each mixer needs to additionally verify the correctness of the output of the previous mixer to be a secure aggregate shuffle argument scheme in our sense.

An Example Application: We show an example case in which our proposed scheme is effective. Suppose that a mix-net is applied to very large-scale voting such as a national election in which voter anonymity is strongly required. In such a case, since the coalition of all the mixers is able to link input encryptions to output decryptions, an increase of the number of mixers in the mix-net is strongly required.

However, in such a large-scale voting, even if one of the previous efficient schemes [7,8,9,10,12,16,13,5,21], where [13] is the most efficient, is used for proving each shuffling, the amount of computation for provers and verifiers will unavoidably be large. Therefore, increasing the number of mixers, i.e., achieving high level of anonymity, puts a heavy computation load on mixers and verifiers. Such a situation is especially undesirable for verifiers, since some of them, such as voters in general, may have only small computational resources.

Since the amount of computation for verifiers in our scheme does not depend on the number of mixers, it is easy to increase the number of mixers in our scheme, i.e., it is easy to achieve a high level of anonymity. Moreover, since our scheme exploits a variant of an efficient arguments for shuffling proposed in [10,8], it can handle large-scale voting unlike the scheme proposed in [1].

Organization: Our paper is organized as follows. Section 2 describes the model and security requirements for aggregate shuffle argument schemes. Section 3 describes the basic idea of our proposed scheme by introducing an example aggregate permutation argument scheme. Section 4 proposes an aggregate shuffle argument scheme which is secure only when some of the players are assumed to be honest and discusses its security requirements. Section 5 presents a method to transform the scheme proposed in Section 4 into a full scheme which is secure even if all players are malicious. Section 6 estimates the efficiency of our scheme and compares it with prior works.

2   Aggregate Shuffle Argument Schemes

2.1   ElGamal Shuffling

Let $\mathcal{G}_q$ be an cyclic group of prime order q in which the discrete logarithm problem is difficult to solve, g be a generator of $\mathcal{G}_q$, a set $\mathcal{K}$ be $\{1, \ldots, k\}$, and $(g_i^{(1)}, m_i^{(1)})_{i \in \mathcal{K}} \in (\mathcal{G}_q \times \mathcal{G}_q)^k$ be a sequence of k ElGamal ciphertexts to be shuffled by using the public-key $m \in \mathcal{G}_q$.

ElGamal shuffling is a procedure that, given G, M and k ElGamal ciphertexts $(g_i, m_i)_{i \in \mathcal{K}}$, randomly choose permutation of indices $\phi : \mathcal{K} \to \mathcal{K}$, randomly choose random numbers $\{\varphi_i \in_R \mathbb{Z}/q\mathbb{Z}\}_{i \in \mathcal{K}}$ for reencryptions, and then outputs ElGamal ciphertexts

$$(g_i', m_i')_{i \in \mathcal{K}} = (g_{\phi(i)} g^{\varphi_i}, m_{\phi(i)} m^{\varphi_i})_{i \in \mathcal{K}}. \tag{1}$$

Shuffling of ElGamal ciphertexts results in the following two properties:

1. There exists a permutation ϕ such that $D((g_i', m_i')) = D((g_{\phi(i)}, m_{\phi(i)}))$ holds for all i. Here, $D(\cdot)$ is a decryption algorithm.
2. As long as the decision Diffie-Hellman problem is difficult to solve, no polynomially bounded algorithm, given only $q, g, m, \{(g_i, m_i)\}_{i \in \mathcal{K}}, \{(g_i', m_i')\}_{i \in \mathcal{K}}$, has an advantage over the random-guessing algorithm in guessing any part of permutation ϕ for randomly chosen $g, m, \{\varphi_i\}_{i \in \mathcal{K}}, \phi$.

2.2   Model of Aggregate Shuffle Argument Schemes

Let k be the number of ciphertexts input to the mix-net, and λ be the number of mixers. Players in aggregate shuffle argument scheme are a set of λ mixers ($\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\ldots,\lambda}$), an aggregator $\mathcal{A}$, and a verifier $\mathcal{V}$. Let *pkey* denote a public key for shufflings, $wit^{(\gamma)}$ denote the witness of $\mathcal{M}^{(\gamma)}$ for its shuffling, $icset^{(\gamma)}$ denote input ciphertexts of $\mathcal{M}^{(\gamma)}$, and $ocset^{(\gamma)}$ denote output ciphertexts of $\mathcal{M}^{(\gamma)}$. Each $wit^{(\gamma)}$ consists of a permutation of k elements and k random number to be used for reencryptions. We assume that output ciphertexts $ocset^{(\gamma)}$ of $\mathcal{M}^{(\gamma)}$ are input ciphertexts $icset^{(\gamma+1)}$ of $\mathcal{M}^{(\gamma+1)}$ for $\gamma = 1, \ldots, \lambda - 1$ and that $icset^{(1)}$ are ciphertexts input to the mix-net and $ocset^{(\lambda)}$ are ciphertexts output by the mix-net. We also let $icset^{(\lambda+1)}$ denote $ocset^{(\lambda)}$.

Definition 1. *(Aggregate Shuffle Argument Scheme) An aggregate shuffle argument scheme consists of two algorithms* (Setup, Shuff) *and two interactive protocols* (Ind-Arg, Agg-Arg).

Setup: *A probabilistic setup algorithm that, given a security parameter k, outputs pkey of size k.*
Shuff: *A probabilistic shuffling algorithm for each $\mathcal{M}^{(\gamma)}$ that, given pkey, $wit^{(\gamma)}$ and $icset^{(\gamma)}$, outputs $ocset^{(\gamma)}$.*
Ind-Arg$_{\mathcal{M}^{(\gamma)}, \mathcal{A}}$: *An interactive protocol between each $\mathcal{M}^{(\gamma)}$ and $\mathcal{A}$ in which $\mathcal{M}^{(\gamma)}$ proves to $\mathcal{A}$ the validity of its shuffle. During this protocol, $\mathcal{A}$ is allowed to interact with $\{\mathcal{M}^{(\iota)}\}_{\iota=1,\ldots,\gamma-1,\gamma+1,\ldots,\lambda}$ through other instances of Ind-Arg protocols and with $\mathcal{V}$ through Agg-Arg. $\mathcal{M}^{(\gamma)}$ is given pkey, $wit^{(\gamma)}$, $icset^{(\gamma)}$, and $ocset^{(\gamma)}$. $\mathcal{A}$ is given pkey, and $\{icset^{(\zeta)}\}_{\zeta=1,\ldots,\lambda+1}$. At the end of the protocol, $\mathcal{A}$ outputs acc or rej.*
    Ind-Arg$_{\mathcal{M}^{(\gamma)}, \mathcal{A}}$
Agg-Arg$_{\mathcal{A}, \mathcal{V}}$: *An interactive aggregation protocol between $\mathcal{A}$ and $\mathcal{V}$ in which $\mathcal{A}$ proves to $\mathcal{V}$ the validity of the output of a mix-net. During this protocol, $\mathcal{A}$ is allowed to interact with $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\ldots,\lambda}$ through Ind-Arg protocols. $\mathcal{A}$ is given pkey and $\{icset^{(\zeta)}\}_{\zeta=1,\ldots,\lambda+1}$. $\mathcal{V}$ is given pkey, $icset^{(1)}$, and $ocset^{(\lambda)}$. At the end of the protocol $\mathcal{V}$ outputs acc or rej.*

2.3   Security Requirements of Aggregate Shuffle Argument Scheme

We say an aggregate shuffle argument scheme is secure and efficient if it has the following completeness, soundness, zero-knowledge, and aggregation properties. Our goal is to propose a secure and efficient aggregate shuffle argument scheme. Completeness and soundness are defined as:

Definition 2. *(completeness) If all players are honest,* $\mathrm{ocset}^{(\gamma)}$ *is the shuffling of* $\mathrm{icset}^{(\gamma)}$*,* $\mathcal{A}$ *outputs acc after the end of each* Ind-Arg*, and* $\mathcal{V}$ *outputs acc after the end of each* Agg-Arg*.*

Definition 3. *(Soundness) The scheme is* sound *if the following three statements hold.*

1. *Interactive protocol* $\mathsf{Agg\text{-}Arg}_{\mathcal{A},\mathcal{V}}$ *is an argument of shuffling from* $\mathrm{icset}^{(1)}$ *to* $\mathrm{ocset}^{(\lambda)}$*.*
2. *Each interactive protocol* $\mathsf{Ind\text{-}Arg}_{\mathcal{M}^{(\gamma)},\mathcal{A}}$ *is an argument of shuffling from* $\mathrm{icset}^{(\gamma)}$ *to* $\mathrm{ocset}^{(\gamma)}$*.*
3. *If* $\mathcal{A}$ *accepts all* $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\ldots,\lambda}$ *and* $\mathcal{A}$ *is honest,* $\mathcal{V}$ *accepts* $\mathcal{A}$*.*

Although, what the verifier wants to verify is only the correctness of total mixmet, it is often convenient if the aggregator is able to detect which mixer has cheated. Hence, we included the second and the third properties in the soundness as well as the first one. This is also the reason why defined protocols among $\mathcal{A}$ and $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\ldots,\lambda}$ in divided manner as $\{\mathsf{Ind\text{-}Arg}_{\mathcal{M}^{(\gamma)},\mathcal{A}}\}_{\gamma=1,\ldots,\lambda}$ and $\mathsf{Agg\text{-}Arg}_{\mathcal{A},\mathcal{V}}$.

A permutation that each mixer has chosen must not be leaked through $\mathsf{Ind\text{-}Arg}_{\mathcal{M}^{(\gamma)},\mathcal{A}}$.

Definition 4. *(Zero-knowledge) The scheme is* zero-knowledge *if each interactive protocol* $\mathsf{Ind\text{-}Arg}_{\mathcal{M}^{(\gamma)},\mathcal{A}}$ *is zero-knowledge.*

All of the above properties can be achieved by simply using any zero-knowledge argument for shuffling for multiple times. Hence, efficiency is the one of the most important factors for defining aggregate shuffle argument schemes.

Definition 5. *(Aggregation) We say an aggregate shuffle argument schemes for a mix-net has* aggregation *property if the computational and communication cost for its verifier does not depend on the number of mixers that compose the mix-net.*

3   Basic Idea

Let $q, \mathcal{G}_q$ be as defined in the previous section. In this section, we consider an example of an aggregate argument scheme for simple permutation of $(g_j \in \mathcal{G}_q)_{j \in \mathcal{K}}$. Note that unlike shufflings, this protocol does not hide its permutation. Hence, the example scheme is not secure. Aggregate permutation argument schemes are an analogy of aggregate shuffle argument schemes in which the shuffle protocol Shuff is replaced by the simple permutation protocol Perm. This example scheme is simple but is rich enough to illustrates the essence of our aggregate shuffle argument scheme.

The difference between simple permutation and ElGamal shuffling is existence of masks that hide plaintexts in the latter scheme. These masks are transformed as shufflings are applied to ciphertexts. Once we have an idea of the above example protocol, what is left to do is to find a method for provers to collaboratively and efficiently cancel these transformation of additional masks. With this cancellation, the provers are able to exploit the simple property that the aggregate permutation scheme has. Although the proposed method has no surprising trick, we had to patiently construct a new variant of the scheme in [8] suitable for these process and construct a protocol for cancellation process.

In the next section, we propose a basic aggregate shuffle argument scheme which is secure if players are forced to be honest in a particular circumstance. The underlying shuffle argument is new variant of [8] whose structure manifests itself in the scheme. The data for cancellation required for γ-th mixer, are collaboratively generated by all other mixers and the aggregator. This basic scheme guarantees zero-knowledge property of $\mathsf{Ind\text{-}Arg}_{\mathcal{M}^{(\gamma)},\mathcal{A}}$ only when these collaboration is done honestly.

Finally, the restriction in the basic scheme can be eliminated by requiring each mixer to verify other mixers, which results in the full scheme.

3.1   Permutation Matrix

Let $\delta_{ij} := 1 \bmod q$ if $i = j$ and $\delta_{ij} = 0$ otherwise. Let $\delta_{ijg} := \delta_{ij}\delta_{jg}$. We define permutation matrices over $\mathbb{Z}/q\mathbb{Z}$ as follows:

Definition 6. *Let q be a prime. A matrix $(\phi_{ij})_{(i,j)\in\mathcal{K}^2}$ is a permutation matrix over $\mathbb{Z}/q\mathbb{Z}$ if and only if it satisfies*

$$\phi_{ij} = 1 \bmod q \quad if \ \phi(i) = j, \quad \phi_{ij} = 0 \bmod q \quad otherwise$$

for any permutation function $\phi : \mathcal{K} \to \mathcal{K}$.

The following Theorem holds with respect to permutation matrices. Proofs for them are found in [7].

Theorem 1. (Theorem 2 in [7]) *For prime $q \bmod 3 = 2$, a matrix $(\phi_{ij})_{(i,j)\in\mathcal{K}^2}$ is a permutation matrix over $\mathbb{Z}/q\mathbb{Z}$ if and only if the equation $\sum_{h=1}^{k} \phi_{hi}\phi_{hj}\phi_{hf} = \delta_{ijf} \bmod q$ holds for all $i, j,$ and f.*

3.2   Aggregate Permutation Argument Scheme

The following example scheme is an aggregate permutation argument scheme.

Setup: *pkey* is g and q such that $q \bmod 3 = 2$.
Perm: *pkey* and $icset^{(\gamma)} = (g_{\gamma,i} \in \mathcal{G}_q)_{i\in\mathcal{K}}$, $wit^{(\gamma)} = \{(\phi_{\gamma,ij})_{(i,j)\in\mathcal{K}^2}\}$ is given to
   $\mathcal{M}^{(\gamma)}$. We assume that each element of $icset^{(\gamma)}$ is randomly generated so that
   it is difficult for $\mathcal{M}^{(\gamma)}$ to compute $\{a_i \in \mathbb{Z}/q\mathbb{Z}\}_{i\in\mathcal{K}}$ such that $\prod_{i=1}^{k} g_{\gamma,i}{}^{a_i} = 0$.
   Then, $\mathcal{M}^{(\gamma)}$ outputs

$$ocset^{(\gamma)} = (g_{\gamma+1,i} = \prod_{j\in\mathcal{K}} g_{\gamma,j}{}^{\phi_{\gamma,ij}})_{i=1,\ldots,k}.$$

Note that $ocset^{(\gamma)}$ is a simple permutation of $icset^{(\gamma)}$.

Ind-Arg: $\mathcal{A}$ sends $(c_{\gamma+1,i} \in \mathbb{Z}/q\mathbb{Z})_{i\in\mathcal{K}}$ to $\mathcal{M}^{(\gamma)}$ and then $\mathcal{M}^{(\gamma)}$ returns

$$(c_j^{(\gamma)})_{j\in\mathcal{K}} = (\sum_{i\in\mathcal{K}1} c_{\gamma+1,i}\phi_{\gamma,ij})_{j\in\mathcal{K}}.$$

Here, $\mathcal{A}$ chooses $(c_{\lambda,j})_{j\in\mathcal{K}}$ as the one given by $\mathcal{V}$ through Agg-Arg. $\mathcal{A}$ accepts $\mathcal{M}^{(\gamma)}$ if the following equations hold:

$$\sum_{j\in\mathcal{K}} c_{\gamma,j}g_{\gamma,j} = \sum_{i\in\mathcal{K}} c_{\gamma+1,i}g_{\gamma+1,i} \quad , \quad \sum_{j\in\mathcal{K}} c_{\gamma,j}{}^3 = \sum_{i\in\mathcal{K}} c_{\gamma+1,i}{}^3$$

Agg-Arg: $\mathcal{V}$ sends randomly chosen $(c_{\lambda+1,j} \in_R \mathbb{Z}/q\mathbb{Z})_{j\in\mathcal{K}}$ to $\mathcal{A}$. Then $\mathcal{A}$ sends $(c_{1,j})_{j\in\mathcal{K}}$ to $\mathcal{V}$. $\mathcal{V}$ accepts if the following equations hold:

$$\sum_{j\in\mathcal{K}} c_{1,j}g_{1,j} = \sum_{i\in\mathcal{K}} c_{\lambda+1,i}g_{\lambda+1,i} \quad , \quad \sum_{j\in\mathcal{K}} c_{1,j}{}^3 = \sum_{i\in\mathcal{K}} c_{\lambda+1,i}{}^3$$

As is in the case of the example scheme that appears in the very beginning of Section 4.1 in [10], Ind-Arg is also an argument of permutation: $(g_{\gamma,j})_{j\in\mathcal{K}} \mapsto (g_{\gamma+1,i})_{i\in\mathcal{K}}$. From the fact that equations

$$\sum_{j\in\mathcal{K}} c_{1,j}g_{1,j} = \sum_{i\in\mathcal{K}} c_{2,i}g_{2,i} = \cdots = \sum_{i\in\mathcal{K}} c_{\lambda+1,i}g_{\lambda+1,i} \tag{2}$$

$$\sum_{j\in\mathcal{K}} c_{1,j}{}^3 = \sum_{i\in\mathcal{K}} c_{2,i}{}^3 = \cdots = \sum_{i\in\mathcal{K}} c_{\lambda+1,i}{}^3 \tag{3}$$

hold, Agg-Arg is also an argument of permutation. Computational cost for $\mathcal{V}$ is equal to that for verification of a single permutation. Note that even from $(c_{\gamma+1,j} \in_R \mathbb{Z}/q\mathbb{Z}\}_{j\in\mathcal{K}}$ and $(c_{\gamma,j} \in_R \mathbb{Z}/q\mathbb{Z})_{j\in\mathcal{K}}$ which is an output of Ind-Arg protocol, permutation ϕ_γ is extractable.

4   Basic Aggregate Shuffle Argument Scheme

In this section, we propose a basic aggregate shuffle argument scheme that partially satisfies the proposed security requirements. That is, Ind-Arg protocol between $\mathcal{M}^{(\gamma)}$ and $\mathcal{A}$ is a zero-knowledge argument only if $\{\mathcal{M}^{(\tau)}\}_{\tau=1,\dots,\gamma-1,\gamma+1,\dots,\lambda}$ and $\mathcal{A}$ are honest.

4.1   Overview of the Strategy for the Basic Scheme

There are mainly three steps in our strategy for converting the example scheme to the basic aggregate shuffle argument scheme. They are (1) replacing each of its arguments of permutation with one of shuffling, (2) modifying each of its arguments to a zero-knowledge protocol, and (3) modifying each of its arguments so that its input for shufflings no longer needs to be randomly chosen.

The schemes in [10,8] are constructed by adopting a similar strategy and we follow this strategy for construction of the basic scheme. Once we replace permutation by shuffling, we can have no simple relations between values $(c_{\gamma,i})_{i \in \mathcal{K}}$, $(c_{\gamma+1,i})_{i \in \mathcal{K}}$, $(g_{\gamma,i})_{i \in \mathcal{K}}$, and $(g_{\gamma+1,i})_{i \in \mathcal{K}}$ unlike those in the example scheme. This is because of the masking values in shuffling are transformed by the shuffling and hide the simple relations.

Therefore, the main problem that we must solve in constructing an aggregate shuffle argument scheme from the schemes in [10,8] is how to compensate for the difference between the values in the example scheme and those in the full scheme so that we can still exploit the simple relations, i.e., Equations (2) and (3). This compensation is done by the value h that appears in our proposed scheme. Although such a compensation is provided for only single shuffling in [8,10], in our scheme, the compensation must be provided for all of the shufflings in a mix-net. Our scheme offers such a compensation by passing the compensatory values from mixer to mixer and modifying them.

Finally, we eliminate the condition that the inputs are required to be chosen randomly. This can be done by preparing randomly chosen elements independent to input ciphertext and simultaneously executing the same shuffling to both of them.

The resulting zero-knowledge argument in which all the collaborated mixers play the role of single prover is not exactly the same as the scheme proposed in [8] but is a new variant. Hence, the security of our scheme can be reduced to that of this protocol. It is easy to see, from the theorems in [8], that this protocol is a zero-knowledge argument, as is that proposed in [8].

In our scheme, mixers and an aggregator exchange ElGamal ciphertexts many times (From Step 2 to Step 8), which makes it appear very complicated. However, this is only a multi-party computation of an encryption of the value h, which is an element of the commitment in a zero-knowledge argument for shuffling. Except for this multi-party computation, each Ind-Arg and Agg-Arg are essentially an simple zero-knowledge argument for shuffling. A shuffling can be represented by a combination of multiplication by a matrix and addition of a vector to input ciphertexts. Hence, for compensation of masks transfered by these operation, we are required to transform these masks in opposite way. Hence, the most of complicated computations are composed of multiplication by a matrices, or by its inverse, and addition of vectors.

The full scheme in which mixers and aggregator may be malicious will be proposed in the next section.

4.2 The Proposed Scheme with Honest Players

Setup: Given a security parameter, Setup outputs $pkey = \mathcal{G}_q, (g, m) \in \mathcal{G}_q^2$ where $q \bmod 3 = 2$ and the size of q is the given security parameter.

Shuff: Let a given $icset^{(\gamma)} = (g_{\gamma,i}, m_{\gamma,i})_{i \in \mathcal{K}}$ be a sequence of k ElGamal ciphertexts.

1. $\mathcal{M}^{(\gamma)}$ randomly chooses a permutation matrix $(\phi_{\gamma,ij})$ and $(\varphi_{\gamma,i} \in \mathbb{Z}/q\mathbb{Z})_{i \in \mathcal{K}}$.
2. $\mathcal{M}^{(\gamma)}$ generates $ocset^{(\gamma)}$, a shuffling of input, as
$$(g_{\gamma+1,i}, m_{\gamma+1,i})_{i \in \mathcal{K}} = (g^{\varphi_{\gamma,i}} \prod_{j \in \mathcal{K}} g_{\gamma,j}{}^{\phi_{\gamma,ij}}, m^{\varphi_{\gamma,i}} \prod_{j \in \mathcal{K}} m_{\gamma,j}{}^{\phi_{\gamma,ij}})_{i \in \mathcal{K}}.$$

Ind-Arg and Agg-Arg:

Since Ind-Arg and Agg-Arg are executed in an interleaving way, we will describe both of them together. In the following procedures, interactions between $\mathcal{M}^{(\gamma)}$ and $\mathcal{A}$ belong to Ind-Arg and interactions between $\mathcal{A}$ and $\mathcal{V}$ belong to Agg-Arg. Suppose that $(f_{1,i}\mathcal{G}_q)_{i \in \mathcal{K}}$ and $f \in \mathcal{G}_q$ are generated from a common reference string or an output of a random oracle so that no one is able to generate non trivial $(a_i \in \mathbb{Z}/q\mathbb{Z})_{i \in \mathcal{K}}$ and $b, c \in \mathbb{Z}/q\mathbb{Z}$ such that $g^c f^b \prod_{i \in \mathcal{K}} f_{1,i}{}^{a_i} = 1$. We assume $\bar{f}_{1,i} = \bar{g}_{1,i} = \bar{m}_{1,i} = 1$ for all $i \in \mathcal{K}$.

1. From $\gamma = 1$ to $\gamma = \lambda$, do the following:
   (a) $\mathcal{M}^{(\gamma)}$ is given $(f_{\gamma,i})_{i \in \mathcal{K}}, \bar{f}_\gamma, \bar{g}_\gamma$, and $\bar{m}_\gamma$.
   (b) $\mathcal{M}^{(\gamma)}$ randomly chooses $(\psi_{\gamma,i} \in_R \mathbb{Z}/q\mathbb{Z})_{i \in \mathcal{K}}$ and $\rho_\gamma \in_R \mathbb{Z}/q\mathbb{Z}$.
   (c) $\mathcal{M}^{(\gamma)}$ generates

$$(f_{\gamma+1,i})_{i \in \mathcal{K}} = (f^{\varphi_{\gamma,i}} \prod_{j \in \mathcal{K}} f_{\gamma,j}{}^{\phi_{\gamma,ij}})_{i \in \mathcal{K}}$$

$$\bar{f}_{\gamma+1} = \bar{f}_\gamma f^{\rho_\gamma} \prod_{i \in \mathcal{K}} f_{\gamma,i}{}^{\psi_{\gamma,i}}$$

$$\bar{g}_{\gamma+1} = \bar{g}_\gamma g^{\rho_\gamma} \prod_{i \in \mathcal{K}} g_{\gamma,i}{}^{\psi_{\gamma,i}}$$

$$\bar{m}_{\gamma+1} = \bar{m}_\gamma m^{\rho_\gamma} \prod_{i \in \mathcal{K}} m_{\gamma,i}{}^{\psi_{\gamma,i}}$$

   (d) $\mathcal{M}^{(\gamma)}$ sends $(f_{\gamma+1,i})_{i \in \mathcal{K}}, \bar{f}_{\gamma+1}, \bar{g}_{\gamma+1}$, and $\bar{m}_{\gamma+1}$ to $\mathcal{A}$ and $\mathcal{A}$ forwards them to $\mathcal{M}^{(\gamma+1)}$. In the case $\gamma = \lambda$, $\mathcal{A}$ forwards them to $\mathcal{V}$.
2. Each of $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\dots,\lambda}$ generates a secret key/public key pair $(x_\gamma \in \mathbb{Z}/q\mathbb{Z}, y_\gamma = g^{x_\gamma})$ of ElGamal encryption and prove the knowledge of x_γ to $\mathcal{A}$. Let y be $\prod_{\gamma=1}^{\lambda} y_\gamma$. All y_γ are sent to $\mathcal{A}$.
3. Let $((c_{1,i}^{(1)}, d_{1,i}^{(1)}) = (1,1))_{i \in \mathcal{K}}$.
   From $\gamma = 1$ to $\gamma = \lambda$, do the following:
   (a) $\mathcal{M}^{(\gamma)}$, given $(c_{\gamma,i}^{(1)}, d_{\gamma,i}^{(1)})_{i \in \mathcal{K}}$, generates $(c_{\gamma+1,i}^{(1)}, d_{\gamma+1,i}^{(1)})_{i \in \mathcal{K}}$ which is reencryptions (by y) of

$$(\prod_{j \in \mathcal{K}} c_{\gamma,j}^{(1) \phi_{\gamma,ji}^{-1}}, g^{\sum_{j \in \mathcal{K}} \psi_{\gamma,j} \phi_{\gamma,ji}^{-1}} \prod_{j \in \mathcal{K}} d_{\gamma,j}^{(1) \phi_{\gamma,ji}^{-1}})_{i \in \mathcal{K}}.$$

   Here, $(\phi_{\gamma,ij}^{-1})$ is the inverse matrix of $(\phi_{\gamma,ij})$.
   (b) $\mathcal{M}^{(\gamma)}$ sends $(c_{\gamma+1,i}^{(1)}, d_{\gamma+1,i}^{(1)})_{i \in \mathcal{K}}$ to $\mathcal{A}$ and $\mathcal{A}$ forwards it to $\mathcal{M}^{(\gamma+1)}$. In the case $\gamma = \lambda$, $\mathcal{A}$ forwards it to $\mathcal{V}$

4. Let $(c^{(2)}_{\lambda+1,i}, d^{(2)}_{\lambda+1,i})_{i\in\mathcal{K}} = (c^{(1)}_{\lambda+1,i}, d^{(1)}_{\lambda+1,i})_{i\in\mathcal{K}}$. From $\gamma = \lambda$ to $\gamma = 1$, do the following:

   (a) $\mathcal{M}^{(\gamma)}$, given $(c^{(2)}_{\gamma+1,i}, d^{(2)}_{\gamma+1,i})_{i\in\mathcal{K}}$, generates $(c^{(2)}_{\gamma,i}, d^{(2)}_{\gamma,i})_{i\in\mathcal{K}}$, which is reencryptions of

   $$\left(\prod_{j\in\mathcal{K}} c^{(2)}_{\gamma+1,j}{}^{\phi_{\gamma,ji}^{-1}}, \prod_{j\in\mathcal{K}} d^{(2)}_{\gamma+1,j}{}^{\phi_{\gamma,ji}^{-1}}\right)_{i\in\mathcal{K}}$$

   (b) $\mathcal{M}^{(\gamma)}$ sends $(c^{(2)}_{\gamma,i}, d^{(2)}_{\gamma,i})_{i\in\mathcal{K}}$ to $\mathcal{A}$, which forwards it to $\mathcal{M}^{(\gamma-1)}$. In the case $\gamma = 1$, $\mathcal{A}$ forwards it to no one.
   Note that each $\mathcal{M}^{(\gamma)}$ obtained an encryption of the following. Here the first and third summations are with respect to sets
   $\{i_\xi \in \mathcal{K}|\xi \in \{\gamma,\ldots,\lambda\}, i_\lambda = j_\lambda, i_{\gamma+1} = i\}$ and $\{j_\xi \in \mathcal{K}|\xi \in \{\lambda,\ldots,\eta\}\}$. (Multiplications and summations of matrices and vectors.)

   $$g^{\sum_{i_\xi}\left(\sum_{\eta=\lambda}^1 \sum_{j_\xi} \psi_{\eta,j_{\eta+1}} \prod_{\xi=\lambda}^\eta \phi_{\xi,j_{\xi+1}j_\xi}^{-1}\right) \prod_{\zeta=\gamma}^\lambda \phi_{\zeta,i_\xi i_{\xi+1}}}$$

5. Let $(c^{(3)}_{1,i}, d^{(3)}_{1,i})_{i\in\mathcal{K}} = (c^{(2)}_{1,i}, d^{(2)}_{1,i})_{i\in\mathcal{K}}$. From $\gamma = 1$ to $\gamma = \lambda$, do the following:

   (a) $\mathcal{M}^{(\gamma)}$, given $(c^{(3)}_{\gamma,i}, d^{(3)}_{\gamma,i})_{i\in\mathcal{K}}$, generates $(c^{(3)}_{\gamma+1,i}, d^{(3)}_{\gamma+1,i})_{i\in\mathcal{K}}$ as a reencryption of

   $$\left(c^{(2)}_{\gamma,i}{}^{\psi_{\gamma,i}} \prod_{j\in\mathcal{K}} c^{(3)}_{\gamma,j}{}^{\phi_{\gamma,ji}}, d^{(2)}_{\gamma,i}{}^{\psi_{\gamma,i}} \prod_{j\in\mathcal{K}} d^{(3)}_{\gamma,j}{}^{\phi_{\gamma,ji}}\right)$$

   (b) $\mathcal{M}^{(\gamma)}$ sends the result to $\mathcal{A}$, which forwards it to $\mathcal{M}^{(\gamma+1)}$. In the case $\gamma = \lambda$, $\mathcal{A}$ forwards it to $\mathcal{V}$

6. Let $(c^{(4)}_{\lambda+1,i}, d^{(4)}_{\lambda+1,i})_{i\in\mathcal{K}} = (c^{(3)}_{\lambda+1,i}, d^{(3)}_{\lambda+1,i})_{i\in\mathcal{K}}$. From $\gamma = \lambda$ to $\gamma = 1$, do the following:

   (a) $\mathcal{M}^{(\gamma)}$, given $(c^{(4)}_{\gamma+1,i}, d^{(4)}_{\gamma+1,i})_{i\in\mathcal{K}}$, generates $(c^{(4)}_{\gamma,i}, d^{(4)}_{\gamma,i})_{i\in\mathcal{K}}$ as a reencryption of

   $$\left(\prod_{j\in\mathcal{K}} c^{(4)}_{\gamma+1,j}{}^{\phi_{\gamma,ji}}, \prod_{j\in\mathcal{K}} d^{(4)}_{\gamma+1,j}{}^{\phi_{\gamma,ji}}\right)_{i\in\mathcal{K}}$$

   (b) $\mathcal{M}^{(\gamma)}$ sends the result to $\mathcal{A}$, which forwards it to $\mathcal{M}^{(\gamma-1)}$. In the case $\gamma = 1$, $\mathcal{A}$ forwards it to no one.
   Note that each $\mathcal{M}^{(\gamma)}$ obtained an encryption of the following. Here the first, third, and 4-th summations are with respect to sets $\{j_\theta \in \mathcal{K}|\theta \in \{\lambda,\ldots,\gamma\}, j_{\gamma+1} = i\}$, $\{i_\eta \in \mathcal{K}|\eta \in \{\gamma,\ldots,\lambda\}, i_\lambda = \theta_\lambda\}$, and $\{i_\eta \in \mathcal{K}|\eta \in \{\xi,\ldots,\lambda\}, i_\lambda = \theta_\lambda\}$

   $$g^{\sum_{j_\theta}\left(\sum_{i_\eta}\sum_{\gamma=\lambda}^1 \psi_{\gamma,i_{\gamma+1}} \prod_{\zeta=\lambda}^\gamma \phi_{\zeta,i_{\zeta+1}i_\zeta}^{-1}\right)\left(\sum_{i_\eta}\sum_{\xi=\lambda}^1 \psi_{\xi,i_{\xi+1}} \prod_{\eta=\lambda}^\xi \phi_{\eta,i_{\eta+1}i_\eta}^{-1}\right) \prod_{\theta=\gamma}^\lambda \phi_{\theta,j_\theta j_{\theta+1}}}$$

7. Let $(c^{(5)}_{1,i}, d^{(5)}_{1,i})_{i\in\mathcal{K}} = (c^{(4)}_{1,i}, d^{(4)}_{1,i})_{i\in\mathcal{K}}$. From $\gamma = 1$ to $\gamma = \lambda$, do the following:

(a) $\mathcal{M}^{(\gamma)}$, given $(c_{\gamma,i}^{(5)}, d_{\gamma,i}^{(5)})_{i\in\mathcal{K}}$, generates $(c_{\gamma,i}^{(5)}, d_{\gamma,i}^{(5)})_{i\in\mathcal{K}}$ as a reencryption of

$$\left(c_{\gamma,i}^{(4)\psi_{\gamma,i}} \prod_{j\in\mathcal{K}} c_{\gamma,j}^{(5)\phi_{\gamma,ji}}, \, d_{\gamma,i}^{(4)\psi_{\gamma,i}} \prod_{j\in\mathcal{K}} d_{\gamma,j}^{(5)\phi_{\gamma,ji}} \right)$$

(b) $\mathcal{M}^{(\gamma)}$ sends the result to $\mathcal{A}$, which forwards it to $\mathcal{M}^{(\gamma+1)}$. In the case $\gamma = \lambda$, $\mathcal{A}$ forwards it to $\mathcal{V}$.

8. Each of $\{\mathcal{M}^{(\gamma)}\}_{i=1,\ldots,\lambda}$ sends to $\mathcal{A}$ $(c_{\gamma,i}^{(7)}, d_{\gamma,i}^{(7)})_{i\in\mathcal{K}}$, $(c_{\gamma,i}^{(8)}, d_{\gamma,i}^{(8)})_{i\in\mathcal{K}}$, and $(c_{\gamma,i}^{(9)}, d_{\gamma,i}^{(9)})_{i\in\mathcal{K}}$ as, respectively, encryptions of the following values:

$$(g^{\psi_{\gamma,i}\psi_{\gamma,i}\psi_{\gamma,i}})_{i\in\mathcal{K}}, \, (g^{3\prod_{j\in\mathcal{K}}(\psi_{\gamma,j}\psi_{\gamma,j})\phi_{\gamma,ji}})_{i\in\mathcal{K}}, \, (g^{3\prod_{j\in\mathcal{K}}\psi_{\gamma,j}\phi_{\gamma,ji}})_{i\in\mathcal{K}}$$

9. $\mathcal{V}$ randomly chooses $(c_{\lambda+1,i} \in \mathbb{Z}/q\mathbb{Z})_{i\in\mathcal{K}}$ and sends it to $\mathcal{A}$, which forwards it to $\mathcal{M}^{(\lambda)}$ (Agg-Arg).

10. Let $(r_{\lambda+1,i})_{i\in\mathcal{K}} = (c_{\lambda+1,i})_{i\in\mathcal{K}}$ and $\bar{r}_{\lambda+1} = 0 \in \mathbb{Z}/q\mathbb{Z}$. From $\gamma = \lambda$ to $\gamma = 1$, do the following:

(a) $\mathcal{M}^{(\gamma)}$, given $(r_{\gamma+1,i})_{i\in\mathcal{K}}$ and $\bar{r}_{\gamma+1}$, generates

$$(r_{\gamma,i})_{i\in\mathcal{K}} = \left(\prod_{j\in\mathcal{K}} r_{\gamma+1,j}\phi_{\gamma,ji} + \psi_{\gamma,i} \right)_{i\in\mathcal{K}}$$

$$\bar{r}_{\gamma} = \prod_{i\in\mathcal{K}} r_{\gamma+1,i}\varphi_{\gamma,i} + \rho_{\gamma} + \bar{r}_{\gamma+1}$$

(b) $\mathcal{M}^{(\gamma)}$ sends the results to $\mathcal{A}$, which forwards it to $\mathcal{M}^{(\gamma-1)}$. In the case $\gamma = 1$, $\mathcal{A}$ forwards it to $\mathcal{V}$.

11. $\mathcal{A}$ computes $(c^{(6)}, d^{(6)})$ as in the following and sends it to all $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\ldots,\lambda}$:

$$\left(\prod_{i\in\mathcal{K}} c_{\lambda+1,i}^{(1)\, c^{\lambda+1,i}} \prod_{i\in\mathcal{K}} c_{\lambda+1,i}^{(3)\, (c^{\lambda+1,i})^2} \prod_{i\in\mathcal{K}} c_{\lambda+1,i}^{(5)}, \prod_{i\in\mathcal{K}} d_{\lambda+1,i}^{(1)\, c^{\lambda+1,i}} \prod_{i\in\mathcal{K}} d_{\lambda+1,i}^{(3)\, (c^{\lambda+1,i})^2} \prod_{i\in\mathcal{K}} d_{\lambda+1,i}^{(5)} \right)_{i\in\mathcal{K}}$$

12. $\mathcal{A}$ and each $\mathcal{M}^{(\gamma)}$ computes $(c_{\gamma}^{(10)}, d_{\gamma}^{(10)})$ as in the following: $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\ldots,\lambda}$:

$$\left(\prod_{i\in\mathcal{K}} c_{\gamma+1,i}^{(7)\, c^{\gamma+1,i}} \prod_{i\in\mathcal{K}} c_{\gamma+1,i}^{(8)\, (c^{\gamma+1,i})^2} \prod_{i\in\mathcal{K}} c_{\gamma+1,i}^{(9)}, \prod_{i\in\mathcal{K}} d_{\gamma+1,i}^{(7)\, c^{\gamma+1,i}} \prod_{i\in\mathcal{K}} d_{\gamma+1,i}^{(8)\, (c^{\gamma+1,i})^2} \prod_{i\in\mathcal{K}} d_{\gamma+1,i}^{(9)} \right)_{i\in\mathcal{K}}$$

Then each $\mathcal{M}^{(\gamma)}$ sends $\mathcal{A}$ the decryption h_{γ} of $(c_{\gamma}^{(10)}, d_{\gamma}^{(10)})$ and proves the validity of its decryption. $\mathcal{A}$ accepts $\mathcal{M}^{(\gamma)}$ if the following equations hold:

$$f^{\bar{r}_{\gamma}-\bar{r}_{\gamma+1}} f_{\gamma}{}^{r_{\gamma}} = (\bar{f}_{\gamma+1}/\bar{f}_{\gamma}) \prod_{i\in\mathcal{K}} f_{\gamma+1,i}{}^{r_{\gamma+1,i}} \tag{4}$$

$$g^{\bar{r}_{\gamma}-\bar{r}_{\gamma+1}} \prod_{i\in\mathcal{K}} g_{\gamma,i}{}^{r_{\gamma,i}} = (\bar{g}_{\gamma+1}/\bar{g}_{\gamma}) \prod_{i\in\mathcal{K}} g_{\gamma+1,i}{}^{r_{\gamma+1,i}} \tag{5}$$

$$m^{\bar{r}_{\gamma}-\bar{r}_{\gamma+1}} \prod_{i\in\mathcal{K}} m_{\gamma,i}{}^{r_{\gamma,i}} = (\bar{m}_{\gamma+1}/\bar{m}_{\gamma}) \prod_{i\in\mathcal{K}} m_{\gamma+1,i}{}^{r_{\gamma+1,i}} \tag{6}$$

$$g^{\sum_{j=1}^{k}((r_{\gamma,j})^3-(r_{\gamma+1,j})^3)} = h^{(\gamma)}. \tag{7}$$

13. Each $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\dots,\lambda}$ sends $\mathcal{A}$ a partial decryption of $(c^{(6)}, d^{(6)})$ with respect to their $\{y^{(\gamma)}\}$. Then, $\mathcal{A}$ generates h which is the decryption of $(c^{(6)}, d^{(6)})$ and sends it to $\mathcal{V}$.
14. $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\dots,\lambda}$ jointly prove the validity of $(c^{(6)}, d^{(6)})$ to $\mathcal{V}$ through $\mathcal{A}$. The detail is omitted but is basic.
15. $\mathcal{V}$ accepts the shuffle if the following equations hold:

$$f^{\bar{r}_1} \prod_{i \in \mathcal{K}} f_{1,i}{}^{r_{1,i}} = \bar{f}_{\lambda+1} \prod_{i \in \mathcal{K}} f_{\lambda+1,i}{}^{c_{\lambda+1,i}}$$

$$g^{\bar{r}_1} \prod_{i \in \mathcal{K}} g_{1,i}{}^{r_{1,i}} = \bar{g}_{\lambda+1} \prod_{i \in \mathcal{K}} g_{\lambda+1,i}{}^{c_{\lambda+1,i}}$$

$$m^{\bar{r}_1} \prod_{i \in \mathcal{K}} m_{1,i}{}^{r_{1,i}} = \bar{m}_{\lambda+1} \prod_{i \in \mathcal{K}} m_{\lambda+1,i}{}^{c_{\lambda+1,i}}$$

$$g^{\sum_{j=1}^{k}((r_{1,j})^3 - (c_{\lambda+1,j})^3)} = h.$$

The security analysis is easy instead of its appearance. All the data communicated for generation of h, which are really complicated, are ElGamal ciphertexts of temporally generated public keys. Hence, zero-knowledge of underlying shuffle argument implies computational zero-knowledge of our basic argument. Therefore, checking the completeness of the our basic protocol is only what we have to do other than proving that the underlying protocol is a zero-knowledge argument for shuffle.

The underlying protocol appears well in interaction between $\mathcal{A}$ and $\mathcal{V}$, i.e., Agg-Arg. Here, our h corresponds to $r_{-2} + r'_{-3} + w$ in [8]. To compute value in h that corresponding to cubic elements (A_{j0}^3) part in w of [8], our mixers must transfer encrypted data back and forth for 3 rounds (Steps 3-8). Steps 9-10 is making challenge well randomized by each mixers. Step 11-12 is final addition in exponent (corresponding to $r_{-2} + r'_{-3} + w$) for generation of h and verification. No terms in our scheme corresponds to $A_{-2i}, A_{-3i}, A_{-4i}$ in [8] but $(c_{\gamma+1,i})_i$ is used to compute our h in Stp 11-12, which is similar to the scheme in [10].

As described above, the underlying shuffle argument is almost the same to that in [8]. Hence, proving that each Ind-Arg between $\mathcal{M}^{(\gamma)}$ and $\mathcal{A}$ is a zero-knowledge argument when $\{\mathcal{M}^{(\tau)}\}_{\tau=1,\dots,\gamma-1,\gamma+1,\dots,\lambda}$ and $\mathcal{A}$ are honest can be done in the same manner as is done in [8,10] with few difference. The details are omitted here.

5 Full Scheme

In the scheme proposed in the previous section, each Ind-Arg between $\mathcal{M}^{(\gamma)}$ and $\mathcal{A}$ is a zero-knowledge argument only if $\{\mathcal{M}^{(\tau)}\}_{\tau=1,\dots,\gamma-1,\gamma+1,\dots,\lambda}$ and $\mathcal{A}$ are honest. This honesty can be guaranteed if the scheme additionally includes the following procedures:

1. Each of $\mathcal{M}^{(\gamma)}$ proves to $\{\mathcal{M}^{(\zeta)}\}_{\zeta=1,\dots,\gamma-1,\gamma+1,\dots,\lambda}$ in zero-knowledge that it correctly generated $(f_{\gamma+1,i})_{i \in \mathcal{K}}$ from $(f_{\gamma,i})_{i \in \mathcal{K}}$. This can be done by using the scheme proposed in [8] or [10].

2. Each of $\mathcal{M}^{(\gamma)}$ verifies that Equations (4)-(7) hold for all $\gamma = 1, \ldots, \lambda$.

This completes the description of the full scheme. Note that the above additional procedures are executed by only $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\ldots,\lambda}$ and $\mathcal{A}$. Hence, the cost of $\mathcal{V}$ does not increase.

In the former procedure, each $\mathcal{M}^{(\gamma)}$ is able to verify that $(f_{\gamma,i})_{i\in\mathcal{K}}$ and $\bar{f}_\gamma$ are generated correctly by $\{\mathcal{M}^{(\zeta)}\}_{\zeta=1,\ldots,\gamma-1}$. Each $\mathcal{M}^{(\gamma)}$ is able to verify that $\{(f_{\zeta,i})_{i\in\mathcal{K}}, (\bar{f}_{\zeta,i})_{i\in\mathcal{K}}\}_{\zeta=\gamma+1,\ldots,\lambda}$ are generated correctly by $\{\mathcal{M}^{(\zeta)}\}_{\zeta=1,\ldots,\lambda}$ from the former procedure. Hence, if $\mathcal{M}^{(\zeta)}$ verifies that if Equations (4)-(7) hold for all $\gamma = \zeta + 1, \ldots, \lambda$ in the latter procedure, $\mathcal{M}^{(\zeta)}$ is able to confirm that $(r_{\zeta+1,i})_{i\in\mathcal{K}}$ and $\bar{r}_{\zeta+1}$ are generated correctly by $\{\mathcal{M}^{(\gamma)}\}_{\gamma=\zeta+1,\ldots,\lambda}$. This can be easily understood from Lemma 3 in [8]. Moreover, since all the $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\ldots,\lambda}$ are required to be honest for $\mathcal{A}$ to accept all Ind-Arg, $\mathcal{V}$ accepts Agg-Arg if $\mathcal{A}$ accepts all Ind-Arg. Therefore,

Theorem 2. *The full scheme satisfies completeness, soundness, and zero-knowledge properties.*

The proof will be given in the full paper.

Theorem 3. *The proposed scheme has an aggregation property.*

The proof is clear from the discussion of the next section.

6   Efficiency

In this section, we estimate the computational cost for each player in our proposed scheme and show that the scheme has an aggregation property. Then we compare the efficiency of our scheme with that of previous schemes.

The most costly computation in our scheme is modular exponentiations in $\mathcal{G}_q$. In comparison, the cost of other computations is negligibly small. Therefore, we estimate the computational cost according to the number of modular exponentiations required in our scheme. We also estimate the amount of data that each player needs to communicate.

We first consider the case when $\mathcal{A}$ can be trusted but $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\ldots,\lambda}$ may be malicious. In this case, $\{\mathcal{M}^{(\gamma)}\}_{\gamma=1,\ldots,\lambda}$ do not verify each other, but $\mathcal{A}$ verifies them. The results of estimation are given in Table 1, where we assumed that $|q|$ is 160 and elements of $\mathcal{G}_q$ can be represented by a 161-bit string (e.g. elliptic curves). For a comparison, an estimated computational and communication complexity of the aggregate shuffle argument scheme of Abe [1] are given in Table 2, and those of a scheme in which the ordinary shuffle argument proposed in [12] is repeatedly used multiple times are given in Table 3.

In the general case when $\mathcal{A}$ cannot be trusted, every $\mathcal{M}^{(\gamma)}$ is required to prove the correctness of $F^{(\gamma)}$ to all $\{\mathcal{M}^{(\zeta)}\}_{\zeta=1,\ldots,\gamma-1,\gamma+1,\ldots,\lambda}$. Here, to avoid the execution of these protocols independently to every other mixer, we assume that all $\{\mathcal{M}^{(\zeta)}\}_{\zeta=1,\ldots,\gamma-1,\gamma+1,\ldots,\lambda}$ collaboratively play the role of a single verifier to $\mathcal{M}^{(\gamma)}$. This is possible by using a multi-party coin-tossing protocol. Therefore,

Table 1. Complexity of our scheme

	$\mathcal{V}$	$\mathcal{M}^{(\gamma)}$	$\mathcal{A}$
# of exponentiations in E	$10k$	$28k$	$13k\lambda$
communication bits	$1,500k$	$5,500k$	$5,500k\lambda$

Table 2. Complexity of the scheme in [1]

	$\mathcal{V}$	$\mathcal{M}^{(\gamma)}$	$\mathcal{A}$
# of exponentiations in E	$640k$	$320k$	$640k\lambda$
communication bits	$750,000k$	$1,500,000k$	$1,500,000k\lambda$

Table 3. Complexity of a scheme that uses [12]

	$\mathcal{V}$	$\mathcal{M}^{(\gamma)}$	$\mathcal{A}$
# of exponentiations in E	$6k\lambda$	$6k$	$6k\lambda$
communication bits	$800k\lambda$	$800k$	$800k\lambda$

in the general case when $\mathcal{A}$ cannot be trusted, the computational cost that each $\mathcal{M}^{(\gamma)}$ additionally requires is that of $\mathcal{A}$. The operation of this scheme would be more practical if each mixer published Fiat-Shamir transformation of their proofs instead of interactively proving their correctness to the single verifier that is formed by the collaboration of all other mixers. This would enable all mixers to independently verify the correctness of the prover. Now we have proved Theorem 3.

7   Conclusion

We proposed a new aggregate shuffle argument scheme which is much more efficient that that proposed by Abe in [1]. The computational cost required for the verifier in our scheme is less than 1/60 of that in Abe's scheme. This is mainly because our scheme is based on the scheme in [8] while Abe's scheme is based on the scheme in [22]. Our scheme is particularly effective for large-scale voting such as in a national election.

References

1. Abe, M.: Universally Verifiable Mix-net with Verification Work Independent of the Number of Mix-servers. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 437–447. Springer, Heidelberg (1998)
2. Abe, M.: Mix-Networks on Permutation Networks. In: Lam, K.-Y., Okamoto, E., Xing, C. (eds.) ASIACRYPT 1999. LNCS, vol. 1716, pp. 258–273. Springer, Heidelberg (1999)

3. Brands, S.: An Efficient Off-line Electronic Cash System Based On The Representation Problem, CWI Technical Report CS-R9323 (1993)
4. Chaum, D.: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Communications of the ACM 24(2), 84–88 (1981)
5. Wikström, D.: A Sender Verifiable Mix-Net and a New Proof of a Shuffle. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, Springer, Heidelberg (2005)
6. Wikström, D.: A Universally Composable Mix-Net. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 317–335. Springer, Heidelberg (2004)
7. Furukawa, J.: Efficient, Verifiable Shuffle Decryption and Its Requirement of Unlinkability. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 319–332. Springer, Heidelberg (2004)
8. Furukawa, J.: Efficient and Verifiable Shuffling and Shuffle-Decryption. IEICE Trans. Fundamentals E88-A(1), 172–188 (2005)
9. Furukawa, J., Mori, K., Obana, S., Sako, K.: An Implementation of a Universally Verifiable Electronic Voting Protocol based on Shuffling. In: Blaze, M. (ed.) FC 2002. LNCS, vol. 2357, Springer, Heidelberg (2003)
10. Furukawa, J., Sako, K.: An Efficient Protocol for Proving a Shuffle. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 368–387. Springer, Heidelberg (2001)
11. Golle, P., Zhong, S., Boneh, D., Jakobsson, M., Juels, A.: Optimistic mixing for exit-polls. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 451–465. Springer, Heidelberg (2002)
12. Groth, J.: A Verifiable Secret Shuffle of Holomorphic Encryptions. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 145–160. Springer, Heidelberg (2002)
13. Groth, J.: A Verifiable Secret Shuffle of Homomorphic Encryptions. Cryptology ePrint Archive, Report 2005/246
14. Jakobsson, M.: A practical mix. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 448–461. Springer, Heidelberg (1998)
15. Juels, A., Jakobsson, M.: An optimally robust hybrid mix network. In: Proc. of the 20th annual ACM Symposium on Principles of Distributed Computation (2001)
16. Neff, C.A.: A Verifiable Secret Shuffle and its Application to E-Voting. In: ACM-CCS 2001, pp. 116–125 (2001)
17. Nguyen, L., Safavi-Naini, R., Kurosawa, K.: Verifiable Shuffles: A Formal Model and a Paillier-Based Efficient Construction with Provable Security. In: Jakobsson, M., Yung, M., Zhou, J. (eds.) ACNS 2004. LNCS, vol. 3089, pp. 61–75. Springer, Heidelberg (2004)
18. Ohkubo, M., Abe, M.: A length-invariant hybrid mix. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 178–191. Springer, Heidelberg (2000)
19. Ogata, W., Kurosawa, K., Sako, K., Takatani, K.: Fault tolerant anonymous channel. In: Han, Y., Quing, S. (eds.) ICICS 1997. LNCS, vol. 1334, pp. 440–444. Springer, Heidelberg (1997)
20. Park, C., Itoh, K., Kurosawa, K.: Efficient Anonymous Channel and All/Nothing Election Protocol. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 248–259. Springer, Heidelberg (1994)
21. Peng, K., Boyd, C., Dawson, E.: Simple and Efficient Shuffling with Provable Correctness and ZK Privacy. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 188–204. Springer, Heidelberg (2005)
22. Sako, K., Kilian, J.: Receipt-free mix-type voting protocol –A practical solution to the implementation of voting booth. In: Guillou, L.C., Quisquater, J.-J. (eds.) EUROCRYPT 1995. LNCS, vol. 921, pp. 393–403. Springer, Heidelberg (1995)

Usable Security Workshop

Preface
Usable Security 2007

Usable Security (USEC 2007) was held on February 15 and 16, 2007. The workshop was co-located with the 11th International Conference on Financial Cryptography in beautiful Trinidad and Tobago.

USEC 2007 brought together an interdisciplinary group of researchers and practitioners to discuss one of the most challenging problems in designing secure systems: the human factor. Usability problems have been at the root of many widely reported security failures in high-stake financial, commercial and voting applications. This workshop sought to deepen our understanding of users' capabilities and motivations in performing security tasks.

The program featured two full paper sessions. We began the first session by discussing browser enhancements designed to resist phishing, man-in-middle and other site-impersonation attacks. Jackson et al. presented an empirical evaluation of an interface for extended validation certificates and their effectiveness in the face of phishing attacks. Masone et al. presented a prototype of a new approach to cookie management and secure client authentication. In the next session, Uzun et al. presented a comparative analysis of techniques for creating security associations between end-user devices. Kuo et al. followed up with a related presentation analyzing secure pairing schemes for Bluetooth and Wi-Fi-enabled devices. Finally, Grossklags et al. presented an empirical evaluation of the effectiveness of security notices, warning dialogs and End User License Agreements.

The program also included two panel sessions – Ross Anderson moderated a panel on "The Future of Phishing," and Raquel Hill moderated a panel entitled "Building Trusted Systems: Does Trusted Computing Enable Trusted Systems?". In our most controversial session, Williams et al. presented a systems demonstration of Prime III, a voting scheme that claims to be both usable and secure. The program concluded with Work in Progress papers, on topics ranging from trust indicators and risk communication, to the design of pervasive computing environments and secure video conference systems.

In the closing session, the attendees and organizers concluded that USEC 2007 made an important contribution to the emerging literature on security usability. We raised many more questions than we answered, and we look forward to the next workshop, where we will continue to solicit research and discussions on this important topic.

I would like to thank the International Financial Cryptography Association for launching USEC and for hosting the workshop alongside FC 2007. Our General Chair, Stuart Schechter, deserves special thanks for all of his efforts in organizing the event. The Program Committee and the external reviewers devoted much time and attention to reviewing submissions and selecting papers. Finally, I would like to acknowledge the Center for Research on Computation and Society

at Harvard University and our Silver Sponsor, CommerceNet, for their generous support.

The pre-proceedings papers and slides from each session are available from the USEC 2007 workshop Web site (http://www.usablesecurity.org/).

June 2007   Rachna Dhamija

Usable Security 2007

February 15–16, 2007
Lowlands, Scarborough, Trinidad and Tobago

Program Chair

Rachna Dhamija, Harvard University, USA

General Chair

Stuart Schechter, MIT Lincoln Laboratory, USA

Program Committee

Ross Anderson	University of Cambridge, UK
Steven Bellovin	Columbia University, USA
Dan Boneh	Stanford University, USA
Simson Garfinkel	Harvard University, USA
Raquel Hill	Indiana University, USA
Jason Hong	Carnegie Mellon University, USA
Burt Kaliski	RSA Security and RSA Laboratories, USA
Robert Miller	Massachusetts Institute of Technology, USA
Andrew Patrick	National Research Council, Canada
Angela Sasse	University College London, UK
Dan Schutzer	Financial Services Technology Consortium, USA
Sean Smith	Dartmouth College, USA
J.D. Tygar	U.C. Berkeley, USA
Paul van Oorschot	Carleton University, Canada
Tara Whalen	Dalhousie University, Canada
Ka-Ping Yee	U.C. Berkeley, USA

Silver Sponsor

CommerceNet

An Evaluation of Extended Validation and Picture-in-Picture Phishing Attacks

Collin Jackson[1], Daniel R. Simon[2], Desney S. Tan[2], and Adam Barth[1]

[1] Stanford University, Stanford, CA
{collinj, abarth}@cs.stanford.edu
[2] Microsoft Research, Redmond, WA
{dansimon, desney}@microsoft.com

Abstract. In this usability study of phishing attacks and browser anti-phishing defenses, 27 users each classified 12 web sites as fraudulent or legitimate. By dividing these users into three groups, our controlled study measured both the effect of *extended validation* certificates that appear only at legitimate sites and the effect of reading a help file about security features in Internet Explorer 7. Across all groups, we found that *picture-in-picture* attacks showing a fake browser window were as effective as the best other phishing technique, the *homograph* attack. Extended validation did not help users identify either attack. Additionally, reading the help file made users more likely to classify both real and fake web sites as legitimate when the phishing warning did not appear.

1 Introduction

Paranoia surrounding fraud remains a barrier to using online commerce for many consumers. The padlock encryption symbol used by browsers to indicate HTTPS encryption is often misunderstood, does not appear on the login pages of many legitimate sites, and does not provide users with a reliable mechanism for distinguishing fraudulent sites from real sites. Attackers have increasingly exploited this weakness with *phishing* attacks, sending email to victims enticing them to visit a fraudulent copy of a web site [1]. Over 26,000 unique phishing attack web sites were reported to the Anti-Phishing Working Group in August 2006 [2]. These attacks have cost banks and card issuers billions of dollars [3].

In response, the certificate authority industry has developed a new technology, tentatively named *extended validation* or *high assurance* certificates [4]. Unlike normal certificates, which indicate only that the owner controls a particular domain name, extended validation certificates also attest to the identity of a legitimate business. Internet Explorer 7 indicates the presence of these certificates by turning the address bar green and providing more information about the certificate owner, as shown in Fig. 1.

Our study measured the effect of this new technology on users determining whether or not a page is legitimate. The participants were divided into three groups: one group was trained in the use of green address bars (indicating extended validation certificates that appear only at legitimate sites), one group saw
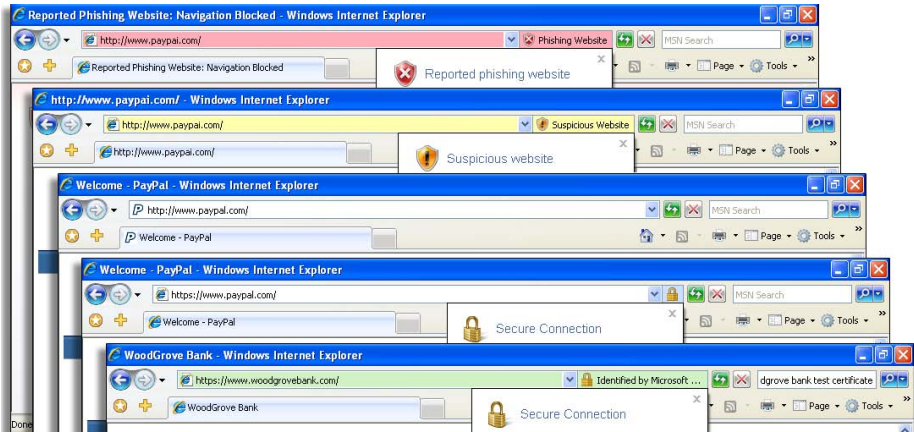
Fig. 1. Phishing, suspicious, HTTP, HTTPS, and extended validation indicators

extended validation indicators but received no training, and a control group was not shown extended validation indicators at all. After familiarizing themselves with two online financial web sites, the participants were shown a series of pages claiming to be those web sites and were asked to classify the pages as legitimate or fraudulent. We compared user responses at the real site, *homograph* [5] sites with similar domain names, and *picture-in-picture* sites that show a fake browser window. Our key findings are:

- Picture-in-picture attacks were as effective as homograph attacks.
- Extended validation did not help users defend against either attack.
- Extended validation did not help untrained users classify a legitimate site.
- Training caused more real and fraudulent sites to be classified as legitimate.

Participants were trained by reading a portion of the Internet Explorer 7 help file that describes both the phishing filter and extended validation features. Our study provides only an upper bound on the efficacy of these indicators because study participants were explicitly instructed to classify sites.

2   Related Work

2.1   Phishing Warnings

One approach to protecting users from phishing attacks is to detect when the browser arrives at an untrustworthy page and warn the user. If the warnings are accurate, and the user heeds them, the phishing page is not able to obtain any information from the user [6]. This approach has been implemented commercially in the form of security toolbars [7,8,9]. These toolbars rely on an up-to-date blacklist, and the composition of the blacklist has a major effect on the accuracy of the toolbar [10]. Although it is difficult for these phishing filters to attain perfect accuracy, they have nonetheless become popular and are

now integrated into most major browsers, including Internet Explorer 7, Mozilla Firefox 2, Netscape 8, and Opera 9.1.

2.2   Positive Trust Indicators

Because it is difficult to build a perfect blacklist of phishing sites, a complementary approach is to show a positive trust indicator, indicating that it is safe for the user to proceed. The lock icon in browsers, which indicates the presence of SSL/TLS encryption, does not ensure the site is trustworthy. Certificate authorities issue domain-validated certificates to anyone who can demonstrate domain ownership by receiving emails addressed to that domain name. The lock icon is frequently ignored by users [11], not present when the login form first appears [12], and displayed on phishing sites that use encryption [13].

Extended validation, which turns the address bar green in Internet Explorer 7, also does not guarantee that the site is "safe" to do business with or that it complies with applicable laws. However, extended validation does provide more accountability for the domain owner, which must be a legally incorporated entity and have a registered office. Unlike regular certificates, extended validation certificates cannot be issued to general partnerships, unincorporated associations, sole proprietorships, and individuals [14].

One disadvantage of positive security indicators is that users have to look for them. In actual usage scenarios, security is rarely a user's primary goal [15]. Anti-phishing tools that provide only neutral or positive information are easier to ignore than phishing warnings [16]. Positive security indicators can also mislead users of a legitimate web site that has been hijacked by an attacker using web vulnerabilities such as cross-site scripting.

2.3   Trusted User Interfaces

Browser security indicators (particularly positive trust indicators) are often prone to user interface spoofing attacks. In an overlapping window environment with a predictable window appearance, an attacker could convince the user that the contents of a web page, under attacker's control, is actually part of the browser [12]. An example picture-in-picture attack with a fake browser window is shown in Fig. 2. User interface spoofing attacks can be foiled using secret images [17] or an unpredictable browser appearance [18,19] that the attacker cannot spoof. These schemes rely on the assumption that the user will not proceed if the trusted image is not present. One of the sites used in our study, Bank of the West, has announced plans to adopt a trusted image scheme in the future.

2.4   Other Authentication Approaches

Most commercial web sites rely on a relatively weak form of password authentication: the browser simply sends a user's plaintext password to a remote web server using SSL/TLS. Unfortunately, the remote web server is not limited to verifying that the password is correct; it can also use the password to log in elsewhere. Although techniques such as SSL/TLS with client-side certificates [20]
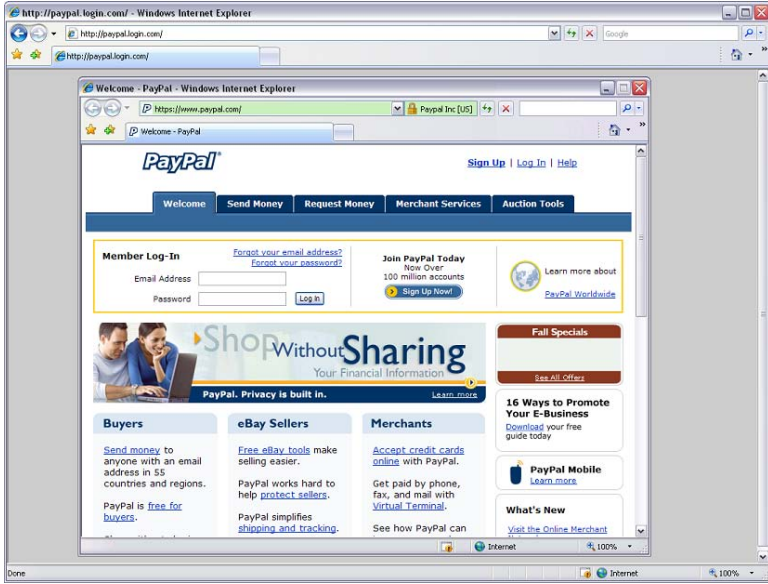
Fig. 2. Picture-in-picture attack. Both the outer (real) window and the inner (fake) window are focused at the same time. The inner window cannot be maximized.

can solve the password theft problem, they are difficult to use and have not yet become widespread. Newly proposed systems make client certificates more usable by employing trusted devices [21] and operating system support [22]. For users who rarely change computers, a long-lasting cookie can be used as a convenient alternative to client certificates, usually as a second factor of authentication [17].

Another approach to the password theft problem is a password manager that automatically generates a unique password for each site, ensuring that the user's password at that site cannot be used anywhere else [23]. These solutions can be vulnerable to picture-in-picture user interface spoofing, so it is important to provide a trusted path to prevent the master password from being compromised [24,25,26]. The trusted path must also be easy to use [27].

Preventing password theft does not protect victims if the phishing attacker does not try to steal the user's password, but instead asks the user directly for other sensitive personal information, such as a social security number.

3   Study Design

Study participants first familiarized themselves with two web sites. One group then received training in the address bar security features, whereas two other groups did not. Participants in all three groups were then asked to classify 12 web sites as legitimate or fraudulent.

3.1 Familiarization

At the beginning of the study, the participants were provided with a computer equipped with the Internet Explorer 7 web browser and instructed to familiarize themselves with two legitimate web sites, PayPal and Bank of the West, presented in a random order. The familiarization step provided participants an opportunity to learn about the look and feel of the real sites before being asked to classify the test sites as legitimate or fraudulent, and, more importantly, gave them an opportunity to learn whether the extended validation security indicator is normally active when using the real site. The participants were randomly divided into three groups who were presented with different experiences:

– **Trained Group.** The trained group was shown extended validation security indicators at each of the real sites. Before the familiarization step, the trained group was also asked to read excerpts from the Internet Explorer help file explaining the security features of the address bar in Internet Explorer 7, including both the phishing filter and extended validation.
– **Untrained Group.** The untrained group was shown extended validation security indicators at each of the real sites, but received no explanation of the meaning of the green address bar.
– **Control Group.** The control group did not see any extended validation indicators during the familiarization step. They received a modified version of the tasks that did not include any extended validation indicators.

Participants in each group were given a fake username and password to use at each site and were instructed to log in when they were ready to continue. The tasks began after the participants had successfully logged in to both sites with the provided username and password.

3.2 Tasks

Once the familiarization step was complete, participants were directed to a web page containing links to 12 web sites in a random order. The link was identified only by a number, preventing the participants from knowing the nature of the site to which they were connecting. They were asked to respond to this prompt:

Imagine you receive an email message that asks you to click on the link shown here. Imagine that you decide to click on the link to see if it is a legitimate web site or a "spoof" (a fraudulent copy of that web site).

The web sites shown were divided into the following categories:
– **Real site.** A site shown in the familiarization step. Sometimes the link would open in a new window, and at other times it would open in the current browser window.
– **Real, but confusing, site.** A deep link into a real site shown in the familiarization step. The page features a warning screen and asks the user for their password, but not a username, as shown in Fig. 3. PayPal regularly sends emails linking to such pages.

Fig. 3. The content of a real, but confusing, PayPal page. Many participants found this page suspicious because it does not ask for a username. The group trained about extended validation was more likely to correctly label this page as legitimate.

- **Homograph attack.** A phishing web page with a domain name that is only a few pixels different from the legitimate site's domain name. The attack sites were `www.bankofthevvest.com` and `www-bankofthewest.com`.
- **Homograph with suspicious page warning.** A homograph attack that triggers a yellow suspicious page warning in Internet Explorer 7. The attack sites were `www.paypai.com` and `www.paypa1.com` (the l is the numeral 1).
- **Picture-in-picture attack.** A phishing web page that shows a fake browser window that appears to be showing the real site.
- **Mismatched picture-in-picture attack.** A picture-in-picture attack that shows a fake browser with a different color scheme than the color scheme of the operating system.
- **IP address blocked by phishing filter.** A web site with no domain name (only a numerical IP address). The browser was immediately navigated away from the page by the Internet Explorer phishing filter, and the address bar turned red. Phishing sites often use IP addresses rather than domain names, but in certain security schemes IP addresses can also be used by legitimate banking sites [28].

3.3 Implementation

During the tasks, the participants used a Windows XP desktop machine in a quiet lab setting. The machine was configured using a `hosts` file containing modified DNS entries for both the spoof and the legitimate domains used in the study pointing to our lab web servers. Additionally, the browser's certificate database had been augmented with our own self-signed root certificates,

enabling us to forge regular and extended validation certificates. Our lab servers were thus able to mount a "man-in-the-middle" attack, intervening between the participant's computer and the real site. The lab web servers acted as reverse proxies, contacting the "real" web site over the Internet on every request and forwarding the response back to the participant's computer with minor changes, simulating the experience of extended validation certificates on the real sites. This configuration also enabled us to construct convincing phishing sites that were exact copies of the real site, differing only in the domain name.

To simulate picture-in-picture attacks, we developed a fake implementation of Internet Explorer in JavaScript, simulating many of the features that a user might use when determining whether a site is legitimate. The simulated browser provided a realistic-looking address bar and a lock icon that displayed fake certificate details when clicked. We provided a fake phishing filter that reported the site as "not a suspicious or reported phishing web site." The fake browser could be navigated, closed, and even dragged, although it could not be dragged outside the confines of the parent page.

3.4   Participant Recruitment and Demographics

Our 27 participants were recruited through the Microsoft Research Usability recruiting service. Two of the participants were non-technical Microsoft employees, and the rest were living in the greater Seattle area, but not affiliated with Microsoft. Potential participants were invited to participate in a study involving "usability of online banking," but were not told ahead of time that the study involved security. For participating, participants received their choice from a list of Microsoft software products.

The participants were 59% male (16) and 41% female (11). None of them were colorblind, and all used Windows as their primary operating system with Internet Explorer as their primary browser. Of the two sites used in the study, none of them had heard of Bank of the West before, whereas 59% (16) had used PayPal. Most, 82% (22), had some experience with online financial services. The average hours of computer usage per week was 36 (min 6, max 80, s.d. 17). Of the participants, 7 (26%) held Masters degrees, 9 (33%) held Bachelors degrees, 9 (33%) reported attending some college, and two held high school diplomas. The average age was 47 (min 23, max 55, s.d. 7.6).

After the tasks, but before the debrief, we asked the participants a few questions to assess their awareness of browser encryption. When shown a picture of an unsecured wireless connection dialog, 88% (23 of 26 respondents) thought that they would be vulnerable to electronic eavesdropping while using the connection for bank transactions, as well as while using the connection to read emails from a web email account. The other 12% (3) thought that they would be secure against electronic eavesdropping while using both types of sites. Because none of the participants provided a different response based on the type of site visited (bank sites use HTTPS, whereas web email sites generally use plain HTTP after the login page), these observations suggest that the participants were not browser encryption experts.
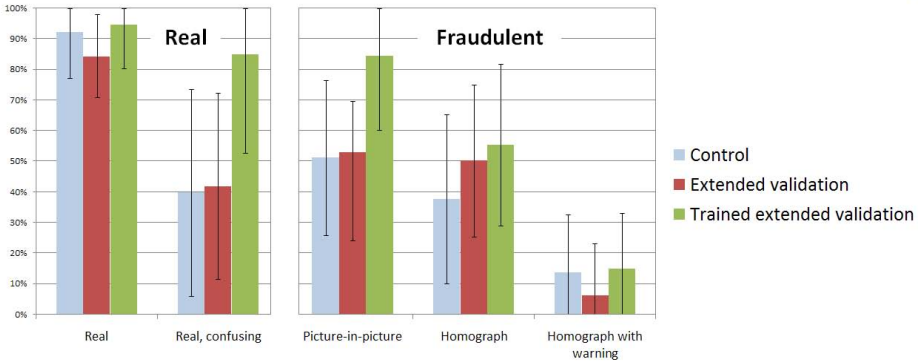
Fig. 4. Percentage of participants who classified sites as legitimate. (95% confidence).

4 Results

A summary of the data collected appears in Fig. 4. Trained participants were more likely to classify the real, confusing site as legitimate, both compared with untrained (p=.031) and with control users (p=.032). The picture-in-picture attacks were more likely to succeed against trained participants than against those in the control group (p=.042), but the observed difference with the untrained group is marginally insignificant (p=.051). Other observed differences in classification were insignificant.

One of the picture-in-picture attacks had silver chrome that matched the operating system theme and two had blue chrome that did not match. Across all groups, we did not observe a significant effect of the chrome color on classification (Friedman's Chi-Square = 0.77, df=1, p=.782). The matched and mismatched picture-in-picture attacks were both classified legitimate by the same percentage of participants (63%).

Across all groups, we did not observe a significant effect of participants having a PayPal account on accurately classifying sites (F=1.12, p=.301) or, specifically, on accurately classifying PayPal sites (F=1.82, p=.191). Because none of the participants had heard of Bank of the West before, we were unable to measure the effect of having an account at Bank of the West on classification choices.

Only three participants categorized all three of the picture-in-picture attacks as fraudulent. Two of these participants tried to use browser features that were not implemented in our JavaScript browser simulation (right clicking and advanced certificate dialog features) and labeled the site as fraudulent because they were not able to get the feature to work. The other participant refused to label any popup window as legitimate.

Across all groups, participants were fooled by homograph pages 11% of the time if a "suspicious page" warning was displayed, compared to 48% of the time if no warning was displayed (t(26)=4.48, p<.001). Although the effect of the warnings is statistically significant, its applicability is limited because the

participants were aware that they needed to classify sites (see discussion in Section 5.5). We also included an IP address that was on the phishing blacklist, which none of the participants classified as legitimate.

When asked afterwards which browser features they had used to categorize web sites as legitimate or fraudulent, 4 of the trained participants indicated that they had used the browser address bar color, one participant in the control group noticed the yellow and red warning colors, and no users in the experimental group indicated that they had used any of the colors.

5   Discussion

5.1   Evaluating Extended Validation

We did not find that extended validation provided a significant advantage in identifying the phishing attacks tested in this study. The untrained extended validation group performed similarly to the control group on all tasks, and none of the untrained extended validation group participants indicated that they had used the address bar color in classifying sites. Extended validation could become more effective over time as it is adopted by more financial web sites and public awareness grows, but at the time of our study (September 2006) we did not observe that it had a significant effect on user behavior.

5.2   Documentation

The trained group was more likely to classify both real and spoof sites as legitimate. This effect can be explained because the portion of the Internet Explorer 7 help files used as the training document included a description of extended validation as well as phishing warnings. Several participants in the trained group focused on the phishing warnings description, expecting that every phishing page would show a warning. This expectation caused them to ignore the lack of an extended validation indicator at some homograph and picture-in-picture pages, and it helped them accurately classify the real (but confusing) sites as legitimate. These findings suggest that browser documentation should be carefully designed not to give the impression that the phishing filter is 100% accurate. In order to isolate the effect of training on extended validation, we plan to limit the training to extended validation for a subsequent study in this area.

5.3   Homograph Defenses

Across all groups, the spoof rate for the "bankofthevvest" homograph attack on Internet Explorer 7 with a Windows XP PC was lower than than the 91% observed by Dhamija et. al. [12] using Firefox on a MacOS X laptop. The font used in the address bar for Mac Firefox has no gap between the double "v" characters, rendering the homograph attack very effective. Internet Explorer 7
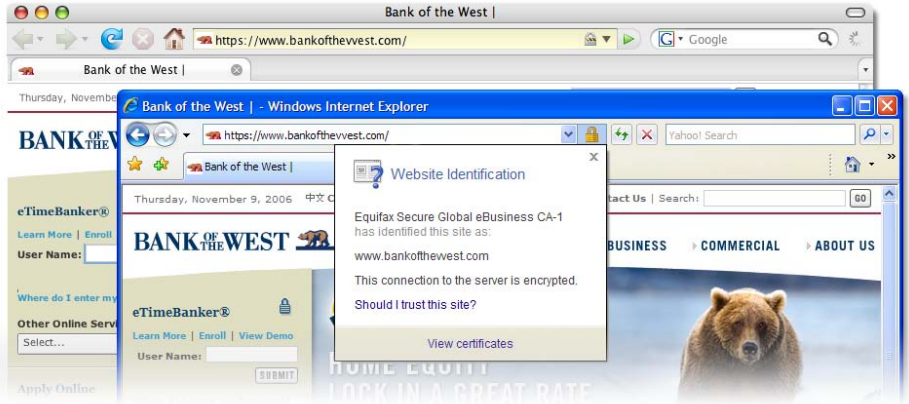
Fig. 5. Comparison of homograph attack on MacOS X with Firefox 2 and Windows XP with Internet Explorer 7

on Windows XP has a gap between "v" letters in the address bar, making the attack easier for users to detect. The certificate details popup, however, uses a font with no gap. One user who looked at the certificate details but not the address bar was fooled by the homograph attack. A comparison of the fonts is shown in Fig. 5.

One proposed defense against homograph attacks is to detect visually deceptive domain names using automated algorithms [29]. It is particularly important to provide this protection in the presence of international Unicode characters, which are hard to distinguish with the naked eye. Internet Explorer 7 disables rendering of international domain names that are not part of the user's configured language.

5.4   Picture-in-Picture Defenses

The general problem of protecting users from spoofed user interfaces in an overlapping window environment is difficult. Although complete solutions do exist [18], they require user interface changes that might seem unnecessary to the user. Without changing the current browser user interface, there are still some visual cues that can be used to identify these attacks.

– **Popups.** External links that open in a new window disable the Back button and can often be perceived as an annoyance [30], yet these links still appear on many major web sites, such as Google's Gmail. Recently, browsers have been discouraging new windows with popup blockers, and browsers such as Firefox and Opera open links in new tabs instead of new windows, when possible. When users restrict their browsing to a single window, the address bar is a more reliable indicator of identity.

– **Mismatched chrome.** One way to expose fake browser windows is to make real browser windows customized for each user, requiring the attacker to guess wildly in order to make a convincing fake [19]. In our study, we observed no significant difference in response when the inner (fake) window had a different chrome color than the outer (real) windows, but the participants were not told to pay attention to the chrome color. This scheme might work better with training. However, most participants found it difficult to notice that the inner window was the wrong color, even during the debrief when it was pointed out to them. Theme differences in Windows applications, such as the Mac-like iTunes interface and the Nullsoft Winamp media player, might have desensitized users to mismatched chrome. Populating the address bar area with a custom icon [25] could be a more effective solution than custom themes.

– **Focus.** In the Windows XP operating system, only one window can be focused at a time. Only the focused window has a bright ("active") title bar. The outer attack page must be focused for the user to enter information into the fake inner window. Thus, a user who sees two focused windows (or a browser window that is focused but appears inactive) can conclude that a fake browser window is present. Unfortunately, this distinction is subtle and hard to remember.

– **Dragging.** A fake browser window cannot be dragged outside of its parent window. Attempting to drag a browser window outside of its parent can thus be used to identify picture-in-picture attacks. However, merely dragging the window around inside its parent does not provide any information about the authenticity of the window.

– **Maximizing.** A fake browser window cannot be maximized, so maximizing a window is an easy way to know that a browser window is not fake. However, windows that cannot be maximized are not a sure sign of fraud as some legitimate sites create popup windows that cannot be maximized.

5.5 Phishing Filter

Some test pages triggered phishing warnings. Those participants who labeled pages with a phishing warning as legitimate did so because they did not notice the warning. However, not all of the fraudulent sites triggered phishing warnings, simulating the reality that phishing warnings appear at some, but not all, phishing sites. By including these warnings, we account for the false sense of security provided by the warnings that might cause users to ignore the extended validation indicator. Our scenario was designed to test the participants' understanding of the browser security indicators, not their awareness of the possibility of an attack. Thus, we explicitly instructed participants to look for fraudulent pages. A study scenario that includes tasks unrelated to security, such as [16], can be used to measure the absolute effectiveness of phishing warnings in real-world scenarios; our results only provide an upper bound on effectiveness.

6 Conclusion

New browser technologies such as extended validation have the potential to defend against fraud by identifying the source of the content displayed on the screen. In this paper, we presented a controlled between-subjects evaluation of the extended validation user interface in Internet Explorer 7. Unfortunately, participants who received no training in browser security features did not notice the extended validation indicator and did not outperform the control group. The participants who were asked to read the Internet Explorer help file were more likely to classify both real and fake sites as legitimate whenever the phishing warning did not appear.

If extended validation becomes widespread, we expect that online criminals will try to mimic its trust indicator, just as they have copied other legitimate financial websites in the past. Like its predecessor, the lock icon, extended validation is vulnerable to picture-in-picture user interface spoofing attacks. We found these attacks to be as effective as homograph attacks, the best known phishing attack. Designing a user interface that resists both homograph and picture-in-picture attacks should be a high priority for designers of future browsers.

References

1. Felten, E.W., Balfanz, D., Dean, D., Wallach, D.S.: Web Spoofing: An Internet Con Game. In: 20th National Information Systems Security Conference (October 1997)
2. Anti-phishing working group: http://www.antiphishing.org
3. Loftesness, S.: Responding to "Phishing" Attacks" (2004), http://www.glenbrook.com/opinions/phishing.htm
4. Franco, R.: Better Website Identification and Extended Validation Certificates in IE and Other Browsers, IEBlog (November 2005)
5. Gabrilovich, E., Gontmakher, A.: The Homograph Attack. The Homograph Attack 45(2) (2002)
6. Chou, N., Ledesma, R., Teraguchi, Y., Mitchell, J.: Client-side defense against web-based identity theft. In: NDSS. Proceedings of Network and Distributed Systems Security (2004)
7. Google, Inc.: Google safe browsing for firefox (2006), http://www.google.com/tools/firefox/safebrowsing/
8. Netcraft: Netcraft Anti-Phishing Toolbar (2006), http://toolbar.netcraft.com/
9. GeoTrust, Inc.: TrustWatch Toolbar (2006), http://toolbar.trustwatch.com/
10. Zhang, Y., Egelman, S., Cranor, L., Hong, J.: Phinding Phish: Evaluating Anti-Phishing Tools. In: NDSS. Proceedings of the 14th Annual Network and Distributed System Security Symposium (2007)
11. Whalen, T., Inkpen, K.M.: Gathering evidence: Use of visual security cues in web browsers. In: GI 2005. Proceedings of the 2005 conference on Graphics interface, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, pp. 137–144. Canadian Human-Computer Communications Society (2005)
12. Dhamija, R., Tygar, J., Hearst, M.: Why Phishing Works. In: Proc. CHI. (2006)

13. Netcraft: Cardholders targetted by Phishing attack using visa-secure.com (October 2004), http://news.netcraft.com/
14. Inc., V.: VeriSign Certification Practice Statement (November 2006), http://www.verisign.com/repository/CPS/VeriSignCPSv3.3.pdf
15. Whitten, A., Tygar, J.: Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In: 8th Usenix Security Symposium, pp. 169–184 (1999)
16. Wu, M., Miller, R., Garfinkel, S.: Do Security Toolbars Actually Prevent Phishing Attacks? In: Proc. CHI. (2006)
17. Passmark: http://www.passmarksecurity.com
18. Ye, Z.E., Smith, S., Anthony, D.: Trusted Paths for Browsers. ACM Transactions on Information and System Security 8(2), 153–186 (2005)
19. Dhamija, R., Tygar, J.: The Battle Against Phishing: Dynamic Security Skins. In: SOUPS 2005. Proceedings of the Symposium on Usable Privacy and Security (2005)
20. Dierks, T., Allen, C.: The TLS Protocol — Version 1.0. IETF RFC 2246 (January 1999)
21. Parno, B., Kuo, C., Perrig, A.: Authentication and Fraud Detection: Phoolproof phishing prevention. In: Di Crescenzo, G., Rubin, A. (eds.) FC 2006. LNCS, vol. 4107, Springer, Heidelberg (2006)
22. Chappell, D.: Introducing Windows CardSpace (2006), http://msdn2.microsoft.com/en-us/library/aa480189.aspx
23. Halderman, J.A., Waters, B., Felten, E.: A convenient method for securely managing passwords. In: WWW 2005. Proceedings of the 14th International World Wide Web Conference (2005)
24. Ross, B., Jackson, C., Miyake, N., Boneh, D., Mitchell, J.: Stronger Password Authentication Using Browser Extensions. In: Proceedings of the 14th Usenix Security Symposium (2005)
25. Yee, K., Sitaker, K.: Passpet: Convenient password management and phishing protection. In: SOUPS 2006. Proceedings of the second symposium on Usable privacy and security, pp. 32–43. ACM Press, New York (2006)
26. Wu, M., Miller, R.C., Little, G.: Web Wallet: Preventing Phishing Attacks by Revealing User Intentions. In: SOUPS 2006. Proceedings of the Symposium on Usable Privacy and Security (2006)
27. Chiasson, S., van Oorschot, P., Biddle, R.: A Usability Study and Critique of Two Password Managers. In: Proc. 15th USENIX Security Symposium (2006)
28. Juels, A., Jakobsson, M., Stamm, S.: Active Cookies for Browser Authentication. In: NDSS. Proceedings of the 14th Annual Network and Distributed System Security Symposium (2007)
29. Fu, A.Y., Deng, X., Wenyin, L., Little, G.: The methodology and an application to fight against unicode attacks. In: SOUPS 2006. Proceedings of the second symposium on Usable privacy and security, pp. 91–101. ACM Press, New York (2006)
30. Nielsen, J.: The top ten web design mistakes of 1999 (May 1999), http://www.useit.com/alertbox/990530.html

WSKE: Web Server Key Enabled Cookies⋆

Chris Masone, Kwang-Hyun Baek, and Sean Smith

Department of Computer Science
Dartmouth College
Hanover, NH USA
{cmasone,jbaek,sws}@cs.dartmouth.edu

Abstract. In this paper, we present the design and prototype of a new approach to cookie management: if a server deposits a cookie only after authenticating itself via the SSL handshake, the browser will return the cookie only to a server that can authenticate itself, via SSL, to the same keypair. This approach can enable usable but secure client authentication. This approach can improve the usability of server authentication by clients. This approach is superior to the prior work on *Active Cookies* in that it defends against both DNS spoofing and IP spoofing—and does not require binding a user's interaction with a server to individual IP addresses.

1 Introduction

In this paper, we present the design and prototype of a new approach to cookie management. We developed this approach to address problems preventing currently usable Web authentication from being secure, and vice-versa.

Initially, we consider the problem of how users can authenticate themselves to servers (*user authentication*). How can end users (or their browsers) be protected from being tricked into releasing their authentication credentials to phishing Web sites? Juels et al [1] recently proposed *Active Cookies* as a solution here. This idea was based on two observations:

- Authentication based on a cookie is more usable and (potentially) more secure than authentication based on user knowledge, since the user need not remember anything and so cannot be tricked into revealing secrets to an adversary.
- In theory, the browser cannot be tricked into revealing a cookie to an adversary, since it is only supposed to send cookies back to the originating domain.

In practice, browsers can be tricked into sending cookies to a spoofed site, via DNS and IP attacks. The bulk of the *Active Cookies* work centers on addressing

---

this work by dispensing with DNS, and instead binding cookies and servers to specific IP addresses. It does not address the issue of IP-based attacks, such as attacks on the *Border Gateway Protocol (BGP)* used by the routers that form the backbone of the Internet to disseminate routing information. Our approach does protect against such attacks. It is important to note that cookie-based authentication schemes are already present in the wild; web sites that offer a "remember me" option at login are one such example.

In addition to user authentication, we also consider the problem of how a user can authenticate servers (*server authentication*). In theory, server-side SSL solves this problem. Server-side SSL PKI provides a flexible, scalable infrastructure for binding server identity to public keys. The SSL protocol provides a way for the user's browser to verify this binding: if a user initiates an SSL request, only the correct server should be able to complete the SSL handshake, since only the correct server should know the private key matching the public key in the presented certificate. In practice, server-side SSL does not work so well, primarily because when a server presents a certificate of questionable validity, the last line of defense is a dialog box that most users will simply click through [2]. Thus, phishers are able to spoof even SSL-protected websites with some measure of success.

In our project, we seek to address both issues by binding cookies to domain names *and public keys*. Once a user (or his browser) has accepted a server's public key, our approach applies *Key-Continuity Management (KCM)* [3] to protect any cookies set by the remote site—including cookie-based authentication credentials. Using KCM in a system means that, once a remote party is associated with a public key, steps are taken to protect the user in the event that an unexpected key is presented at a later date. By applying this methodology to server-side SSL, we reduce to one the number of times the user has to perform the SSL-certificate inspection ceremony. This, we believe, increases SSL PKI usability and helps address the server authentication problem. Perhaps more importantly, using KCM allows our approach to protect the user against IP-spoofing attacks—an improvement over Active Cookies—while also allowing DNS and SSL PKI work as intended.

In this paper, Section 2 explains our problem in more detail. Section 3 discusses our design. Section 4 presents our prototype. Section 5 presents how we evaluated it. Related work is presented in Section 6. Section 7 concludes with some ideas for future work.

2   The Problem

The Web is the primary medium today for electronic service delivery. Even services whose compromise can have serious ramifications for the parties involved—such as banking, high-value commerce, and access to health care data—now use the Web as a portal. Thus, service providers are motivated to try to assure that an alleged end user really is who she purports to be, before providing her with service. The potential value of these transactions has lead to a community of

adversaries who can find profit in subverting this authentication. Securing the process has thus become critical.

However, for a secure electronic service to make business sense, it needs to attract a sufficiently large user base. If authenticating to a service is too difficult or awkward for end users, or too difficult or expensive for the deployer, then it will fail. Users will either be driven away, or driven to find some way to work around the service's security architecture [4,5]. An unusable user authentication strategy can weaken the security of an entire system.

In theory, technologies such as client-side SSL can provide a painlesss way for users to authenticate themselves to servers, without the server learning enough to impersonate that user somewhere else. This enables a user to use one authenticator at many sites, and insulates him (somewhat) from malicious servers. In practice, however, client-side SSL requires a PKI for users at large—which appears to be practical currently only within enterprise populations (such as a corporation or a university). As a result, deployments gravitate toward knowledge-based authentication—userid and password. (However, it's not clear how "usable" passwords really are—humans are not too good at remembering such things.)

Server authentication—how users authenticate servers—is a related issue. The continued problem of Web-based phishing despite a myriad of experimental anti-phishing toolbars shows that the current technology base does not do a very good job. Ordinary users still have trouble determining if the server their browser is interacting with is in fact the bona fide representative of the service provider they intended to contact.

Besides the application-level issues (can the user figure out what the browser's UI is trying to tell them?), system designers need also worry about network-level attacks. For example, a web user typically identifies his intended destination server via a host name. The browser and the PC it's running on then use local *Domain Name Service (DNS)* resources to translate the host name to an *IP address*. The browser and its PC then use local routing resources (built up globally via the *Border Gateway Protocol (BGP)*) to determine how to send network communications to the machine with that IP address. These levels of indirection, and the global protocols that support maintenance of this distributed information, are a critical part of what makes the Internet robust and scalable. Unfortunately, these infrastructures are well-known to be vulnerable to attack. Adversaries can corrupt DNS to fool a host into contact the wrong IP address (e.g., [6,7]). Adversaries can also corrupt BGP to fool a host into thinking that an IP address belongs to an adversary's machine (e.g., [8,9]). Spammers are reputed to make use of BGP weaknesses in practice [10,11]. Active Cookies, since cookies are bound to the IP address of the server that sets them, are vulnerable to these kinds of IP-based attacks. Our approach, which relies on public keys and is totally agnostic to IP addresses, is not.

Secure user authentication can enable effective server authentication. A server can echo back some user-specific personal information should she successfully authenticate. However, this breaks down if a phisher can fool a user into disclosing her authenticators.

In theory, one way to address these problems would be to use easy client-side authentication, such as passwords, over server-side SSL. This would require that, each time the user interacts with this server, she correctly interprets the browser's signals regarding whether the server has correctly carried out the handshake, whether its certificate is valid and from a trustworthy source, and whether the certificate indicates the keyholder is in fact the intended service provider. In practice, of course, this is not workable. Users cannot figure this out.

As Section 1 discusses, the recent Active Cookies work takes a different direction. When a user first establishes a channel with the server, the server deposits a cookie that embodies his authentication. However, the server binds that cookie to an IP address, not a host name. Subsequently, the browser will only disclose that cookie to a server that appears to have that IP address. Unfortunately, this approach has problems with security and usability. The approach protects against DNS attacks but is vulnerable to IP-based attacks (such as attacks on BGP). If the initial channel is to be trusted, we need a way to authenticate the server. Subsequent communications need to be encrypted, if an eavesdropper is not to learn the cookie. More critically, the approach dispenses with the flexibility, load balancing and fault tolerance enabled by DNS and multiple IP addresses; it uses IP addresses for authentication, rather than a technology such as SSL PKI that was actually designed for it. On the usability side, Active Cookies would require that web pages which use cookies have addresses with numeric IP addresses in them, as opposed to human-friendly domain names. Phishers often use such web addresses, and security professionals are trying to educate users to be suspicious of them. Requiring legitimate web applications to use numeric IP addresses would seem to be counterproductive. Thus, an ideal solution to the user authentication problem would allow DNS to do its work and map human-friendly domain names to numeric IP addresses behind the scenes.

This leaves us with the challenge: can we do better? Can we develop a usable way for users to authenticate to servers that:

- like Active Cookies, protects against phishers using Web spoofing and DNS attacks;
- unlike Active Cookies, resists BGP attacks;
- unlike Active Cookies, uses DNS, IP and server-side SSL for their intended purposes; and
- unlike passwords with server-side SSL, prevents the user from having to correctly interpret browser SSL signals each time they connect?

3   Design

To address this challenge, we propose *Web Server Key Enabled Cookies (WSKE-Cookies)*. We leave DNS and IP as they are, but apply KCM to server-side SSL PKI, making server authentication easier for users. This, in turn, allows

cookie-based authentication schemes—which are easier to use than passwords—
to be used more safely. We are aware that WSKECookies (pronounced "Whiskey
Cookies") do not address the *registration problem*, that is the process of acquiring
an authentication cookie in the first place. We consider this issue to be out of
scope, and acknowledge that assuming an attacker is not privy to the initial
contact between a user and a web server is accepting a risk, but it is worth
noting that users of Secure Shell (SSH) have been accepting this risk for years.

The attack model against which WSKECookies defends consists of an attacker
acquiring (or generating) an SSL certificate for his web server, and then using
DNS or IP-spoofing attacks to route traffic destined for a target site to that
server. The certificate used by the attacker will either not match the domain
name to which the user is connecting and/or not be from a Certification Au-
thority that the user's browser is configured to trust. In these cases, the browser
asks the user for input about whether to drop the connection or proceed. A sim-
ple solution to the problem of protecting users' authentication cookies would be
to refuse to send cookies over any connection in which errors arise during SSL
session negotiation. One reason web browsers do not currently do this is compat-
ibility; the most recent survey of SSL certificates on the web by Security Space
shows that about 60% would cause warnings upon connection [12]. Our solution
should not "break the web" by rendering web applications on all these servers
unusable, and so we choose not to block cookies by default. Moreover, there is
an attack combining DNS or IP spoofing, redirection, and cleverly crafted SSL
certificates that can connect a user over SSL to a spoofed website without any
warnings at all, provided that the user's initial connection goes to an insecure
site [13,14]. For instance, if the user types www.gmail.com into his browser, he will
be connected, by default, to http://www.gmail.com, which can easily be DNS or
IP spoofed by an attacker with no warnings. At that point, the attacker can
redirect the user to an SSL site that she controls and for which she has a valid
SSL certificate. If she chooses a plausible name for this site, it is likely that the
user will be fooled.

Ideally, we would like to build WSKECookies as a man-in-the middle that
lives at the client end, watches every https connection, remembers the domain
names and public keys of servers that set cookies, and prevents cookies from
being released to domains that cannot prove knowledge of the correct public
key. Web browsers already ensure that cookies only go to the same domain as
the one that set them, so our job then becomes to guarantee that the public key
associated with a domain name does not change between visits.

The first step is to note when cookies are being set and remember the domain
name and public key of the server which set them. This is not difficult given
the architecture of Mozilla Firefox, our development platform. Conceptually,
protecting cookies is not difficult either. Figure 2(a) outlines the architecture of
the relevant portion of Firefox, and notes the ideal area where our code would
hook in. After the browser has readied the outgoing https request (including the
cookie), it initiates an SSL session with the server. At any point between the
arrival of the server's SSL certificate in the browser and the browser's sending
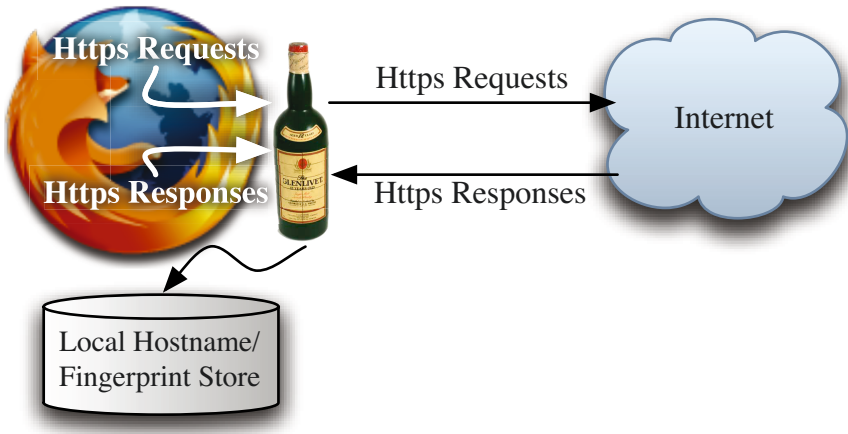
Fig. 1. WSKECookies is implemented as a Firefox extension that imposes itself between all outgoing https requests and all incoming https responses. Once a webserver has set a browser cookie via an https connection, our extension remembers the domain name and public key of that host in a local database. Every time the browser attempts to send an https request containing a cookie to a remote site, our code verifies that the current SSL connection to that site was established using the same key as the first time the user went there. If not, all cookies are removed from the request.

of the request, our code could feasibly jump in and verify that the server's key hasn't changed since the cookie was set. If the key is different, our code would remove the cookie from the request and perhaps provide some feedback to the user.

The use of this framework would be similar to the Active Cookies framework. When a user initially enrolls at the server, she verifies the channel is trusted, and the server deposits a cookie that enables her authentication. The server designs their site to echo some type of personal identifier back to the user upon successful authentication. On subsequent interactions, the server regards presentation of this cookie as proof that it's that user; that user regards presentation of this information as proof that it's that server.

Unlike Active Cookies, in our framework, the user would explicitly use server-side SSL to authenticate the initial channel.

4   Prototype

Active Cookies does not require modifying the browser. Our approach does. So, we felt that it was necessary to build a proof-of-concept to demonstrate the idea. We chose the Mozilla Firefox platform (as mentioned above) because of its status as a mature but open-source framework [15,16]. Unfortunately, the realities of the Firefox architecture provided some hurdles.

Firefox provides three notable programmer's hooks during the process of sending an https request and handling the associated response:
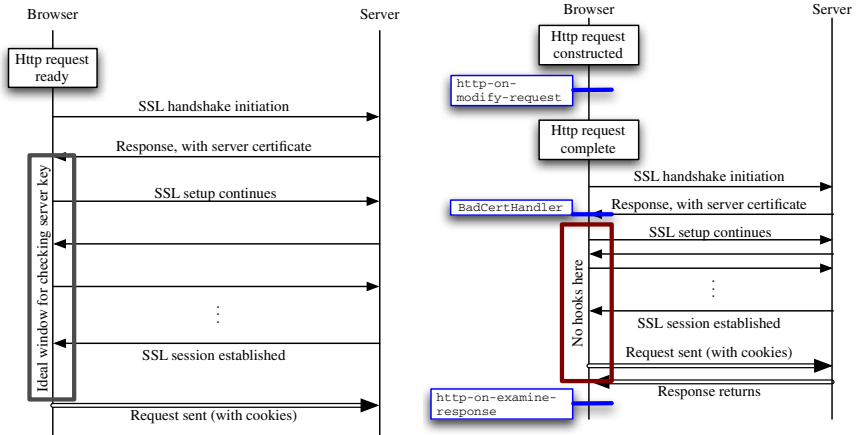
Fig. 2. (a) A ladder diagram of the relevant interactions in Firefox. The ideal period of time for our code to take effect is noted. (b) A ladder diagram of Firefox's preparation of an outgoing https request and the handling of the associates response. The three most useful programmer's hooks that are available to us during this process are also noted here.

- the `http-on-modify-request` event,
- the `http-on-examine-response` event, and
- the `BadCertHandler` object.

(See Figure 2(b)). The first two are similar to signals that can be caught and acted upon, while the last is an object that provides event handler functions which are called in response to various kinds of "bad" server certificates that may be encountered during SSL negotiation. The `BadCertHandler` object's event handling functions are not provided access to the http request, and thus cannot alter it to prevent cookies from being leaked. Thus, we are left with the two events.

The `http-on-examine-response` event fires *after* the SSL session is established (it has to be, as the request and response both had to travel over the secured channel). The remote server's certificate is therefore available in the browser. Thus, this event provides an easy way for WSKECookies to note the initial setting of a cookie by a remote server via https, and also to remember its domain name and public key for future reference. This situation is shown in Figure 3(a). The browser does not yet have any cookies set for the domain it is about to access, so no action must be taken on the outgoing request. When the response comes back, our code is notified and can cull through the response's headers for the domain name, access the server certificate used to set up the secured channel, pair this information up and store it to disk for later usage. Our implementation currently uses an XML-based flat-file database [17,18].
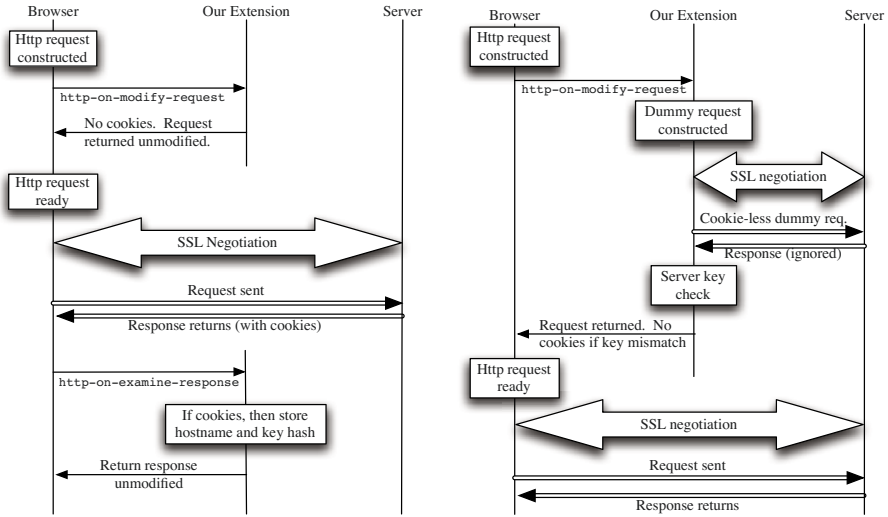
Fig. 3. (a) A ladder diagram of the browser's first interaction with a server that sets cookies via https. Since no cookies for this domain yet exist, our extension code does not need to worry about any being sent out. When a cookie is set, our extension logs the domain name and its public key fingerprint to an XML database on the local disk for later usage. (b) A ladder diagram of the browser connecting to a domain for which is has cookies it is willing to give up over https. The `http-on-modify-request` event fires, giving control to our code. WSKECookies builds a dummy request to send to the server in question, making sure no cookies are leaked. The response from this request is ignored, but the certificate is harvested and used to perform a check on the key. If it matches the key used when the cookies were set initially, our code leaves the initial request alone and allows the browser to go ahead and send it. If not, our code removes the cookies from the initial request and allows the browser to send only this modified version to the server.

As shown in Figure 2(b), the `http-on-modify-request` event fires *before* an SSL session is established with the server being accessed. This means that, at the time our code is given control, we cannot access the server's certificate, because *the browser does not know it yet*. This is not necessary behavior; rather, it is simply the order in which Firefox chooses to do things. Our current proof-of-concept works around this issue by creating a dummy request and sending that to the same URI that the original request was attempting to reach (Figure 3(b)). This dummy request has no cookies, and the response is ignored. We remember the channel object used by the dummy interaction in a hash table [19], so that we can safely ignore the proper response. The whole point is to force the browser to negotiate an SSL session with the desired server so that its certificate can be harvested. We are aware that this creates a small time-of-check-time-of-use (TOCTOU) vulnerability in our current implementation (if the attacker strikes between the time when the response to the dummy request comes back and

the sending of the original request, we will not notice), but the risk here seems negligible. It may also be possible for an attacker that is aware of WSKE to transparently pass the dummy connection through, and then man-in-the-middle the real connection. In future revisions of our code, we hope to come up with a cleaner way to address this issue, probably by adding hooks into NSS (the code module that carries out the SSL handshake). In our testing, the use of these dummy requests did not affect the functionality of web applications that make use of cookies over secure channels, as long as the responses were not allowed to filter through to the browser.

5   Evaluation

5.1   Attack Resistance

To evaluate WSKECookies against possible attacks, we set up a small testbed that consists of two Apache2 Web servers and a Bind9 DNS server:

- The legitimate web server, *Bob*, holds a valid X.509 certificate that matches its domain name, `www.wske.com`.
- The attacker's web server, *Trudy*, holds a different X.509 certificate that matches the domain name, `www.wske.com`. Trudy's certificate may or may not be signed by a trusted root. We tested both cases.
- The DNS server will be used to create the effect of a DNS spoofing attack. Specifically, we can modify the DNS server so that we can direct the traffic that was meant for `www.wske.com` to either Bob or Trudy.

When Alice, a web client, connects to Bob via https, WSKECookies stores Bob's domain name, `www.wske.com`, and the fingerprint of the public key in Bob's certificate. We simulated an IP address spoofing attack by simply bringing Bob down from the network and have Trudy take over Bob's IP address and domain name. A BGP attack would have a similar effect, from Alice's point of view. Moreover, by changing the domain name `www.wske.com` to map to Trudy's IP address, we simulated a DNS spoofing attack, redirecting all the traffic intended for Bob to Trudy (see Figure 4). In both cases, WSKECookies correctly detects that the public key fingerprint in Trudy's certificate does not match Bob's public key fingerprint.

It is worth noting that cookies can also be accessed through a JavaScript interface. However, our approach is easily adapted to address this threat as well; when JavaScript code on an SSL-protected page attempts to access a protected cookie, WSKECookies could verify that the page in question was loaded from a server that presented the appropriate key.

5.2   Usability

In terms of usably protecting users' authentication cookies, WSKECookies is transparent to the user. As long as they are not being spoofed, users will see
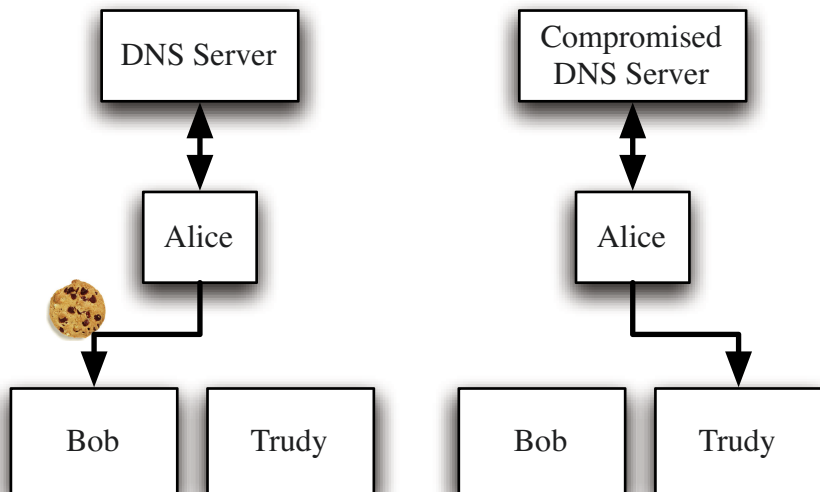
Fig. 4. Our DNS-spoofing attack scenario. (a) Our initial setup, where cookies can and should be released. (b) The "evil" setup, where DNS cannot be trusted and WSKE-Cookies protects the client.

no change in their experience. If they *are* being spoofed, users will be unable to release their authentication credentials. The issue of effectively communicating to the user in the latter case should be explored, and is an interesting area for future work.

5.3   Deployability

In most cases, WSKECookies can be deployed transparently to the providers of web services, as no server-side changes are required. One caveat arises if a site uses a load-balancing solution with its secure web servers in which each machine has a certificate with a different public key. It is difficult for us to verify how common this situation is in the wild, but this risk exists nonetheless. We mention a possible workaround in Section 7. Another concern surrounds server certificate renewal, which commonly happens once every few years. Many websites simply purchase a new server certificate when an old one is about to expire. This certificate usually has a different public key than the old one. Thus, WSKECookies set when the old certificate was in use would all be useless when the new one comes into effect. However, given that users clear their cookies or reinstall their web browser not infrequently, web sites cannot count on long-lived authentication cookies and must structure their web applications accordingly. Thus, users may need to re-register with a web site when the certificate changes. As this happens only once every year or more, it does not seem to be that great a risk. Indeed, if a WKSECookies-like scheme came into common usage, web

sites could avail themselves of certificate renewal, which allows the same key to be rolled into the new server certificate. This is available today, though its frequency of use is unknown.

6   Related Work

Active Cookies [1] was the most directly related work at the time this paper was written, and we have discussed it extensively above. There is also a plethora of work in anti-phishing, using blacklists [20], browser extensions to help users understand security indicators [21,22], or trusted paths from the server to the user [23] to try to arm users against attackers. However, these methods all require some level of diligence and understanding on the part of the user. Our approach imposes less of a burden, and also protects against a range of DNS and IP-based spoofing attacks.

After the acceptance of this paper, we were contacted by the authors of a system called *Locked Cookies* that implements the same conceptual method of cookie protection, but by modifying the Firefox source and compiling a new binary. Similar to WSKECookies, authentication cookies are bound to a source address as well as a public key. This work has since been published as a technical report [24].

Some have suggested that websites encrypt authentication cookies with a secret key before placing them on a user's machine. In fact, this is already common practice. However, if an attacker can steal this encrypted cookie, he does not need to decrypt it all. He simply has to present it to the target website as it is, and he will be successfully authenticated. Thus, solutions such as WSKECookies are still required.

7   Conclusions and Future Work

At the end of Section 2, we laid out a challenge. The WSKECookie approach we then presented meets these criteria. Like Active Cookies, WSKECookies protects against phishers using Web spoofing and DNS attacks. Unlike Active Cookies, WSKECookies also resists BGP (and other IP-related) attacks, and uses DNS, IP and server-side SSL for their intended purposes. Unlike passwords with server-side SSL, WSKECookies prevents the user from having to correctly interpret browser SSL signals each time they connect. As such, its greater usability seems clear. WSKECookies requires no user interaction during the browsing process.

A disadvantage, of course, is that WSKECookies requires changes to the browser software. Our current proof-of-concept makes these changes via the standard extensions framework.

The work reported in this paper leaves several directions for future research.

On a basic implementation level, we want to revise our proof-of-concept code to eliminate the potential vulnerabilities we discussed in Section 4. It would of course be possible to edit the source of Firefox and create a new binary that enables WSKECookies. However, we would prefer to keep our implementation

in the form of an extension, if possible. Despite the fact that we would need to overwrite some object code in the Firefox binary at runtime to clean up our approach, we believe this should be possible [25]. A similar approach would allow us to close the JavaScript hole as well. We may also collaborate with the authors of Locked Cookies to attempt to build their solution as an extension. Regardless, the fact that the existing architecture forced us into our current situation suggests some deeper philosophical questions: namely, if the SSL handshake is intended to provide the client a chance to authenticate the server, why is it that we have to hack into the SSL code to allow the client a chance to examine the certificate information before proceeding with the request? (For that matter, the apparently standard practice of having a browser renegotiate with each https request is surprising.)

On a deeper level, we also want to empirically validate the design assumptions that underlie this framework. Can users be trusted to use the SSL signals to authenticate the server before initial enrollment? Does involving the user exactly once in the SSL ceremony make it more usable than involving the user each time? Do users really find cookie-based authentication more usable than the alternatives?

We also want to explore alternatives to our design. For example, rather than binding the cookie to the server public key, we could bind it to the server's distinguished name and the public key of the trust root; this approach would allow for more of PKI—such as revocation, renewal, and perhaps even proxy certificates—to play a role in the scheme. This approach would continue in our philosophy of, building on, rather than discarding, current network and trust infrastructure.

Code Availability

Our Firefox extension is available for download at http://www.cs.dartmouth.edu/~pkilab/cookies/

References

1. Juels, A., Jakobsson, M., Stamm, S.: Active cookies for browser authentication, http://www.ravenwhite.com/files/activecookies--28_Apr_06.pdf
2. Dhamija, R., Tygar, J.D., Hearst, M.: Why phishing works. In: Proceedings of SIGCHI Conference on Human Factors in Computing Systems, pp. 581–590 (2006)
3. Garfinkel, S.: Design Principles and Patterns for Computer Systems That Are Simultaneously Secure and Usable. PhD thesis, Massachusetts Institute of Technology (2005)
4. Good, N., Dhamija, R., Grossklags, J., Thaw, D., Aronowitz, S., Mulligan, D., Konstan, J.: Stopping spyware at the gate: A user study of notice, privacy and spyware. In: SOUPS. Proceedings of the Symposium on Usable Privacy and Security, pp. 43–52 (July 2005)
5. Yee, K.P.: Guidelines and Strategies for secure Interaction Design. In: Cranor, L.F., Garfinkel, S. (eds.) Security and Usability: Designing Secure Systems That People Can Use. pp. 247–273, O'Reilly, Sebastopol (2005)

6. DNSSEC.NET: DNS Threats and DNS Weaknesses.
   http://www.dnssec.net/dns-threats.php
7. ISC: BIND Vulnerabilities.
   http://www.isc.org/index.pl?/sw/bind/bind-security.php
8. Murphy, S.: BGP Security Vulnerabilities Analysis. Internet Draft draft-murphy-bgp-vuln-01.txt (October 2004)
9. Nordstrom, O., Dovrolis, C.: Beware of BGP Attacks. SIGCOMM Computer Communication Review 34, 1–8 (2004)
10. Housley, R.: Personal commmunication (April 2006)
11. Ramachandran, A., Feamster, N.: Understanding the Network-Level Behavior of Spammers. In: Proceedings of ACM SIGCOMM (September 2006)
12. Space, S.: Secure server survey by security space and E-Soft (September 2006), http://www.securityspace.com/s_survey/sdata/ 200608/certca.html
13. Smith, S.W., Martini, J.C.: The Guidebook for Security Craftsmen (working title). Addison-Wesley, forthcoming book material from AWL 0321434838 (2007)
14. Sirer, G.: Personal Communication (2006)
15. XULPlanet.com: XULPlanet.com (2006), http://www. xulplanet.com
16. Foundation, M.: Mozilla Cross-Reference (2006), http://lxr.mozilla.org/seamonkey/
17. Wilgus, K.: Cookie store firefox 1.5 extension (2006), http://wigginz.com/cookiestore/
18. MonkeeSage: Basic javascript file and directory IO module v0.1 (2006), available at http://kb.mozillazine.org/Io.js
19. Synovic, M.: Dev Notes: Implementing HashTable in JavaScript (2006), http://weblogs.asp.net/ssadasivuni/archive/2003/09/ 17/27902.aspx
20. Netcraft: Netcraft anti-phishing toolbar (2007), http://toolbar.netcraft.com
21. SpoofStick: Spoofstick home (2007), http://www.spoofstick.com/
22. Close, T.: mozdev.org - petname: index. Petname tool (2007), http://petname.mozdev.org
23. Ye, E., Smith, S.: Trusted Paths for Browsers. In: 11th USENIX Security Symposium (2002), http://www.cs.dartmouth.edu/~sws/papers/usenix02.pdf
24. Karlof, C.K., Shankar, U., Tygar, D., Wagner, D.: Locked cookies: Web authentication security against phishing, pharming, and active attacks. Technical Report UCUCB/EECS-2007-25UCB/EECS-2007-25, Electrical Engineering and Computer Sciences University of California at Berkeley (February 2007)
25. Santos, N.J.: Limited delegation (without sharing secrets) in web applications. Technical Report TR2006-574, Dartmouth College (2006)

Usability Analysis of Secure Pairing Methods[*]

Ersin Uzun[1,3], Kristiina Karvonen[2], and N. Asokan[2,3]

[1] University of California, Irvine CA 92697, USA
[2] Helsinki University of Technology, Helsinki, Finland
[3] Nokia Research Center, Helsinki, Finland
`euzun@ics.uci.edu, kk@tml.hut.fi, n.asokan@tkk.fi`

Abstract. Setting up security associations between end-user devices is a challenging task when it needs to be done by ordinary users. The increasing popularity of powerful personal electronics with wireless communication abilities has made the problem more urgent than ever before. During the last few years, several solutions have appeared in the research literature. Several standardization bodies have also been working on improved setup procedures. All these protocols provide certain level of security, but several new questions arise, such as "how to implement this protocol so that it is easy to use?" and "is it still secure when used by a non-technical person?" In this paper, we attempt to answer these questions by carrying out a *comparative* usability evaluation of selected methods to derive some insights into the usability and security of these methods as well as strategies for implementing them.

1 Introduction

The process of setting up a security association between two devices is sometimes referred to as *pairing*. Secure pairing of electronic devices that lack any previous association or infrastructure support, is a challenging problem especially when it needs to be done by ordinary end users without technical expertise. The increasing popularity of powerful mobile electronic devices has made this problem more urgent than ever. Laptops, personal digital assistants (PDAs) and mobile phones all have integrated advanced communication technologies. When the same devices are used for monetary transactions also, the security of these protocols gains a whole new importance. However, no standard user friendly method for establishing secure communication among arbitrary devices exists.

Recently, several different proposed solutions to this secure device pairing problem have appeared in the research literature. Typically these protocols utilize human authenticated and possibly location-limited [8] auxiliary communication channels including visual [12,17], aural [10], short-range wireless channels like Near Field Communications (NFC) [1], and actual physical contact. Each of these proposals makes its own assumptions about the hardware capabilities of devices involved.

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-3-540-77366-5_37

[*] The full version of this paper appears as Nokia Research Center technical report NRC-TR-2007-002.

Several standardization bodies also recognized the seriousness of the problem and have begun work on specifying more usable and more secure procedures for device pairing. Wi-Fi Alliance is working on specifications for *Wi-Fi Protected Setup* [5]. Microsoft has released specifications for *Windows Connect Now-NET* [3], which is closely related to Wi-Fi Protected Setup. Bluetooth Special Interest Group has released a white paper on *Simple Pairing* [2] and is expected to release the specifications soon. The Universal Serial Bus (USB) forum has recently released the specifications for *Wireless USB Association Models* [4] which specifies the procedures for pairing two Wireless USB devices. Unlike the research papers, the standards specifications have to consider devices with a range of hardware capabilities. Consequently, the specifications do not dictate a single pairing method. All of them support the use of at least one type of secure auxiliary channel. For example, Bluetooth Simple Pairing supports the use of NFC and Wireless USB Association Models support the use of USB cables. All the specifications also allow the users themselves to be used as auxiliary channels (See Section 3).

To the best of our knowledge, no comparative usability study of user interaction methods for secure pairing exists. We conducted a comparative usability analysis of different methods in order to identify user preferences, evaluate usability as well as to infer general guidelines for implementing some of the proposed pairing methods.

A single test user cannot effectively compare more than a handful of pairing methods in one test session. Therefore, in our study we concentrated on those user interaction methods implied by the emerging standards specifications. Based on a first round of testing, we refined and narrowed the tested interaction methods further and carried out a second round of testing.

2   Related Work

Although a number of papers have proposed different solutions to the secure device pairing problem, most of them did not report on any significant usability testing. One exception is the Network-in-a-Box project by PARC [6]. They use location limited channels (such as infra-red, physical contact, USB-storage) to provide human verifiable authentication of devices as a pre-requisite to admitting them to a wireless network. Their user testing was to compare the usability of the proposed approach with the traditional methods for configuring wireless network clients. In contrast, our objective is to compare the usability of different proposed approaches to one another.

3   Pairing Protocols and User Interaction Methods

Based on the pairing protocols described in the emerging specifications for secure device pairing [2,3,5,4], we initially selected five different user interaction methods to be tested, as described below.

In all the emerging specifications, the typical approach for secure pairing consists of running Diffie-Hellman key agreement protocol over the insecure channel between the devices and then authenticating this key agreement. Authentication is achieved by transferring some information via a secure auxiliary channel. In this paper, we focus on the case where the users themselves constitute the secure auxiliary channel. The auxiliary channel is used chiefly in one of two ways:

A. Transfer short string(s) so that integrity checksums computed independently by either device can be compared.
B. Transfer a short secret passcode so that both devices share the same short secret.

In approach A, both devices execute a *short authenticated string (SAS) protocol*, such as those described in [7,15,21]. Each device then independently computes a short checksum based on its view of the protocol run. The SAS protocols ensure that if there is an *active* man-in-the-middle, the two checksums are likely to be different. Bluetooth Simple Pairing specification [2] and WUSB Association Models specification [4] support this approach to secure device pairing. The former requires 6 digit checksums while the latter requires 2-4 digit checksums. Neither explicitly specifies the user interaction by which the checksums are compared. There are three obvious possibilities for the interaction methods:

1. **Compare-and-Confirm:** Each device shows its checksum on its display. The user is then prompted to compare the displayed strings and indicate, on each device, whether the two strings are the same or not.
2. **Select-and-Confirm:** During standardization discussions, there was some concern that the *Compare-and-Confirm* method might be too easy for the users leading to their answering the prompt without actually doing the comparison. A comparison method that forces the user to pay more attention might be preferable. In the *Select-and-Confirm method*, one device shows the checksum on its display. The other device shows a set of values including its own checksum, as well as some other randomly chosen strings. On the second device, the user is asked to select the entry that matches the string shown on the first device, or indicate a failure if there is no matching value. If the entry chosen by the user matches its own checksum, the second device indicates success. Otherwise it indicates a mismatch. On the first device, the user is prompted whether the second device indicated success or not.
3. **Copy-and-Confirm:** Not all devices have displays. A typical pairing scenario is between a phone/computer and a keyboard. The *Copy-and-Confirm method* is intended to be used in such scenarios. The device with the display shows its checksum and asks the user to type this value into the second device. The second device compares the entered value with its own checksums and indicates success if the values are the same. On the first device the user is prompted whether the second device indicated success or not.

In approach B, both devices execute a *short-secret authentication protocol*. Both WiFi Protected Setup [3] and Bluetooth Simple Pairing [2] take the approach of splitting the shared secret into k ($k > 1$) equal-sized components and

running the MANA III protocol [9] k times where in each round each party demonstrates its knowledge of the kth component. WiFi Protected Setup uses 2 rounds and requires a 4 or 8 digit passkey. Bluetooth Simple Pairing uses 20 rounds and requires a 6 digit passkey. In both cases, the passkey should not be used more than once. Unlike the checksum in approach A, the passkey must be kept secret from attackers until the pairing process has successfully completed. There are two possible user interaction methods:

4. **Copy:** One device chooses a passkey and displays it to the user and the user is asked to type the displayed value into the second device. The devices automatically run shared secret authentication protocol which succeeds or fails depending on the user's ability to copy the passkey correctly into the second device and the presence of an active attacker. Unlike in the *Compare-and-Confirm method*, no further user interaction is needed here.
5. **Choose-and-Enter:** The user is asked to choose a random passkey and enter it into both devices. Then the devices automatically run shared secret authentication protocol which succeeds or fails depending on the user's ability to enter identical values into both devices and the presence of an active attacker.

In all of the above approaches, the likelihood of a successful man-in-the-middle attack is inversely proportional to the size of the set of values the passkey or checksum can take [20]. The only exception is WiFi Protected Setup, where the level of security is inversely proportional to *half* the length of the passkey space. In other words, to achieve a 4-digit level of security in WiFi Protected Setup, 8 digit passkeys need to be used. The security of all of the approaches is predicated on the assumption that the software implementing the pairing procedure on each device has a *trusted path* to the user: approach A requires that the attacker cannot hide or alter the UI (messages and prompts shown to the user) of the pairing procedure on either device; approach B requires further that the attacker cannot read the passcode displayed to the user.

4   The Study

In computer security, even one user error can be too much. In this regard, the principles of usability of security clearly deviate from the general usability principles. Usually, a trial-and-error approach is acceptable for the learning period when taking a new system into use or playing around with the advanced features of, say, an Office application. However, in usability of security this is not possible. The same holds, of course, for other security-critical systems, such as airplane cockpits or management of nuclear power supplies.

In a security-related interaction, we can group user errors into two categories. A *fatal error* results in the violation of a security goal. All other errors are *safe errors*. Although acceptable fatal error rate may change depending on the application, we assume that any non-zero fatal error rate in the sample size of 40 is unacceptable for security applications. With respect to the pairing methods

described in Section 3, we consider the following fatal errors in our study. In approach A, a fatal error occurs when the checksums computed by each device are different, but user input causes one or both devices to conclude that the checksums match. Fatal errors are possible in all three interaction methods of approach A. In approach B, a fatal error occurs if the user chooses an easy-to-guess passkey in the *Choose-and-Enter* method. There is no possibility of a fatal error in the *Copy* method.

This leads to the first two research questions we want to investigate regarding the *security* of the tested methods:

1. Do users accidentally/carelessly make fatal errors in the tested methods?
2. Does Select-and-confirm have a lower fatal error rate than Compare-and-Confirm?

In addition to the security implications of the interaction methods, we also want to find out the *effectiveness* of the methods both quantitatively, and in terms of user perception. This leads to the next two research questions:

3. How do the methods compare in terms of user perception?
4. How do the methods compare in terms of measurable parameters of effectiveness? (time to completion and total user error rate)

4.1   Test Design and Procedure

Introduction of the tests to users: When test users know that they are testing something related to security, their behavior tends to change drastically [14]. In order to keep user behavior realistic, we designed all test material and procedures so that (a) until the end of the test, security-relevance of the procedure is not emphasized, and (b) the feedback on user actions was independent of whether the action constituted a user error or not.

Choice of devices: The test scenario was one user pairing two devices of the same kind. The same user controlling both devices is the most common real-life scenario. But the devices involved are usually not similar. In order to account for this, we used only the most basic user interactions in designing the user interfaces. Similar user interfaces can be implemented in most types of devices.

Test procedure: Users were first given brief introduction to the study. They were then asked to fill out the background questionnaire (Appendix C) to get demographic information and learn about their mobile device usage history. Next, users were given a brief introduction to the devices to show them the basic operations needed during the test, such as how to move the cursor, erasing a character, etc. The tests were then presented to the user sequentially in random order. Finally they filled out the post-test questionnaire (Appendix D). In the post-test questionnaire, users were given screenshots of each tested method for easy reference. They were asked to associate given adjectives (e.g., "easy", "professional" etc.) with the methods, which method they would like for their own device and what they found difficult about the interactions/UIs during the test.

Tests were run in a private room with no disturbance during the whole process. The testing time was around 20 minutes per user including at least 5 minutes of free discussion at the end where they could give us any additional verbal feedback. The testing procedure remained same throughout the study although the tested method variants and test devices changed.

4.2 Test Implementation

To investigate the likelihood of fatal errors in the methods involving comparing checksums, we needed to simulate a man-in-the-middle scenario by having the devices use different checksums. To measure effectiveness parameters, we needed to record the time for completion. Finally, we needed to present the tests in random order to account for learning effects. We designed a software framework that aids in all of the above. The framework sets up a communication channel between the two devices for co-ordination and takes care of logging completion times user actions. It also enables partially automated test planning. All common functionality, such as inter-device communication or logging, is exposed via a simple application programming interface. In effect, the framework allows usability testing of any multi-device distributed application. The test developer needs to implement the graphical user interface, and few service calls which can be invoked by the framework. Further details about this framework can be found in [13].

4.3 Participant Profile

We did two rounds of usability tests with 40 participants in each. Both tests were conducted in university environments in two different countries with a clear majority being U.S. and Finnish citizens. We used similar means of recruitment announcement, such as mailing lists and bulletin boards, to attract similar participant groups in both environments. The distribution of gender, age and education of the test participants are given in Table 1.

Table 1. Participant Profile

	First Group(40 people)	Second group(40 people)
Gender	Male: 60% Female: 40%	Male:70% Female: 30%
Age	18-24: 22% 25-29: 52% 30-34: 15% 35+: 11%	18-24: 20% 25-29: 47% 30-34: 15% 35+: 18%
Education	High School: 13% Bachelor: 30% Graduate Degree: 57%	High School: 32% Bachelor : 28% Graduate Degree: 40%

The groups had other similar characteristics. In both groups, the average computer usage history for participants was around 12 years and the average computer usage was 7 hours per day. All participants in our study had either a PDA or a mobile phone, or both.

4.4   First Round

In this first round of our study, we conducted our usability tests in a university in the United States.

Material. We used iPAQ devices running Windows CE operating system. User interaction consisted of using an on-screen keyboard on a color screen. The Windows CE environment is intended for mobile devices but it provides a windowed GUI environment that is similar to PC and other PDA operating systems.

Tested methods. Each method described in Section 3 was tested. The settings used for each method are described below (Screenshots can be found in appendix A).

1. *Compare-and-Confirm*: We used randomly generated 4-digit numbers to be presented as "checksums". In half the cases, chosen randomly, we showed different values on the two devices. The issuer prompt was "Check if both devices display the same value". Users were given two button choices labeled as YES and NO to give their answers.
2. *Select-and-Confirm*: The first prompt on the first device was "Please select "XXXX" from the list on the other device" followed by the question "Did the other device indicate success?" and YES/NO buttons. The second device simultaneously showed the instruction "Please choose the value other device is displaying" and a list consisting of four 4-digit numbers, including the value shown on the first device. A success or failure pop-up screen appeared depending on whether the user chose the correct value in the list or not.
3. *Copy-and-Confirm*: The first device showed the text "Enter the displayed key to the other device" followed by a 4-digit checksum and the question "Has the other device indicated success?" The second device instructed the user "Please enter the value the other device is displaying" and showed a success or failure pop-up depending on whether the value was copied correctly.
4. *Copy*: We tested two variants: one using 4-digit passcodes and the other using 8-digit passcodes. The first device showed a key and the text "Enter the displayed key to the other device". The second device instructed the user "Please enter the value the other device is displaying".
5. *Choose-and-Enter*: The prompt was "Choose a 4-digit hard to guess number and enter it into both devices".

Results. The quantative data is summarized in Table 2. Participants were asked to associate given adjectives with the methods. The Participant opinions are summarized in Figure 1. The graph shows the percentage of the participants who associated certain adjective with a certain method variant.

   We can make the following observations

- *Copy-and-Confirm* as well as *Copy* with 8 digit passkeys were perceived to be hard to use.
- Fatal error rate was unacceptably high in all the methods except *Copy*.
- *Select-and-Confirm* had a 7.5% lower fatal error rate than *Compare-and-Confirm*.

Table 2. Summary of first round usability tests

Method	Variant	Avg. Comp. Time (sec.)	Fatal Error Rate	Total User Error Rate
Compare-and-Confirm		15.6	20%	20%
Select-and-Confirm		22.5	12.5%	20%
Copy-and-Confirm		27.6	10%	20%
Copy	4-digits	20.8	N/A	7.5%
	8-digits	31.7	N/A	5%
Choose-and-Enter		32.7	>42.5%	45%
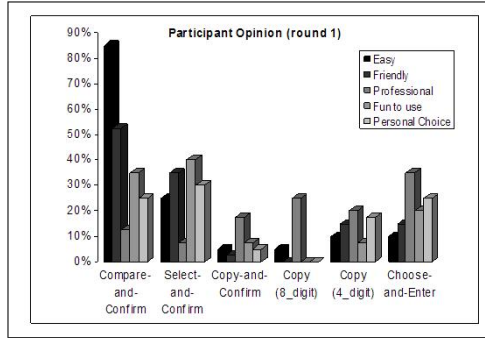
Fig. 1. Summary of participant opinions in the first round

The *Choose-and-Enter* method had an extremely high fatal error rate: 42.5% of the users chose passkeys that were in a small set of predictable sequences we screened for. It also had the longest average completion time. Since there is no way to improve the fatal error rate in this method, we decided to abandon it.

The *Copy-and-Confirm* method had a high fatal error rate, and was not perceived to be easy to use. From user feedback, it was evident that users were confused about having to do two things (type a passkey, and confirm). Therefore, we decided to abandon this method as well. It implied that in situations where *Copy-and-Confirm* would have been applicable, it would be necessary to use the *Copy* method.

The *Compare-and-Confirm* and the *Select-and-Confirm* methods both had unacceptably high fatal error rates. In *Compare-and-Confirm*, all user error was fatal. We decided to experiment further by modifying the UI in these cases.

The *Copy* method was inherently not prone to fatal errors, although users did not perceive it as a user-friendly method.

In the next round, we decided to focus on the methods *Compare-and-Confirm*, *Select-and-Confirm* and *Copy*.

4.5   Round Two

We conducted our second round of tests in a Finnish university. The participant profile was quite similar to our first study as explained in section 4.3.

Material. We used Nokia E60 series mobile phones running Symbian S60 3rd edition. All test material, including questionnaires and user interfaces were available in both English and Finnish. Participants chose their preferred test language.

Tested methods. We implemented the three variants selected at the end of the first round. Based on the first round experience, we made some changes intended to improve usability and security, as described below. All methods are tested with 6-digit numbers, used either as checksum or passcode. We chose this value because it is the longest value mentioned in the standards [2]. Although [5,3] allow 8 digit passcodes, we ruled it out based on the results of the first round, as well as the established cognitive fact that the maximum number of chunks of information that can be kept in working memory is 7 [16]. In the UI, the numeric code was consistently referred to as a PIN, regardless of whether it was used as a passkey or checksum. Screenshots of the implementations can be found in appendix B.

1. *Compare-and-Confirm*: The wording of the question was changed to "Compare the PIN numbers shown on both devices, are they DIFFERENT?" and user was given two choices of SAME and DIFFERENT. The default response key was assigned to the option DIFFERENT, so that accidental or careless user error will no longer be a fatal error (Note also that the default label used exactly the same word as in the question). This was done in order to gain the users' attention. When a difference is suggested, users tend to concentrate more on finding it (e.g., [19]. Further, Hammer et al [11] have shown that (i) people use positive constraints more intuitively, although they fail to use them perfectly and (ii) the use of negative constraints enables a less natural, but potentially more accurate categorization strategy. This meant that in the usual case, the user's thought process has to deal with something akin to double negation: when the number sequences were the same, the response to the prompt is "no", which the user has to mentally map to the key labeled SAME. This design choice could be a potential source of difficulty since it is well known in cognitive psychology that processing of double negation is more complex and thus slower. We tested two variants, one with matching checksums and the other with non-matching checksums.

2. *Select-and-Confirm*: The selection list offered four choices to select from but "No Match" was added as an option to make the action more intuitive when the correct value is not in the list. Design of the selection screen was changed to target more user attention. The first prompt changed to "Please select the PIN below on other device" followed by the checksum in a separate line and the second prompt "Has the other device indicated success after selection?". The pop-up screen showing success or failure was also redesigned to give explicit next action guidance, E.g. "Successful!, please choose YES on the other device to continue". We tested two variants one in which the set on

the second device included the checksum shown on the first device, and the
other in which it did not.

3. *Copy*: Screen text in first device was changed to "Please enter the PIN below
   into the other device" followed by the PIN in a separate line. Second device
   prompt was "Please enter the PIN other device is displaying and press OK
   when you are done".

Results The data collected in this round is summarized in Table 3.

Table 3. Summary of second round usability tests

Method	Variant	Avg. Comp. Time(sec.)		Fatal Error Rate	Total Error Rate
		Match	No match		
Compare-and-Confirm	6-digit & new GUI	16.4	13	0%	2.5%
Select-and-Confirm	6-digit & new GUI	16.4	26.4	5%	7,5%
Copy	6-digit	13	N/A	N/A	2.5%

We also changed some of the adjectives we used in post-test questionnaire
aimed towards getting more precise information while still keeping the gathered
information comparable between rounds. A graph summarizing the user opinion
is in Figure 2.
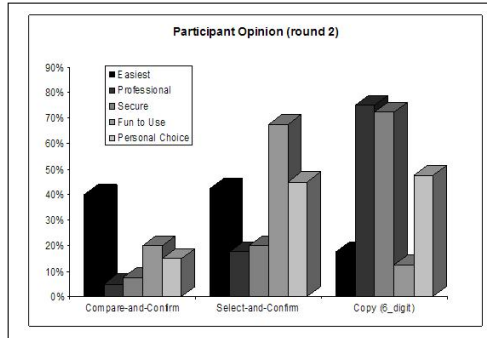
Fig. 2. Summary of participant opinion in second round

We can make the following observations

- *Compare-and-Confirm* and *Select-and-Confirm* are both perceived as easy
  but not professional.
- *Compare-and-Confirm* and *Copy* had no fatal errors, while *Select-and-Confirm*
  still had unacceptable fatal error rate.
- *Copy* is perceived as hard but professional. It was the most preferred personal
  choice as the pairing method users would like to have available on their
  devices.

5   Analysis and Discussions

Strictly speaking, the conclusions drawn from the data collected can be considered only as indicative of the whole user base due to the differences in the test set-up between the rounds and the relatively small number of participants.

The fatal error rate in *Compare-and-Confirm* improved significantly from 20% to 0%. There were four differences between the two test rounds: participant groups, devices, checksum lengths, and the UI design. As discussed in Section 4.3, the profiles of the two participant groups were similar. The user interaction is so simple in *Compare-and-Confirm* that the change in devices cannot account for the improvement. The increase in checksum length is not very likely to have improved the user error rate; although it cannot be ruled out as a factor since users may have been more careful when faced with a harder task. This leaves us to conclude that the changes to the UI design is the likely cause.

Select-and-Confirm had unacceptable fatal error rates in both rounds. The user actions on the two devices need to be followed strictly in the prescribed order: select on the second device, wait for a response, and only then answer the second prompt on the first device. It is difficult to design the UI so that it strongly guides the user to follow this prescribed order and minimizes the likelihood of flouting it.

The *Copy* method has natural resistance against fatal errors as long as the devices are not compromised or the attacker cannot interfere with the display. The completion time and total user error rate were lower in the second round, which is to be expected since typing digits is easier on cell phones than PDAs.

The *Copy-and-Confirm* and *Choose-and-Enter* methods were abandoned after the first round due to their high fatal error rate and negative user perception. We recommend using *Copy* instead of *Copy-and-Confirm* although *Copy* requires keeping the PIN secret. The *Choose-and-Enter* method can also be replaced with *Copy* method in many cases. Users perceived *Compare-and-Confirm* and *Select-and-Confirm* as easy to use, and considered *Copy* difficult. However, they considered *Compare-and-Confirm* and *Select-and-Confirm* to be less secure and less professional than *Copy*. These properties are often found to be interrelated and also desirable by the users for seemingly irrational reasons (see e.g. [18]).

The popularity of *Compare-and-Confirm* was significantly lower in the second round. This is probably due to the increase in the checksum length, as well as due to the UI change. Some users were surprised by the negative question and unexpected labeling of response actions, and expressed that they would have preferred e.g. the usual "Cancel" and "OK" options instead of "SAME" and "DIFFERENT". User perception may be improved by breaking up the checksum into chunks of two or three digits.

Checksums and passkeys used in the pairing methods are very different from traditional PINs: checksums are not secret; passkeys are limited to single-use and need not be remembered. Nevertheless users assume checksums and passkeys are similar to the type of PINs they are already familiar with. They use this assumption as a reference point for their opinions about tested methods. This

had both a negative (PINs are hard to remember) as well as positive (users are familiar with using PINs) bias to the test setting.

Based on these observations we formulate the following guidelines for designing UIs for the tested methods.

– Default user action (e.g., default button) must correspond to the safest choice.
– User actions must be labeled using words that are specific to the task expected from the user. Generic (and familiar) labels like YES/NO, CANCEL/CONTINUE should be avoided. Especially those labels that have direct negative and positive associated meaning should be avoided.
– Multi-step interactions where users can inadvertently and easily change the prescribed order of interactions should be avoided. If such interactions are unavoidable, the UI should make sure that it is difficult to change the prescribed order.

For creating usable procedures with numbers, the cognitive issues involved must be taken into account. For example, checksums and passkeys must not be longer than 7 digits.

Returning to the research questions we started out with in Section 4, we can conclude the following. *Copy* is inherently resistant to fatal errors. Fatal errors in *Compare-and-Confirm* can be avoided by careful design of the UI. *Select-and-Confirm* does not have a lower fatal error rate than *Compare-and-Confirm*. The users clearly differentiated among the methods in terms of ease-of-use and perceived level of security. However the methods tested in the second round were similar in terms of measurable parameters like completion time, fatal and total error rates, and security.

6   Future Work

In this study, we concentrated on obvious interaction models implied by the emerging standards. However, there are other promising methods that either use different auxiliary channels or the human authentication in different means. We are testing handful of these methods using visual, aural, NFC channels and some methods relying on more basic human sensory capabilities.

After the first round, we identified several UI improvements. We made *all of them* for the second round for pragmatic reasons. We are currently doing more controlled, smaller-scale tests to better understand the effects of different UI improvements.

We assume throughout the study that the pairing procedure has a trusted path to the user. This can be implemented, for example, if the control of the display cannot be taken out from the pairing software when it is active. When this is not the case, more attack possibilities exist, such as sending a text message to a cell phone during the pairing procedure and hoping that the user will follow instructions in the message. We plan to include these kinds of attack scenarios in our future work.

References

1. Near field communications forum (2004), `www.nfc-forum.org/`
2. Bluetooth special interest group: simple pairing whitepaper (2006), `http://www.bluetooth.com/Bluetooth/Apply/Technology/Research/Simple_Pairing.htm`
3. Windows connect now-ufd and windows vista specification (2006), `http://www.microsoft.com/whdc/Rally/WCN-UFDVistaspec.mspx`
4. Wireless usb specification: Association models supplement. revision 1.0. USB Implementers Forum (2006), `http://www.usb.org/developers/wusb/`
5. Wi-fi protected setup specification (January 2007), `http://www.wi-fi.org/published_specifications.php`
6. Balfanz, D., Durfee, G., Grinter, R.E., Smetters, D.K., Stewart, P.: Network-in-a-box: how to set up a secure wireless network in under a minute. In: SSYM 2004. Proceedings of the 13th conference on USENIX Security Symposium, Berkeley, CA, USA, pp. 207–222. USENIX Association (2004)
7. Cagalj, M., Capkun, S., Hubaux, J.: Key agreement in peer-to-peer wireless networks. In: Proceedings of the IEEE (Special Issue on Cryptography and Security) (2006)
8. Balfanz, D., Smetters, D.K., Stewart, P., Chi Wong, H.: Talking to strangers: Authentication in ad-hoc wireless networks. In: Symposium on Network and Distributed Systems Security (NDSS 2002) (February 2002)
9. Gehrmann, C., Mitchell, C., Nyberg, K.: Manual authentication for wireless devices. RSA Cryptobytes 7(1), 2937 (2004)
10. Goodrich, M.T., Sirivianos, M., Solis, J., Tsudik, G., Uzun, E.: Loud and clear: Human-verifiable authentication based on audio. In: ICDCS 2006. Proceedings of the 26th IEEE International Conference on Distributed Computing Systems (2006)
11. Hammer, R., Hochstein, S., Weinshall, D.: Category learning from equivalence constraints. In: XXVII Annual Conference of the Cognitive Science Society (CogSci 2005) (July 2005)
12. McCune, J.M., Perrig, A., Reiter, M.K.: Seeing-Is-Believing: Using Camera Phones for Human-Verifiable Authentication. In: 2005 IEEE Symposium on Security and Privacy, pp. 110–124 (2005)
13. Kostiainen, K., Uzun, E., Asokan, N., Ginzboorg, P.: Framework for comparative usability of distributed applications. Technical Report NRC-TR-2007-005, Nokia Research Center (2007)
14. Kuo, C., Perrig, A., Walker, J.: Designing an evaluation method for security user interfaces: Lessons from studying secure wireless network configuration. interactions 13(3), 28–31 (2006)
15. Laur, S., Nyberg, K.: Efficient mutual data authentication using manually authenticated strings. In: Pointcheval, D., Mu, Y., Chen, K. (eds.) CANS 2006. LNCS, vol. 4301, pp. 90–107. Springer, Heidelberg (2006)
16. Miller, G.A.: The magical number seven, plus or minus two: Some limtis on our capacity for processing information. Psycological Review 63, 81–97 (1956)
17. Saxena, N., Ekberg, J.-E., Kostiainen, K., Asokan, N.: Secure Device Pairing based on a Visual Channel. In: 2006 IEEE Symposium on Security and Privacy (2006)
18. Norman, D.A.: The Design of Everyday Things, Basic Books (September 2002)

19. Palmer, J.: Attentional limits on the perception and memory of visual information. Journal of Experimental Psychology: Human Perception and Performance 16(2), 332–350 (1990)
20. Suomalainen, J., Valkonen, J., Asokan, N.: Security associations in personal networks: A comparative analysis. Technical Report NRC-TR-2007-004, Nokia Research Center (2007)
21. Vaudenay, S.: Secure communications over insecure channels based on short authenticated strings. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 309–326. Springer, Heidelberg (2005)

Appendix A: Screenshots from Round One Implementation

Compare-and-confirm:
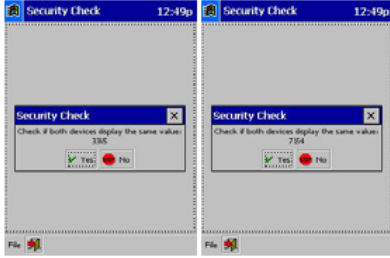
Select-and-Confirm (Selection and Confirmation Phases):

Copy-and-Confirm (Copy and Confirmation Phases)

Copy   Choose-and-Enter

Appendix B: Screenshots from Round Two Implementation

Compare-and-Confirm

Select-and-Confirm (Selection and confirmation Phases)
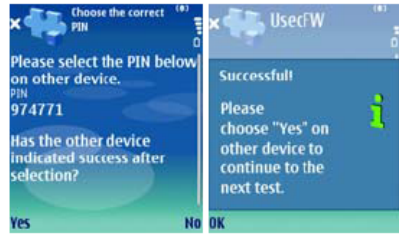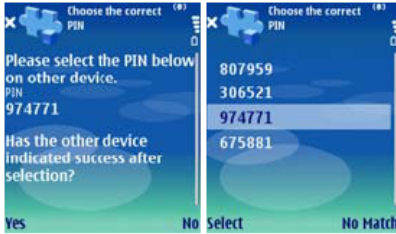
Copy

Framework screen in between tests
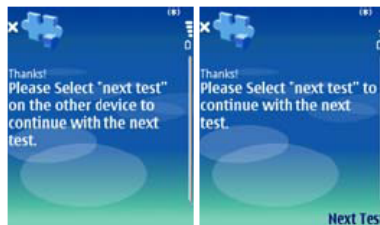
Appendix C: Background Questionnaire

Demographics
Age
☐ 18-24   ☐ 25-29   ☐ 30-34   ☐ 35-39   ☐ 40+

Nationality   Native Language

Sex
☐ Male   ☐ Female

Highest Grade Competed
☐ High School   ☐ Bachelor   ☐ Masters   ☐ Doctorate

If you are college graduate, please list your major

Computer experience
For how long you have been using computers?

On a typical day, how many hours do you work with computers?

Mobile device experience
Do you have any personal mobile device such as cell phone, PDA, pocket pc, smart phone?
☐ YES   ☐ NO

Is your mobile device capable of establishing any of Bluetooth, infrared or WI-FI connection?
☐ YES   ☐ NO   ☐ N/A

Do you use any of its Bluetooth, infrared or WI-FI functionality on a regular basis?
☐ YES (how often?   )   ☐ NO   ☐ N/A

Please check the corresponding box if you have done it with your mobile device before:
☐ Playing two-player mobile phone games
☐ Using a wireless headset with your mobile phone
☐ Connecting your computer or PDA to the internet using your mobile phone
☐ Wirelessly synchronizing your mobile phone calendar with your computer calendar

I currently use wireless communication in my following devices
    a.
    b.
    c.
    d.

In general, I feel secure while using wireless communication
☐ Agree   ☐ Disagree   ☐ Neutral   ☐ Don't Know

Appendix D: Post-Test Questionnaire

Please answer the following questions based on your experience using the methods. Where appropriate, we would appreciate if you would explain your answers and reasoning in the spaces provided or orally to us.

1. Please circle the method name that you think it is best described by the adjective

Easiest:	Compare	Select	Type
Hardest:	Compare	Select	Type
Professional:	Compare	Select	Type
Most secure:	Compare	Select	Type
Least secure:	Compare	Select	Type
Fun to use:	Compare	Select	Type
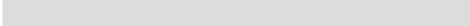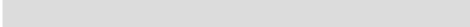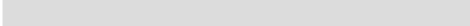
I would like my personal mobile device to be equipped with the following method(s):

Compare   Select   Type

2. I found the following aspects of certain methods very difficult to use

A.
B.
C.
D.
E.

4. I would prefer a method different from all of the above, or combination of those
☐ Yes (please explain it to us orally)
☐ No

5. I would find high level security useful in wireless connections to or from my mobile phone.
☐ Yes   ☐ No
If yes, how would it be helpful / useful?

If no, why would it not be helpful / useful?

6. Please add any comments in the space provided that you feel will help us to evaluate the methods or come up with a better one. We would especially appreciate your input on comparing the methods from different perspectives. (You can answer this question orally if you would like to).

Low-Cost Manufacturing, Usability, and Security: An Analysis of Bluetooth Simple Pairing and Wi-Fi Protected Setup

Cynthia Kuo[1], Jesse Walker[2], and Adrian Perrig[3]

[1] Carnegie Mellon University
cykuo@cmu.edu
[2] Intel Corporation
jesse.walker@intel.com
[3] Carnegie Mellon University
perrig@cmu.edu

Abstract. Bluetooth Simple Pairing and Wi-Fi Protected Setup specify mechanisms for exchanging authentication credentials in wireless networks. Both Simple Pairing and Protected Setup support multiple setup mechanisms, which increases security risks and hurts the user experience. To improve the security and usability of these specifications, we suggest defining a common baseline for hardware features and a consistent, interoperable user experience across devices.

1 Introduction

Bluetooth- and Wi-Fi-enabled devices are increasingly common. Already, manufacturers ship around 10 million Bluetooth units and 4 million Wi-Fi units *each week* [1,2]. Inevitably, consumers will perform security-sensitive transactions – including financial transactions – over Bluetooth or Wi-Fi. Thus, institutions should demand a basic level of assurance: that these technologies do not expose their systems or their customers' accounts to additional risks. This implies that (1) the security mechanisms in Bluetooth and Wi-Fi should be at least as strong as the rest of the system; and (2) the mechanisms should be easy to use so that consumers can configure and use them correctly.

We evaluate the security and usability of setup in the Bluetooth SIG's Simple Pairing specification (August 2006) [3] and the Wi-Fi Alliance's Protected Setup specification (released December 2006) [4]. These specifications were developed with two goals in mind: first, to make the technologies easy for non-expert users; and second, to address vulnerabilities in earlier versions of the technology. Simple Pairing and Protected Setup are not yet available in consumer products at the time of this writing; we present analysis based on the specifications.

Our description and analysis focus on the introduction of one device to another. In Simple Pairing, introduction enables two devices to communicate with one another via Bluetooth. In Protected Setup, it occurs when a device enrolls in an existing Wi-Fi network; we presume the initial setup of an access point has

already taken place. Both Simple Pairing and Protected Setup specify multiple methods for introduction. This creates a number of security and usability issues, which we examine in detail.

2   Properties of Secure and Usable Setup

In this section, we define properties required for the secure and usable setup of two wireless devices. From a security perspective, setup establishes a secure channel that provides secrecy and authenticity – even in the presence of an active adversary. From a usability perspective, the entire user experience should be intuitive, consistent, and robust. The following subsections will address each set of requirements.

2.1   Secure Setup Requirements

We evaluate the security of Bluetooth and Wi-Fi setup against three factors: 1) conformance to a standard model for authentication; 2) simplicity; and 3) level of security provided. We explain each factor below.

1) Conforms to the standard model for establishing authentication credentials. Wireless communication is inherently vulnerable to message injection and eavesdropping attacks; we cannot rely on the wireless channel alone for establishing credentials. Thus, we rely on an additional out-of-band channel.

The *standard model* for establishing authentication credentials consists of the two devices being introduced, the wireless communication channel (called the in-band channel), the additional out-of-band channel, and an active adversary that controls the in-band channel. In this paper, the in-band channel is Bluetooth or Wi-Fi. We adopt a Dolev-Yao active attacker, who can eavesdrop, insert, modify, delay, and reorder messages sent in the in-band channel. In the standard model, it is assumed that the active adversary cannot control the out-of-band channel.

Because devices are rarely asked to establish authentication credentials, some might argue that our threat model is too strong. Today, the chances that an attacker is present during setup is small. However, this may change: networking technologies, such as Bluetooth and Wi-Fi, are quickly becoming ubiquitous and inevitably will be used for sensitive (e.g., financial) transactions.

2) Preserves simplicity to reduce vulnerabilities. Experts can better find and correct vulnerabilities in simpler security designs. When a design is too complex, understanding whether it excludes all important vulnerabilities is infeasible; thus, the design must be assumed insecure.

3) Provides a high level of security. According to a NIST recommendation for key management, today's cryptographic mechanisms should require an attacker to perform at least 2^{80} operations through 2010. At least 2^{112} operations should be required to provide secure operation through 2030 [5].[1] If we require

---

[1] The NIST recommendations are intended for unclassified government data. However, the ANSI X9 standards, which govern the use of cryptography by the financial industry, historically align with NIST guidelines.

2^{80} operations and we assume that an attacker can perform 2^{50} operations, the attacker has no more than a 2^{-30} probability of success. For our analysis, expecting one guess in a billion to be successful (on average) is an acceptably low probability of attack success.

2.2   Usable Setup Requirements

A usable setup experience refers to all of the end user-facing details preceding, during, and following credential exchange. For example, a usable setup experience helps an end user: initiate credential exchange; identify precisely which devices are communicating (to the exclusion of other devices in range); understand whether the wireless connection is functional and secure; and recover from errors. We highlight three critical factors in usable network setup below:

1) Maintains a consistent user experience across devices. The setup process should be similar for any two devices. For instance, pairing a Bluetooth headset with a cell phone should feel congruous to enrolling a laptop in a Wi-Fi network. A consistent experience provides two main benefits. First, end users can learn how to perform the setup process and apply this knowledge to subsequent setup attempts. Second, vendors can better support their products. Nearly all network-enabled devices need to interoperate with devices from other manufacturers. By implementing the setup process in a consistent manner – for example, using the same user interaction flow – vendors will be able to anticipate how other devices behave. This facilitates producing more accurate documentation and providing better technical support.

2) Provides confirmation of which parties are communicating. End users need to be confident that the devices which are configured to communicate are the intended devices. Schemes such as Talking to Strangers [6] or Seeing-is-Believing [7] achieve this property through *demonstrative identification*, i.e., identifying which devices are communicating based on physical context. Also, devices should confirm the in-band connection is functional.

3) Incorporates robust error handling. Failure is a common outcome when adding new devices to a wireless network – even for experts. End users need comprehensible error messages when errors occur. This helps users troubleshoot the errors themselves and helps technical support staff with troubleshooting.

3   Bluetooth Simple Pairing

Bluetooth is a Personal Area Networking standard based on short range radios [8]. Devices such as phones, printers, modems, game consoles, and headsets use Bluetooth to communicate among themselves. Bluetooth is useful when two or more devices are in close proximity and require only modest bandwidth.

A Bluetooth device plays the role of either "master" or "slave." A master can communicate with up to seven slave devices, and a Bluetooth network consisting of one master and its slaves is called a piconet. The master controls the timing of all Bluetooth communications on a piconet.

The process of adding a new slave device to a Bluetooth piconet is called pairing. Bluetooth Simple Pairing [3] is a set of security enhancements to the Bluetooth pairing mechanism. The goal of Bluetooth Simple Pairing is to establish authentication credentials between the Bluetooth master and slave devices.

Bluetooth Simple Pairing supports four different pairing models: "Numeric Comparison," "Just Works," "Out of Band," and "Passkey Entry."

The **Numeric Comparison** model is intended when both devices can display a six digit number and both provide "Yes" and "No" buttons. For example, a PDA can use this pairing scheme with a PC. During the pairing process, each device displays a six digit number computed from the pairing protocol. The user of each device is supposed to compare the two numbers and select "Yes" if they match and "No" if they differ. Numeric Comparison is executed over Bluetooth, which is the in-band channel in the standard model for authentication. The display of the number on each device, the visual comparison of the numbers by human beings, and the Yes/No selection together comprise the out-of-band channel. Since there are six digits in the PIN ($= 10^6 \approx 2^{20}$ possibilities), an attacker can compromise the PIN with a probability of at least 2^{-20}.

The **Just Works** method is intended when at least one of the devices has no display or "Yes/No" buttons. A common use case is the pairing of a Bluetooth headset with a cell phone. This method uses Numeric Comparison internally, but does not display the six digits for comparison, even if one of the devices has a suitable display. Indeed, displaying the number is not useful, since

Fig. 1. Numeric Comparison

the corresponding value cannot be compared on the putatively paired device. Because the Just Works method lacks any out-of-band channel required by the standard model, this method provides no security against active attack.

The **"Out-of-band"** method can be used when an alternate communication medium exists on both devices, such as Near Field Communication (NFC). The alternate communication medium transfers a key between the intended devices and functions as the out-of-band channel in the standard model. Two parameters determine the amount of security possible with this pairing method. First, transfer of a larger key can

Fig. 2. Passkey Entry

provide more security, particularly when compared to other methods. Second, the efficacy of the alternate communication channel to resist adversarial control is important in determining security. If an attacker can read or write the transferred data, then the credentials established by the method may be compromised. Hence, the security of this method depends fundamentally on the user properly exercising the alternate communication channel.

The **Passkey Entry** method is intended when one of the devices has a display and the other a keypad. The device with the display randomly generates a six-digit number, and the user enters this on the other device using the keypad.

The displayed six-digit number, keypad, and human user together constitute the out-of-band channel for this method. Like Numeric Comparison, an attacker can compromise the six-digit passkey with a probability of at least 2^{-20}. However, this is only true the first time a passkey is used. The protocol splits the passkey into 20 bits and reveals one bit over 20 rounds of exchanges. An eavesdropper can compute each bit of the passkey after it has been sent. Thus, a passkey can only be used (securely) once.

4   Wi-Fi Protected Setup

IEEE 802.11, commonly called Wi-Fi, is a Local Area Network standard [9]. It is widely used in laptop computers, PDAs, cell phones, bar code scanners, and other mobile devices with significant bandwidth requirements.

Wi-Fi is usually deployed as an infrastructure network, which consists of one or more access points, and one or more mobile devices called stations. Each station forms a connection, called an association, with a single access point.

Wi-Fi uses the 802.11i standard [10] for security. 802.11i is also called WPA2. WPA2 uses the IETF EAP protocol [11] to mutually authenticate a station and the network and to derive a session key. The session key provides confidentiality, integrity, and origin authenticity for each frame that a station and its access point exchange. Thus, Wi-Fi security relies on a long-lived authentication credential being established between the station and the network.

Wi-Fi Protected Setup was developed to address consumers' credential configuration problem. It is more complex than Bluetooth Simple Pairing; the host/peripheral model constrains the Bluetooth approach, while Wi-Fi attempts to address more complex relationships among wireless devices. The Wi-Fi scheme uses three different devices: the registrar, which is the network enrollment center; an access point; and an enrollee, which is the device being added to the network.

Wi-Fi Protected Setup supports three setup methods: Push Button Configuration, PIN entry, and Out-of-band channel.

Push Button Configuration (PBC) has no security in the standard model. The user pushes buttons on both the registrar and the enrollee devices. The button push causes both to initiate an unauthenticated Diffie-Hellman exchange. The method assumes that the Diffie-Hellman peer is the correct device, i.e., that a malicious active attacker is not present. There is no out-of-band channel.

The **PIN** method is the Wi-Fi Protected Setup default. The enrollee device has a four- or eight-digit PIN which is entered on the registrar's keypad. The PIN method uses the PIN as an authentication key to protect a Diffie-Hellman exchange. The transfer of the PIN from the enrollee device to the registrar is the out-of-band channel for the PIN method.

Fig. 3. PIN

A random eight-digit PIN represents $10^8 = 2^{26.65}$ possibilities. However, the PIN protocol splits the PIN into two four-digit numbers. Each side commits to its value for each half of the PIN and exchanges
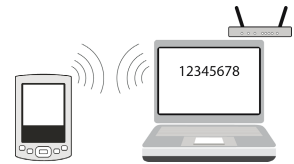
Table 1. Security Characteristics of Individual Setup Models

Our secure setup requirement looks for no more than a 2^{-30} probability of attack success.

	Out-of-band Channel in Standard Model	Probability of Attack Success
Bluetooth Simple Pairing		
Numeric Comparison	Display of 6-digit number on both devices; visual comparison of numbers and response input by human	2^{-20}
Just Works	None	Very likely
"Out-of-band"	The alternate communication channel	Depends on the selected channel
Passkey Entry	Display of 6-digit passkey on one device; entry of number into second device	2^{-20} for the first time a passkey is used
Wi-Fi Protected Setup		
Push Button Configuration	None	Very likely
PIN	Display of 4- or 8-digit PIN on one device; entry of PIN into second device	2^{-14} for the first time an 8-digit PIN is used; 2^{-7} for the first time a 4-digit PIN is used
"Out-of-band"	The alternate communication channel	Depends on the selected channel

information to reveal the PIN value. A man-in-the-middle attacker can guess each half of the PIN separately [12]. This means the attack success probability for an 8-digit PIN is at least 2^{-14} (to establish a connection with the registrar). Like Bluetooth's Passkey Entry, a PIN can only be used once; otherwise, a man-in-the-middle attacker can reconstruct the PIN and establish a connection with both parties.

The Wi-Fi "**Out-of-band**" method is similar to the Bluetooth out-of-band method. An alternate communication channel, such as an NFC channel, transfers some information between the registrar and the enrollee. This transfer constitutes the out-of-band channel for the method. It is possible to obtain an arbitrary amount of security in the standard model, provided the user actively participates in protecting the alternate channel from attack.

Thus, like Bluetooth Simple Pairing, Wi-Fi Protected Setup can meet commonly accepted security levels only in the case of its Out-of-band method, and then only with the active cooperation of the user.

5   What Causes Poor Security?

Table 1 summarizes our discussion of each setup model. Not only are there security issues in individual setup models, the multitude of setup methods also introduces unnecessary complexity: any two given devices may support two arbitrary sets of setup models. In system safety engineering, this problem is called *interactive complexity*. A system is *interactively complex* "when the level of interactions reaches the point where they cannot be thoroughly planned, understood, anticipated, and guarded against. In interactively complex systems, designers find it difficult to consider all the potential system states and operators have

difficulty handling all normal and abnormal situations and disturbances safely and effectively" [13].

Simple Pairing and Protected Setup are interactively complex. Bluetooth Simple Pairing specifies four pairing models, which mean there are $2^4 - 1 = 15$ combinations of setup models from which each vendor chooses. (Fifteen combinations is the power set of the four models, minus the empty set.) Between two Bluetooth devices, there are 120 possible setup combinations, which is the number of combinations (between two devices that each have 15 possible combinations of setup models) with repetition: $\binom{15+2-1}{15-1} = 120$. Similarly, Wi-Fi Protected Setup supports three setup models, and there are $2^3 - 1 = 7$ combinations from which each vendor chooses. Between two Wi-Fi devices, there are $\binom{7+2-1}{7-1} = 28$ possible setup combinations.

Fig. 4. Typical Protected Setup Protocol Diagram: PIN Enrollment in Wi-Fi Network

The specifications need to anticipate how these different combinations may interact with one another. However, it is challenging to thoroughly evaluate 120 or even 28 combinations.

While accommodating the needs of many vendors, these options make any design and implementation more prone to mistakes. The specification contains more details, which security experts must review. Vendors must decide how many and which of the setup models to implement.

The number of combinations could be reduced by prioritizing the setup methods. For instance, suppose two Wi-Fi devices each support Push Button Configuration (PBC) and Out-of-band. Out-of-band should receive higher priority than PBC. Otherwise, an attacker could implement a dumbing-down attack, forcing the two devices to use the insecure PBC method. Wi-Fi does not – but Bluetooth does – prioritize setup models.

There is another security issue that deserves discussion. Four setup methods do not require screens: Simple Pairing's Just Works and Out of Band methods; and Protected Setup's Push Button Configuration (PBC) and Out-of-band methods. The lack of screen-based feedback to the user could magnify errors and facilitate attacks.

Just Works and PBC were designed specifically for devices without screens, such as Bluetooth headsets or Wi-Fi-enabled printers. Both methods rely on timing and proximity for their security. As long as there are no other devices in setup mode and in wireless range of the intended devices, setup occurs between the intended devices. As long as there are no malicious devices in wireless range, setup is secure. Clearly, the potential for unintended outcomes exists. For example, imag-

ine using push button configuration on Christmas morning in New York City –
neighbors might connect to each other's Wi-Fi networks accidentally.

The Out-of-band methods in Simple Pairing and Protected Setup also rely on
device proximity, but the risk depends on the particular technology.

The security issues raised in this section can all be traced to the explosion of
setup options. Each option increases the complexity of the setup process – and
increases the possibility of mistakes in both design and implementation.

6   What Causes Poor Usability?

The multitude of setup methods not only detracts from the security of Simple
Pairing and Protected Setup, it diminishes the usability as well. Simple Pairing
and Protected Setup have not been introduced in consumer devices yet, but there
are indications that they are too complicated. This is evident not in specification
– but in what is *missing* from the specification. Many critical design choices
remain undefined.

Both specifications focus on a narrow subset of the setup experience: the
exchange of cryptographic keys. For example, Figure 4 shows a protocol diagram
for enrollment in Wi-Fi Protected Setup using a PIN. Figure 4 indicates that the
user only needs to enter the PIN number. Thus, the enrollment process appears
simple. However, the diagram omits all the steps leading up to and following the
credential exchange.

Appendix A lists some of the questions that implementers and end users will
face. Unless vendors coordinate their implementation efforts (which is unlikely),
many of the implementation questions will be pushed to end users. This means
that the setup process may be far more involved than Figure 4 indicates. Figure 5
shows one plausible scenario for the end user experience of Wi-Fi Protected
Setup. Note that Figure 5 is extremely optimistic, *ignoring potential errors* and
questions of which device is the registrar.

Figure 5 also ignores subtleties in the Wi-Fi Protected Setup specification. For
example, the end user decides whether a PIN will be copied from the enrollee
or the registrar. This has important implications for network setup. Suppose an
end user has an uninitiated device and an access point. Entering the AP's PIN
onto the device means that the device will be authorized to act as an external
registrar. Entering the device's PIN onto the AP means that the device will be
enrolled in the network without registrar authority. The distinction is subtle,
but the security implications may be significant.

Moreover, failing to address the questions in Appendix A could lead to non-
interoperable software. For example, a potential enrollee may only support push
button configuration; the registrar, produced by a security-conscious vendor,
may only support PIN and NFC configuration. Setup will clearly fail. Without
detailed specifications, implementers may make decisions that are incompatible
with one another. This has the potential to create a non-interoperable system –
even if the underlying protocols interoperate.

With the current specifications, we expect the following four usability issues will arise:

1. More setup models reduces consistency. Consistency allows users to apply what they learn from one situation to another, similar situation. It also increases users' confidence in their abilities, as they master applications quickly. Specifying several setup models reduces the consistency of the end user experience. The interaction flow from one setup model will be different from the flow of another model. As a result, the investment that users make in learning how to perform setup may not be fully leveraged. For example, learning how to compare numbers on Bluetooth devices may not benefit users when they set up Bluetooth devices via NFC. Users may be further confused when they use the PIN method for Wi-Fi devices.

2. The quality of error handling, documentation, and technical support suffers. Without a consistent user experience on every device, a vendor cannot anticipate the setup process that users will encounter. Thus, each vendor can only build error handling mechanisms for one-half of the setup process. Each vendor can only document one-half of the setup process. If the product documentation fails, users call technical support. Technical support may not be able to support products from different vendors. Technical support then shifts the blame, instructing users to call the other vendor.

Fig. 5. Example End User Decision Tree in Wi-Fi Protected Setup (Omitting Potential Errors)

3. A failure to require confirmation of which devices are communicating may lead to confusion and errors. Many setup scenarios will include devices without screens. Without clear feedback from a screen, users may

not receive confirmation that a wireless connection was successfully established. Without screens and adequate error messages, troubleshooting errors is nearly impossible. Also, if the credential exchange occurred via out-of-band channel, the devices should verify that the in-band connection was established successfully.

4. Users will not understand the level of security assurance associated with each setup model. As discussed in Sections 3 and 4, some setup models provide greater levels of security assurance than others. Simple Pairing and Protected Setup need to communicate that some connections are relatively secure, while others are not. Users should not be conducting sensitive business, for example, over connections established with Just Works or Push Button Configuration.

7   Discussion

As we discussed in the previous two sections, interactive complexity will cause numerous security and usability issues. The number of setup models in Bluetooth Simple Pairing and Wi-Fi Protected Setup needs to be reduced – preferably to one or two scenarios. Preferably the scenario(s) would be consistent across Bluetooth, Wi-Fi, and other technologies.

We argue that the security and usability of Simple Pairing and Protected Setup can be improved by two simple actions: specifying (1) a common denominator for hardware features and (2) a common user interaction flow for application software. We elaborate on these two points below.

1. Common denominator of hardware features. First, devices must ship with a common feature set. For example, suppose that we require a level of security where an attacker needs to perform at least 2^{80} operations to break a system, or, assuming that an attacker can perform 2^{50} operations, that an attacker has an attack success probability of at most 2^{-30}. The Out-of-band method is the only one capable of meeting such stringent requirements. Suppose vendors choose to use NFC. Ideally, all devices ship with a screen, 2 buttons, and NFC capability. At the minimum, one device ships with a screen, 2 buttons, and NFC capability; the other device possesses at least an LED, 2 buttons, and NFC capability.

We summarize the tradeoffs between in-band device capabilities, secure setup models, and feedback capability in Table 2. The table for out-of-band setup differs in that the security mechanism is identical for all combinations. However, the feedback capability is identical to what is shown in Table 2.

At least one screen is needed so that success confirmations and descriptive error messages can be relayed to the user. Ideally, both devices possess a screen; this would increase consistency, enable better confirmation of which two devices are communicating (by displaying information about the other device on the screen), and facilitate error handling.

Less capable devices – such as devices without screens or keypads – cannot be introduced to each other securely. Vendors should consider whether it is worthwhile to push Bluetooth or Wi-Fi into these devices.

Table 2. Tradeoffs between Manufacturing Cost, Secure Setup Mechanism, and Feedback Capability

		In-band Channel (Bluetooth, Wi-Fi)					
		Higher Manufacturing Cost			**Lower Manufacturing Cost**		
	Output	Screen	Screen	Screen	LED	LED	LED
	Input	Keypad	2 Buttons	1 Button	Keypad	2 Buttons	1 Button
Higher Cost	Screen Keypad	4 | H	4 | H	4 | H	2 | M	1 | M	1 | M
	Screen 2 Buttons		3 | H	3 | H	2 | M	0 | M	0 | M
	Screen 1 Button			3 | H	2 | M	0 | M	0 | M
Lower Cost	LED Keypad				1 | L	1 | L	1 | L
	LED 2 Buttons					0 | L	0 | L
	LED 1 Button						0 | L

Secure Setup Mechanism (Please see Table 1 for attack success probabilities.)

0	N/A. A pair of devices that lack both screens and keypads will be set up insecurely.
1	One device has a keypad with which the user can enter a PIN number (or alphanumeric string). This device may or may not have a screen. The other device has neither a screen nor a keypad, but only a static PIN number (i.e., printed on a sticker). Cryptographically, a static PIN can only be considered secure the first time it is used.
2	One device has a screen on which to display a generated PIN. The other device has a keypad but no screen. Users type the generated PIN on the second device.
3	Both devices have screens but lack a full keypad. Both devices display a generated PIN, and users compare whether the PIN is the same.
4	Both devices have screens, and at least one has a keypad. Setup occurs in one of two ways: the comparison method in option (3), or the PIN input method in (2).

Feedback Capability

L	Low. Neither device has a screen to display success or error messages.
M	Medium. One device has a screen to confirm a successful setup or to display error messages. The size and capabilities of the screen obviously limit the quality of user feedback; this indicator focuses on feedback capability, *not* general usability.
H	High. Both devices have screens for displaying feedback.

Retrofitting pre-existing hardware for a new security solution always requires some compromises. Bluetooth Simple Pairing and Wi-Fi Protected Setup are reasonable first steps for securing credential exchange. However, they should be viewed as transitory specifications. In the long run, the industry should aim for a single out-of-band channel that will be used for setup, whether it is NFC, USB, or some other technology.[2] Complementary technologies, such as decoy devices or scanners to detect attackers, can also be used to strengthen solutions.

2. Common user interaction flow for consistency in user interfaces. A common feature set is a necessary but not sufficient condition for creating a consistent user experience. Consistency does not mean that every user interface must be identical. Wireless setup can become more consistent simply by: ensuring

---

[2] There is a precedent for hardware changes in the design of 802.11i. 802.11i uses AES in CCM mode as its long-term solution. However, when 802.11i was designed, the majority of legacy Wi-Fi devices had microprocessors with insufficient available MIPs to support AES. 802.11i provides TKIP as a patch that can be deployed on legacy hardware.

Table 3. HomePlug Secure Mode vs. Bluetooth Passkey Entry and Wi-Fi PIN Methods

	Length	Composition	Printed on Sticker or Generated on Screen?
HomePlug	12	Alphanumeric	Printed on sticker
Bluetooth	6	Numeric	Generated on screen
Wi-Fi	4 or 8	Numeric	May be printed on sticker, but generated on screen preferred; 8-digit PINs recommended; 4-digit PINs acceptable for less capable screens

that the setup application appears in the same location and has the same name and icons across devices; implementing a similar interaction flow across devices; and specifying a framework for error messages and troubleshooting procedures.

Relying on user interfaces as market differentiators is a dated concept. The setup operations for networked devices must be interoperable on the user experience level.

8   Related Work

To date, researchers have focused more on the security of Bluetooth Simple Pairing and Wi-Fi Protected Setup than the usability. Suomalainen et al. analyze the security of several setup methods, including Simple Pairing and Protected Setup [12]. Nyberg presents a Man-in-the-Middle attack on (an earlier version of) Protected Setup [14]. (Researchers also noted vulnerabilities in earlier versions of Bluetooth pairing [15,16,17].) Uzun et al. compare the usability of different pairing methods and their implementations [18].

Newman et al.'s description of setup in HomePlug AV raises many design issues also discussed here [19]. In HomePlug, users select from two setup modes: Simple Connect Mode and Secure Mode. Simple Connect Mode is similar to Wi-Fi's Push Button Configuration. Secure Mode is analogous to Bluetooth's Passkey Entry and Wi-Fi's PIN methods; the three setup methods are compared in Table 3.

Like Bluetooth and Wi-Fi, HomePlug is designed to support a wide range of device capabilities, from computers to devices without screens and keyboards. Simple Connect Mode can be used with any device. If a device is accidentally recruited into the wrong network, a user resets the station until the correct network is found. If a rogue device is detected on a network, a user must reform the entire network to remove the device. Accidental recruitment will not occur with Secure Mode, but a sufficient user interface must be available.

Other schemes for exchanging authentication credentials using demonstrative identification include Stajano and Anderson's Resurrecting Duckling [20], Balfanz et al's Talking to Strangers [6], Balfanz et al's Network-in-a-Box [21], and McCune et al's Seeing-Is-Believing [7]. In Resurrecting Duckling, an uninitialized network node uses the first key that it receives. Ideally, the imprinted key is transferred using an out-of-band channel. Talking to Strangers proposes using location-limited channels, such as audio or infrared, for credential exchange. This idea is extended in Network-in-a-Box, which uses infrared to secure a wireless network. In Seeing-Is-Believing, cell phone cameras take pictures of 2D barcodes,

which encode public keys. For mutual authentication, each device displays its unique 2D barcode, and the opposite device takes a picture of the barcode. Devices can also act as intermediaries for less capable devices.

9   Conclusion

This focus of this paper is not security setup per se; it is about making setup processes consistent. Consistency makes setup more usable – and by extension, more secure. Security features can only benefit consumers if setup is successful.

Many of the problems in Bluetooth Simple Pairing and Wi-Fi Protected Setup stem from the multitude of setup methods available. Several methods exist to accommodate vendors who opt for lower manufacturing costs. The feature sets selected for lower costs *force* system designers to use setup methods that are neither usable nor secure.

Simple Pairing and Protected Setup could be improved by:

1. requiring a common set of hardware features for compliant devices; and
2. specifying a consistent user experience, via common menu options, common user interaction flows, and a common framework for error logging.

This requires that the specifications converge to a small number of setup scenarios – preferably one, maybe two. It may raise some vendors' manufacturing costs, but consumers will be better able to setup wireless devices themselves.

Acknowledgements

References

1. Bluetooth SIG: Authorities raid Chinese factory suspected of infringing on Bluetooth SIG registered trademarks (September 2006) `http://www.bluetooth.com/Bluetooth/Press/SIG/AUTHORITIES_RAID_CHINESE_FACTORY_SUSPECTED_OF_INFRINGING_ON_BLUETOOTH_SIG_REGISTERED_TRADEMARKS.htm`
2. In-Stat: Year over year Wi-Fi chipset sales. Personal Communication, Kelly Davis-Felner (October 2006)

3. Linsky, J., Bourk, T., Findikli, A., Hulvey, R., Ding, S., Heydon, R., Singer, S., Kingston, S., Wenham, S., Suvak, D., Edlund, M., Chen, P., Aissi, S., Hauser, P., Benaloh, J., Yuval, G., Yacobi, Y., Lafky, J., Simon, D., Roberts, D., Stanwyck, D., Lauter, K., Muchnik, G., Kerai, K., Nyberg, K., Asokan, N., Lobo, N., Ginzboorg, P., Everaere, D., Meindl, R., Bertoni, G., Reuveni, E., Shimojo, Y.: Simple Pairing Whitepaper, revision v10r00 (August 2006), http://www.bluetooth.com/NR/rdonlyres/0A0B3F36-D15F-4470-85A6-F2CCFA26F70F/0/SimplePairing_WP_V10r00.pdf

4. Lortz, V., Roberts, D., Erdmann, B., Dawidowsky, F., Hayes, K., Yee, J.C., Ishidoshiro, T.: Wi-Fi Simple Config Specification, version 1.0a (February 2006)

5. Barker, E., Barker, W., Burr, W., Polk, W., Smid, M.: National Institute of Standards and Technology (NIST) Special Publication 800-57 (Draft): Recommendation for Key Management - Part 1 General (Revised) (May 2006)

6. Balfanz, D., Smetters, D., Stewart, P., Wong, H.C.: Talking to Strangers: Authentication in ad-hoc wireless networks. In: NDSS 2002. Proceedings of the Symposium on Network and Distributed Systems Security, San Diego, CA, Internet Society (February 2002)

7. McCune, J.M., Perrig, A., Reiter, M.K.: Seeing-is-believing: Using camera phones for human-verifiable authentication. In: Proceedings of the IEEE Symposium on Security and Privacy (2005)

8. IEEE: IEEE 802.15.1-2005 – IEEE Standard for Information Technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific Requirements. Part 15.1: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Networks (WPANs(tm)) (2005)

9. IEEE: IEEE 802.11-1999 – IEEE Standard for Information Technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific Requirements. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications (2003)

10. IEEE: IEEE 802.11-1999 – IEEE Standard for Information Technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific Requirements. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications – Amendment 6: Medium Access Control Security Enhancements (2004)

11. Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., Levkowetz, H.: RFC 3748: Extensible Authentication Protocol (June 2004)

12. Suomalainen, J., Valkonen, J., Asokan, N.: Security associations in personal networks: A comparative analysis. Technical report, Nokia Research Center, Technical Report NRC-TR-2007-004 (2007)

13. Leveson, N.: System Safety Engineering: Back to the Future (2002), http://sunnyday.mit.edu/book2.pdf

14. Nyberg, K.: Connect Now to MitM. In: Presentation at Crypto 2006 Rump Session (August 2006)

15. Jakobsson, M., Wetzel, S.: Security weaknesses in Bluetooth. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 176–191. Springer, Heidelberg (2001)

16. Whitehouse, O.: Presentation at RUXCON (2004), http://www.ruxcon.org.au/files/2004/12-ollie_whitehouse.pdf

17. Shaked, Y., Wool, A.: Cracking the Bluetooth PIN. In: MobiSys. The Third International Conference on Mobile Systems, Applications, and Services, pp. 39–50 (June 2005)

18. Uzun, E., Karvonen, K., Asokan, N.: Usability analysis of secure pairing methods. In: Dietrich, S., Dhamija, R., (eds.) FC 2007 and USEC 2007. LNCS, vol. 4886, pp. 307–324. Springer, Heidelberg (2007)
19. Newman, R., Gavette, S., Yonge, L., Anderson, R.: Protecting domestic power-line communications. In: SOUPS. Symposium On Usable Privacy and Security (July 2006)
20. Stajano, F., Anderson, R.: The Resurrecting Duckling: Security issues for ad-hoc wireless networks. In: Malcolm, J.A., Christianson, B., Crispo, B., Roe, M. (eds.) Security Protocols. LNCS, vol. 1796, pp. 172–194. Springer, Heidelberg (2000)
21. Balfanz, D., Durfee, G., Grinter, R.E., Smetters, D.K., Stewart, P.: Network-in-a-Box: How to set up a secure wireless network in under a minute. In: USENIX. Proceedings of the 13th USENIX Security Symposium (August 2004)

A   Questions Left Unanswered in the Bluetooth and Wi-Fi Specifications

A.1   Implementers

– Who initiates the setup process? Does the user set both devices into setup mode? Does one device always look for other devices in setup mode?
– Where is Protected Setup/Simple Pairing application located in OS menu? How does a user initiate setup?
– Does the application check if wireless is enabled? Should wireless be turned on automatically?
– If multiple setup methods are available, which method will be used for setup? Who will decide? The devices? Will more secure methods take precedence over less secure methods? Will users decide?
– For Bluetooth Numeric Comparison and Wi-Fi PIN methods: Which device (if any) generates the PIN? How is this decided? By the devices or the user?
– Is there a timeout value for a generated PIN? What is it?
– For the Bluetooth Just Works scenario, should a device just accept a connection, or prompt the user?
– Is there a timeout value for Just Works mode? What is it?
– Which device or manual (if any) provides directions on what the user should do?
– Which setup methods will a device support?
– Which device (if any) is logging data to aid troubleshooting?
– For Wi-Fi Protected Setup: Does the access point need to be present during enrollment? What happens if the enrollee and the registrar are out of WLAN range?
– For Wi-Fi Protected Setup: What device keeps a record of the keys that have been issued?
– If the out of band channel is used for setup, will the in-band connection be verified?

A.2   Users

– My AP has a PIN, my phone has a PIN, my computer has a PIN. Which PIN do I enter where?
– Which devices or manuals should I consult to confirm whether setup succeeded?
– Whose tech support line should I call if setup failed?

Empirical Studies on Software Notices to Inform Policy Makers and Usability Designers⋆

Jens Grossklags and Nathan Good

School of Information, University of California, Berkeley
102 South Hall, Berkeley, CA, 94720, USA
{jensg, ngood}@ischool.berkeley.edu

Abstract. We evaluate the usability of End User License Agreements (EULAs) of popular consumer programs. Results from an empirical evaluation of 50 popular programs show the lack of accessibility and readability of notices. Our data from a recent study with 64 users involved in installation tasks confirms the public perception that notice to and consent by the user is not achieved.

Keywords: HCI, privacy, security, legal issues, End User License Agreement, notice.

1   Introduction

Human-Computer Interaction (HCI) research is of increasing importance to security researchers as well as policy makers. As Internet access has become more prevalent, many issues that previously concerned only a few sophisticated technical users are now issues affecting the public at large. Issues surrounding digital privacy, copyright, electronic voting, notice & consent, and location-based systems are being pushed into the public policy arena because of commoditization of technology. Public policy advocates have traditionally accessed academic research as one means of understanding a problem, and HCI research provides a deeper understanding of the many technological issues discussed today. Indeed, many of the recent issues with new technologies have roots in problems that HCI has dealt with for years. For example, inadequate usability of programs and systems have caused security and privacy concerns for a broad range of issues such

as the sharing of private personal information over P2P networks [1], Phishing attacks [2], electronic voting machines [3], and email message encryption [4].

Our research focuses on the primary means that security and privacy related information is currently communicated to the end user: the software notice and license agreements. We find software with potentially unwanted consequences and risks such as Spyware and Adware to be a particularly significant field of study. We observe that in the marketplace millions of programs are installed bundled with advertisements and privacy-invading technologies [5] [6]. Many of these installations are made without any notice and consent procedures (e.g., through drive-by downloads), however, a surprisingly large number of programs are installed through deliberate user action and involve some form consent process. Users desire the functionality of programs they download, but frequently seem ill-informed about potential risks and negative consequences of installations. Moreover, the reason that Spyware is difficult to accurately define is that the same piece of software may be considered unacceptable Spyware by one user, an acceptable trade for other services by another, or a valuable personalization system by a third [7] [8]. Because of this user-centered definition of what constitutes Spyware, for some portion of software that meets the definition of Spyware, it seems inappropriate to adopt an outright ban. Early efforts to combat Spyware (much like anti-virus software efforts) measured their success based on how infrequently the software was installed. While such a measure can help provide security, it may also limit users' access to certain software combinations by denying them the opportunity to trade some privacy, speed, or attention for services or information they actually value. Imagine if your computer 'protected' you by preventing you from ever transmitting your credit card information over the Internet; it would perhaps reduce your vulnerability to identity theft, but would at the same time deny you the benefits of shopping online. As a response to usage restrictions due to security software (e.g., firewalls, anti-spyware) users might experience frustration. Left with their dissatisfaction users will often disable security technologies and, therefore, reduce overall security of the computer system.

Our work is relevant to the public policy debate on the balance of power between consumers and commercial entities as it is primarily represented by the terms of standard form contracts (and it has been estimated that 99% of all commercial contracts are standard form contracts [9]). On the one hand, businesses strive for monetary earnings but want to minimize potential liabilities out of transactions conducted in the marketplace. Accordingly, the typical vendor software license has much less to do with the licensing of technology than it does with the creation of multiple revenue streams flowing from the user to the vendor and the elimination or minimization of most forms of accountability from the vendor to the user [10]. On the other hand, users want to benefit from the functionality of a program and other aspects that create hedonic and intangible values while limiting privacy, security and other risks of the interaction. Further, users want to reduce the effort involved in making sound decisions; standard form contracts help in an overwhelming number of situations to reduce transaction costs for businesses and consumers.

Generally, economic forces should help to balance consumer desires and concerns with business interests. However, a recent research study supported the view that market conditions are generally uncorrelated with contract terms (for example, by asking how price and market concentration determine the harshness of contract terms). The study also indicated that license terms on average provide less consumer protection than the Uniform Commercial Code baseline regulations [11].

In absence of simpler and more conspicuous modes of communication to the consumer (e.g., short notices [12]) these agreements also serve as important information sources for download and installation decisions by communicating privacy and security choices. Our prior research suggests that users are often even uninformed about aspects of a program they genuinely are concerned about (such as pop-up advertisements and Spyware). The result is unwanted installations of programs that are later regretted [8]. The current paper explores this disconnect between consumer wishes and their market choices in more detail.

Our research task is focused on evaluating the readability and usability of End User License Agreements (EULAs) that represent the legal state of the art of informing users and obtaining user consent for software. In Section 2 we present preliminary results from an empirical study of 50 popular consumer programs on the accessibility and readability of the associated EULAs. In Section 3 we present selected results from a user study involving 64 users in program installation tasks. Users were observed during their interaction with an experimental program installation environment. We recorded their reading behavior, decisions to complete or cancel an installation and their responses to post-experimental surveys.

Both studies are significant extensions of our prior work [8]. On the one hand, we discussed in our first paper the readability metrics of only 5 programs that we randomly selected. The current study gives a more thorough overview of the notice of consent practices for an important sample of 50 consumer programs that are the most popular freeware/shareware or free-to-test versions across multiple functional categories. On the other hand, we also conducted a more thorough experimental analysis. In Good *et al.* [8] we reported results of an in-depth user study on notices with a small sample set of 30 users across three experimental conditions. Many questions were left open and in need of further experimentation to determine or substantiate results.

2   Empirical Study of End User License Agreements

As the data set, we chose Download.com's top 50 most downloaded software programs for the week ending April 9, 2006. Download.com is a popular source for primarily free or free-to-try consumer software downloads covering major software vendors as well as small distributors but the program offerings are not necessarily representative of all consumer programs available.

Related to our study Kucera *et al.* [13] reported on the prevalence of Spyware in a similar sample of download.com's most popular programs.[1] When defining

---

[1] Kucera *et al.* [13] obtained data for the week ending January 12, 2003.

Spyware narrowly as programs that surreptitiously collect personal information from computers linked to the Internet the authors confirmed the existence of Spyware for three of those programs. The current policy of the distributor does not allow for software including viruses or Spyware.[2] However, the website does not provide a clear definition of these terms.

Recently, the Anti-Spyware Coalition formulated a broader characterization of Spyware (and other potentially unwanted software).[3] Their definition includes technologies deployed without appropriate user consent and/or implemented in ways that impair user control over: (1) Material changes that affect their user experience, privacy, or system security; (2) Use of their system resources, including what programs are installed on their computers; and/or (3) Collection, use, and distribution of their personal or other sensitive information.

Our focus in this study is on analyzing the readability of license terms distributed with typical software available to consumers. We defer the content analysis of these agreements and a technical analysis to later stages of our research. It is to be expected that many of the terms are unremarkable and of little concern to the user [14]. However, we note that our preliminary analysis suggests that the programs included in our sample included terms (including privacy implications, restrictions of usage and legal rights, distribution of Adware) that are likely in conflict with the preferences of many consumers and may overlap with a broader definition of Spyware.

2.1 Timing and Presentation of the End User License Agreement Presentation to the User

For each software program, we initiated the downloading and installation process, and stopped the process at the point where we encountered a EULA. We copied the EULA that appeared on-screen and canceled the download at this point, and thus did not capture any additional terms that may have been presented to the user after this point. If we did not encounter a EULA during the installation process or after program installation we expanded the search to the distributors' website. See Fig. 1 for a typical display situation of a EULA during the installation process.

We observed that the terms were presented at different stages during the installation process for different programs: e.g., before the installation had begun or after the installation process. Knight Online 1.299 showcased a so-called 'first-run notice' that occurs the first time a (or potentially each) user starts the program.[4]

---

[2] See, for example, http://www.upload.com/1200-21_5-5081541.html, last visited February 5, 2007.

[3] http://www.antispywarecoalition.org/documents/DefinitionsJune292006.htm

[4] A recent report by Microsoft [15] distinguishes between Just-In-Time, First Run, Installation Time, and Out-of-The-Box notices. Out-of-the-Box notices were not observable from our download.com sample. We did not test for Just-In-Time notices that occur in the moment before sensitive data is transmitted or some other potentially harmful or unwanted action is undertaken by the program. The majority of the programs featured Installation Time notices.

Common sense regarding notice and consent would dictate that information to users should be provided before an installation is initiated or completed. In research reported elsewhere we investigate the impact of the timing of notices more thoroughly [12].

Fig. 1. End User License Agreement presentation during installation (McAfee AntiVirus vso_10027_en-us-30day)

More problematic from an accessibility standpoint is the omission of notice during the installation process. Adobe Reader and Irfanview, had EULAs only on their websites. It is doubtful whether users would search for these terms if not included in the installation dialog. One further significant difference appeared between the two programs. Irfanview's installer was directly accessible at Download.com's website. Users interested in Acrobat were redirected to Adobe to initiate download.[5] Under the 'download' button on Adobe's site the EULA was accessible by clicking on a link "By downloading software from the Adobe web site, you agree to the terms of our license agreements [...]". Software providers differ in the type and presentation of notice they provide to the user. From a legal perspective these installation scenarios introduce different challenges and likely impose different consequences on the user. For example, courts have started to differentiate between different modes of presentation of notice when they decide whether a user is bound by terms. See, for example, Casamiquela [16] who discusses caselaw and legal theory on browsewrap versus clickwrap agreements.[6]

---

[5] We expect that many users have access to Adobe Acrobat's installation file also without visiting Adobe's website.

[6] Clickwrap agreements include scenarios in which a software vendor requires users to click "I agree" or similar buttons or click-check radio buttons or boxes to signify consent. A license is likely to be characterized as browsewrap if only a link to the terms is available to the user instead of the complete license.

We were unable to locate a license agreement for Limewire, Limewire (Mac), and Morpheus on the respective company websites or during the installation. There is anecdotal evidence that file-sharing companies refrain from using EU-LAs in an effort to limit possible liability for contributory infringement, as the presence of a license agreement would establish an ongoing relationship with the customer. While the typical presentation of a EULA follows the pattern observable in Fig. 1 access to the agreement is not always obvious. For example, the installation dialogue displayed in Fig. 2 only links to the read me file that, however, contains a contractual document.[7] We believe that this access regime is from a user point of view totally unexpected.

EULAs were often presented in a format that limits users in gaining a quick overview over the terms covered. For example, the notice screen displayed in Fig. 1 allows the user to only review about 50 words at a time without scrolling. The complete notice, however, is 5500 words long.

2.2   Length of EULAs and Simple Readability Measures

For the software programs which had EULAs, the average EULA length was 2752 words (std.dev. = 2228.8), corresponding to about 11 pages of double-spaced text. Assuming that the reading difficulty of the EULA is average as reported in psychology research, the average reading time for the EULA is about 13 minutes.[8] The shortest EULA (Little Fighter) was 111 words, the longest (Adobe Reader) was 9313 words, corresponding to approximately 41 pages of double spaced text and an average reading time of 47 minutes. It is likely that the length of time it will take an average consumer to understand the EULA is even longer than 47 minutes.

Practically, the average EULA is even longer. Many of the EULAs have website links to additional information and terms that are incorporated into the EULA, such as Terms of Use, Terms of Service, Privacy Policies, and third party EULAs. For this study we did not review any of these additional linked documents. However, to fully understand what they were agreeing to, the user would also have to research various statutes and rules that are mentioned within the text of the EULAs, such as the Commercial Arbitration Rules of the American Arbitration Association. In one particularly egregious example, Good *et al.* [8] evaluated a KaZaA EULA, and noted that it contained 17458 words in

---

[7] Excerpt from readme file: "THIS SOFTWARE IS PROVIDED AS IS WITHOUT WARRANTY OF ANY KIND, WHETHER EXPRESS OR IMPLIED, INCLUDING WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE WHICH ARE HEREBY DISCLAIMED. [...]"

[8] Assuming an average reading rate of 200 words/minute. Lewandowski *et al.* [17] found an average reading rate for college students of 189 words/minute when subjects were given oral reading probes measuring words read correctly per minute (WRCM). Younger students and elderly citizens will likely read and comprehend slower on average. One reviewer correctly observed that it would be more precise to measure reading speeds specifically for EULAs. We have so far not conducted the required experiment.

the EULA itself, 4 hyperlinks to outside sites and policies, 78 locations of third parties and policies, and 5 opt-out options, and would take an average reader approximately 88 minutes to read.

We computed the Flesch-Kincaid Reading Level [18], and Flesch Reading Ease levels [19] for the EULAs in the program sample. The Flesch-Kincaid Reading Level uses average sentence length and average number of syllables per word to give a rough measure of a document's readability.[9] The scores range from 1.0 to 12.0, corresponding to the reading level of an average student in grades 1 through 12, respectively.[10] 63% of the EULAs scored 12.0, the highest score possible. The average score was 11.2 (std.dev. = 1.6), with scores ranging from 5.7 to 12.0. Because the scores were bounded by 12.0, and because of the large percentage of EULAs with the maximum score, the average score of 11.3 is likely skewed lower than it should be.

The Flesch Reading Ease also uses average sentence length and average number of syllables per word to give a rough measure of a document's readability. The Reading Ease scoring scale ranges from 0 to 100, with a higher score corresponding to easier reading ease. As a rule of thumb, scores of 90-100 are considered easily understandable by an average 5th grader. 8th and 9th grade students could easily understand passages with a score of 60-70, and passages with results of 0-30 are best understood by college graduates. Reader's Digest magazine has a readability index of about 65, Time magazine scores about 52, and the Harvard Law Review has a general readability score in the low 30s. This test has become a U.S. governmental standard. Many government agencies require documents of forms to meet specific readability levels. Most states require insurance forms to score 40-50 on the test.

The average Reading Ease score was 35.7 (std.dev. = 10.7), with a low of 18.5 (WinZip), and a high of 69.8 (Mario Forever). Fully 89% of the EULAs scored under 50, and only 1 EULA (Mario Forever) scored in the ideal range when for writing for the general population (60-70).[11] As no EULA scored at either extreme end of the range, the Reading Ease score is likely a better measure of readability than the Reading Level score. Readability studies were conducted for the domain of privacy notices. Jensen and Potts [20] found an average reading ease of 34.2 for popular entertainment websites and 36.5 for health care sites. Breese and Burman [21] found a reading ease of 42.2 for privacy practices of 185

---

[9] One critic of the Flesch-Kincaid models noted that "to measure readability, coherence and comprehensiveness of a text, more than surface features need to be taken in consideration. Quantitative and qualitative factors like the number of anaphora, number of overlapping text segment, vocabulary difficulty, sentence and text structure, concreteness and abstractness, are equally needed. It is the sum of these and other factors that constitutes cohesion." University of Memphis Institute of Education Sciences, http://cohmetrix.memphis.edu/cohmetrixpr/readability.html, last visited on May 11, 2006.

[10] The program we used for the computations does not allow for measures larger than 12.

[11] See e.g., http://www.diabetesvoice.org/issues/2004-09/Diabetes-related_websites_are_they_readable.pdf

Fig. 2. Need for Speed Most Wanted PC demo installation dialogue

institutions listed in the 2004 US News & World Report's 'best hospitals' issue. Hochhauser [22] found an average reading ease of 34 for 60 financial privacy notices (and a level of 39 for 31 HIPAA notices [23]).

If privacy or security risks are disclosed in the EULA then the length and reading ease will directly impact users' comprehension and decision making. However, if consumers cannot understand the terms to which they are ostensibly agreeing, have they really formed a valid contract with the company, or do they have a duty to read?

Hochhauser [22] suggests that several language and presentation modifications can be undertaken to improve readability and understanding. For example, the use of active everyday language, short explanatory sentences in bulleted lists, avoiding imprecise language including double negatives and effective highlighting of important terms can contribute to reader's improved decision making.

But grammatical simplification of contracts will not solve all comprehension problems. Research by, for example, Masson and Waldron /citeMasson demonstrates that the success of simplification of sentence structure etc. is hampered through the complexity of the legal concepts that are at the heart of online notices. Not only legal concepts are hard to understand. Acquisti and Grossklags [25] discuss consumers' limited knowledge and understanding of privacy and security risks. Further, misaligned economic incentives limit distributors' desire to improve EULA terms (see, for example, Vila *et al.* [26]) and consumers feel that it is not worth it to read notices [8] [26] [27].

Some commentators have discussed the role of experts, consumer advocates and user-to-user recommendations as a tool to improve decision making. For example, Hillman [27] argues that mandatory display of license terms on Web sites will improve access of consumer protection organizations. However, he cautions that the improved accessibility might backfire (at consumer rights) if terms still do not receive added scrutiny, or are not read more often compared to the cur-

rent notice regime. Download.com alone distributes 35000 programs - it appears unlikely that even all somewhat popular programs do receive enough scrutiny.

In future work we aim to more closely research interface aspects of EULA presentation to the user. We are also interested in analyzing the contents of these agreements to a greater extent.

3   Experiment

Below we report survey results and basic reading time measures observed in the experimental part of the project.

3.1   Experimental Setup

The complete experiment consisted of a set of recorded installation decisions, followed by two surveys. Subjects were given a unique number, and sheet outlining the basic scenario of the experiment. All of the experiments and surveys were done by each subject independently on a computer located in a laboratory with dividers. As the user passed each portion of the experiment, the application would record the actions and provide the next portion of the experiment.

The experimental portion of our framework was designed to mimic the experience of installing software applications, but also allows us to modify the notice and consent process encountered. We constructed a windows application in C# that would not only depict the installation process as realistic as possible, but also log all user actions (e.g., buttons clicked, time per screen) during the study. Additionally, the application we constructed would provide a launching pad that could dynamically configure each subject's experience based on their user number we provided at the beginning of the experiment. At any time, a
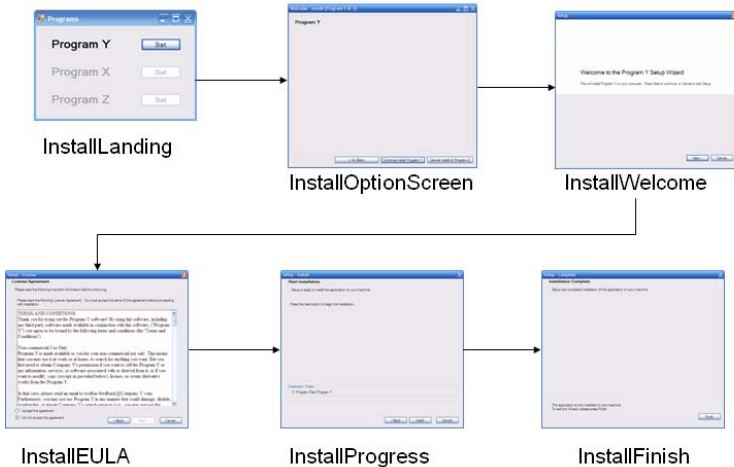
Fig. 3. Framework architecture of experiment

user may cancel the installation and return to the landing screen to start with the next program. Additionally, users may move back and forth between screens as in typical installation programs by hitting the back key. Participants' id was matched up with a random program ordering. A representation of the framework architecture is presented in Fig. 3.

We selected popular consumer programs from our previous study [8] to facilitate comparability of the results and user experience. We chose a browser toolbar, a weather information service and a file sharing application. For the experiment each brand name was removed and replaced with a generic title. The experimental program titles and descriptions are: (Program X) - Weather Information Program, (Program Y) - Browser Toolbar, and (Program Z) - File Sharing Program. We also replaced the brand names and other identifying information in the EULA statements with the above generic titles.

All three programs vendors disclose in the End User License Agreements that they take significant influence on the user's desktop experience. They differ in the disclosed impact on privacy and security. Some aspects of these programs fall within the broader definition of Spyware and Adware.

64 subjects participated in this part of the experiment.[12] Subjects were paid $20 for their participation, and were recruited by a university service with access to a subject pool of several thousand students. On average we had a young and very computer-experienced group of users. For example, More than 80% stated that they maintained their home computer themselves.

3.2   Survey Results

Only very few users reported reading EULAs often and thoroughly when they encounter them (1.4%). Members of a larger group categorize themselves as those who often read parts of the agreement or browse contents (24.8%). However, 66.2% admit to rarely reading or browsing the contents of EULAs, and 7.7% indicated that they have not noticed these agreements in the past or have never read them.

Supporting these results, Jensen and Potts [20] report that for a university service standalone website requiring registration only 0.24% of over 50000 users visited the site's privacy policy. Another software provider reported from an experiment in which a $1000 cash prize was offered in the EULA that was displayed during each software installation, yet the prize was only claimed after 4 months and 3,000 downloads of the software [28].

3.3   Reading Behavior in the Experiment

In this paper we report data for individuals that installed programs X, Y, and/or Z leaving us with 45, 58, 55 observations for the respective programs.[13] On

---

[12] Until now we have completed three experimental treatments with a total of 240 users. Complete results of these experiments are reported in Good *et al.* [12] in which we focus on short notices and the timing of notice presentation to the user (see this paper for more details on the user population and experimental setup).

[13] Subjects that canceled installation did not always progress through the installation routine to the point at which they were able to review the EULA.

average individuals resided on the screen that showed the complete End User License Agreement in a scrollbox for one or two minutes (Program X: 59.7 sec, 66.4 std.dev.; Program Y: 64.9 sec, 64.4 std.dev.; Program Z (with outliers): 162.6 sec; 323 std.dev.; Program Z (without 2 outliers): 106.6 sec, 141.0 std.dev.). More than 55% of the experimental subjects spent less than one minute on this screen. Only 3.7% deliberate on this screen for more than 5 minutes. It appears the installation of the filesharing program Z caused more individuals to slow down in their reading behavior. We plotted the reading times for the three different programs in Fig. 4. The theoretical time required to pass through the EULAs is 14 min, 10 min, 14 min for Program X, Y, and Z, respectively.[14]

We were also interested in the time individuals spent on the EULA screen in comparison to the other parts of the installation dialogue. Since this screen was the only one that contained important information about the program we would expect the ratio between the two measures to be below one. The other screens prompted individuals to merely click to continue. Up to 32.8% of the users spent more time clicking through screens without important information compared to the EULA screen (Program X: 71.1%; Program Y: 67.2%; 74.5%).[15]
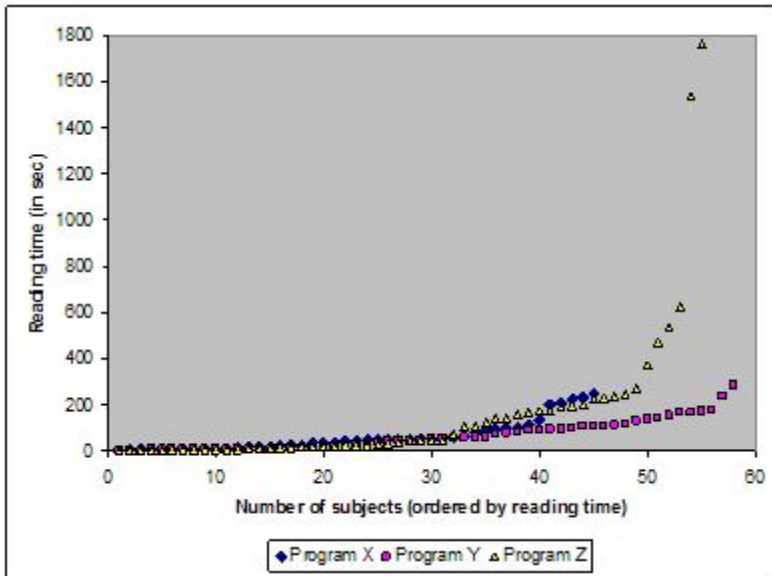
Fig. 4. Reading time for End User License Agreement Screen for the three different programs (in sec)

---

[14] Again, using an average reading speed of 200 words/minute [17].

[15] This cut-off level is somewhat arbitrary, but we posit that the reading time on the EULA screen should be, in general, a multiple of the time spent on basically content-free screens that merely state a generic program name and progress of the installation process.

3.4   Discussion

In contrast to EULA statements, food labels and credit card statements have been subject to substantial standardization and simplification. However, complete information about food ingredients and consequences of signing up for a new credit card are difficult to present to the user in a unified format and labels always need to be selective. Different states take different approaches towards what warnings and information are useful for consumers in their decision making. Similarly, consumer perceptions and reading behavior varies widely across the population. Individuals' health concerns are a strong driver for reading behavior. For example, Kreuter *et al.* [29] found that patients with high blood pressure searched labels for sodium information, however, did not investigate other ingredients more often than the rest of the population.

With respect to software installations the presence of individual differences in reading behavior and other behaviors suggests that personalized solutions have promise. Analogously, consumers with certain allergies are insufficiently supported by many current food labels. Some Web users might be well-served by the current notice and EULA system, or would be with short summary notices. Others seem likely to ignore such notices and might be willing to accept more restrictions on their installation (e.g., longer delays sequences of confirmations, or approval from another individual) in order to reduce their own risk and later regret. There are many paths to explore in this direction. We also note that a state-by-state approach seems unworkable for program downloads from the Internet. Therefore, enforcement action will likely be needed from the federal government or agencies such as the FTC.

The results serve as a benchmark for reading behavior if individuals are unaffected by brand recognition, message framing and sophisticated user interface design techniques. It is not a reading speed test. Rather the study provides insight into the distribution of reading times across a reasonably-sized subject group in a controlled laboratory context. Surprisingly, even without prior knowledge of the programs' names or about the programs' terms concerning privacy, security and usage rights and without time pressure almost no subjects spend enough time on the EULA screen to pass through the notice agreement. In contrast, Hillman [27] reported that one third of the law student respondents to his survey would more likely read notices if the vendor is unknown.

Well-known limitations of laboratory studies apply also to our experiment. We cannot prove that individuals would behave exactly in the same way outside of the laboratory, but we expect similar behavior. Our subject pool consisted mainly out of young and computer-literate college students. We believe them to be a natural target audience for the type of programs in the study. Other demographical groups are likely to demonstrate slightly different behaviors, for example, older people often report higher privacy concern and might act accordingly.

4   Conclusion

We have presented results on readability and presentation of EULAs from 50 popular free or free-to-try programs available for download on a distribution page. We suggest that the length and complexity of documents can significantly lower the notice and consent success rate achieved.[16] According to readability expert Mark Hochhauser [30], the length of legal documents often creates information overload leading to increased stress, impaired judgment and helplessness. This effect is particularly strong for older readers. Moreover, rewriting these documents in simple language is often impossible [30] and the underlying legal concepts might still be too hard to understand for interested readers [24]. All these effects appear particularly strong in EULAs since their length and the range of issues covered in them is beyond, for example, Web privacy notices. We suggested in public FTC hearings that federal authorities should revisit their basic approach to benchmarks with respect to industry self-regulation to create reliable standards for consumers to rely upon [31].

We also observed different presentation styles and variations in the timing of notice display. This is an additional source of confusion to Web users who will not expect to find important legal information, for example, only on the company's Web site or buried in a read me file. In treatments not discussed in this paper we explicitly modified the notice experience for the user so that especially designed short notices would appear either at the start or the end of the installation dialogue in addition to the long-form EULA [12].

Without significant improvements to notice and consent procedures for consumer programs it is doubtful that most consumers genuinely assent to the use of their desktops for advertisements, the installation of software with behavior that falls within the broad definition of Spyware, or limitation of usage rights. We do not expect that there exists a one-size-fits-all solution, in particular, given the increasing popularity of mobile and small-screen devices. Notice and consent involves many stakeholders. Companies are urged to improve their information dissemination practices and regulation may carefully readjust misaligned incentives in the market place. But improved notice procedure will likely result also in a more substantive obligation for users to read contractual agreements.

References

1. Good, N., Krekelberg, A.: Usability and privacy: A study of Kazaa P2P file-sharing. In: CHI 2003. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 137–144 (2003)
2. Dhamija, R., Tygar, J.D., Hearst, M.: Why Phishing Works. In: CHI 2006. Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 581–590 (2006)
3. Bederson, B.B., Lee, B., Sherman, R.M., Herrnson, P.S., Niemi, R.G.: Electronic voting system usability issues. In: CHI 2003. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 145–152 (2003)

---

[16] This also includes an informed decision not to install a particular program.

4. Whitten, A., Tygar, J.D.: Why Johnny can't encrypt: A usability evaluation of PGP 5.0. In: Proceedings of the 8th USENIX Security Symposium, pp. 169–184 (1999)
5. AOL and National Cyber Security Alliance: AOL/NCSA online safety study (December 2005), http://www.staysafeonline.info/pdf/safety_study_2005.pdf
6. Earthlink: Earthlink spy audit: Results complied from Webroot's and Earthlink's Spy Audit programs (2005), http://www.earthlink.net/about/press/pr_spyAuditReport/
7. Delio, M.: Spyware on My Machine? So What? Wired News (December 06, 2004) http://www.wired.com/news/technology/0,1282,65906,00.html
8. Good, N., Dhamija, R., Grossklags, J., Aronovitz, S., Thaw, D., Mulligan, D., Konstan, J.: Stopping Spyware at the Gate: A User Study of Privacy, Notice and Spyware. In: SOUPS 2005. Proceedings of the Symposium On Usable Privacy and Security, Pittsburgh, PA , pp. 43–52 (July 6-8, 2005)
9. Slawson, W.D.: Standard Form Contracts and Democratic Control of Law Making Power. Harvard Law Review 84, 529–566 (1971)
10. Overly, M., Kalyvas, J.R.: Software Agreements Line by Line: A Detailed Look at Software Contracts and Licenses & How to Change Them to Fit Your Needs. Aspatore Books (2004)
11. Marotta-Wurgler, F.: Competition and the quality of standard form contracts: An empirical analysis of software license agreements. New York University working paper (2005)
12. Good, N., Grossklags, J., Mulligan, D., Konstan, J.: Noticing Notice: A large-scale experiment on the timing of software license agreements. In: CHI 2007. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 607–616 (2007)
13. Kucera, K., Plaisent, M., Bernard, P., Maguiraga, L.: An empirical investigation of the prevalence of spyware in internet shareware and freeware distributions. Journal of Enterprise Information Management 18(6), 697–708 (2005)
14. Schechter, R.E.: The Unfairness of Click-On Software Licenses. Wayne Law Review 46, 1735–1803 (2000)
15. Microsoft Corporation: Privacy Guidelines for Developing Software Products and Services (October 10, 2006)
16. Casamiquela, R.J.: Contractual Assent and Enforceability in Cyberspace. Berkeley Tech. L.J. 17, 475–495 (2002)
17. Lewandowski, L.J., Codding, R.S., Kleinmann, A.E., Tucker, K.L.: Assessment of Reading Rate in Postsecondary Students. Journal of Psychoeducational Assessment 21(2), 134–144 (2003)
18. Kincaid, J., Fishburn, R., Rogers Jr., R., Chissom, B.: Derivation of New Readability Formulas for Navy Enlisted Personnel. CNTECHTRA Research Branch Report , 8–75 (1975)
19. Flesch, R.: A new readability yardstick. Journal of Applied Psychology 32, 221–233 (1948)
20. Jensen, C., Potts, C.: Privacy policies as decision-making tools: An evaluation on online privacy notices. In: CHI 2004. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 471–478 (2004)
21. Breese, P., Burman, W.: Readability of Notice of Privacy Forms Used by Major Health Care Institutions. Journal of the American Medical Association 293, 1593–1594 (2005)
22. Hochhauser, M.: Lost in the Fine Print: Readability of Financial Privacy Notices (2001), http://www.privacyrights.org/ar/GLB-Reading.htm

23. Hochhauser, M.: Readability of HIPAA Privacy Notices (2003),
    http://benefitslink.com/articles/hipaareadability.pdf
24. Masson, M.E.J., Waldron, M.A.: Comprehension of legal contracts by non-experts:
    Effectiveness of plain language redrafting. Applied Cognitive Psychology 8, 67–85
    (1994)
25. Acquisti, A., Grossklags, J.: Privacy and Rationality in Individual Decision Making.
    IEEE Security and Privacy 3(1), 26–33 (2005)
26. Vila, T., Greenstadt, R., Molnar, D.: Why We Can't Be Bothered To Read Privacy
    Policies: Models of Privacy Economics as a Lemons Market. In: Camp, L.J., Lewis,
    S. (eds.) Economics of Information Security, pp. 143–153. Springer, Heidelberg
    (2004)
27. Hillman, R.A.: Online Boilerplate: Would Mandatory Website Disclosure of E-
    Standard Terms Backfire. Michigan Law Review 104, 837–856 (2006)
28. PC Pitstop: It pays to read EULAs (2007), http://www.pcpitstop.com/spycheck/
    eula.asp
29. Kreuter, M.W., Brennan, L.K., Scharff, D.P., Lukwago, S.N.: Do nutrition label
    readers eat healthier diets? Behavioral correlates of adults' use of food labels.
    American Journal of Preventive Medicine 13(4), 277–283 (1997)
30. Hochhauser, M.: Compliance v Communication. Clarity: Journal of the Interna-
    tional Movement to simplify legal language 50, 11–19 (2003)
31. Turow, J., Hoofnagle, C., Mulligan, D., Good, N., Grossklags, J.: Consumers &
    Privacy In the Coming Decade, Session on Communicating with Consumers in the
    Next Tech-ade - The Impact of Demographics and Shifting Consumer Attitudes.
    In: Public Hearings on Protecting Consumers in the Next Tech-ade, Federal Trade
    Commission, Washington D.C (November 6-8, 2006)

What Instills Trust? A Qualitative Study of Phishing

Markus Jakobsson[1], Alex Tsow[2], Ankur Shah[2], Eli Blevis[2],
and Youn-Kyung Lim[2]

[1] Indiana University, Bloomington and RavenWhite Inc.
`markus@indiana.edu, markus@ravenwhite.com`
[2] Indiana University, Bloomington
`{shahak,atsow,eblevis,younlim}@indiana.edu`

Abstract. This paper reports the highlights of a user study which gauges reactions to a variety of common "trust indicators" – such as logos, third party endorsements, and padlock icons – over a selection of authentic and phishing stimuli. In the course of the *think-aloud* protocol, participants revealed different sensitivities to email messages and web pages. Our principal result is the analysis of what makes phishing emails and web pages appear authentic. This is not only of interest from a pure scientific point of view, but can also guide the design of legitimate material to avoid unnecessary risks. A second result of ours are observations of what makes legitimate content appear dubious to consumers. This is a result with obvious applications to online advertising.

Keywords: authenticity, design, email, experiment, phishing, psychology, stimuli, think-aloud, user interface design, web pages.

1 Introduction

Over the last few years, the problem of phishing has grown at an alarming rate. As service providers are becoming increasingly aware of the threat, more and more effort is spent on countermeasures—whether technical, educational or legal. In order to keep pace with such countermeasures, and with increasing competition between groups, phishers are also becoming more sophisticated. There are both social and technical examples of this trend—better spelling, use of subdomains and cousin domains to deceive users, and improved psychological design of the request all fall within the former class. The use of DNS modifications to avoid takedown, and of keyboard loggers to capture information belongs to the latter. With research in social aspects of phishing—the deceit component—lagging behind efforts dealing with technical aspects, the next wave in phishing may very well rely on an attack that has a better deceptive component. In order to understand what direction such an attack may take, we study the role of authenticity in phishing.

We performed an experiment to determine what aspects of email and web pages effectively convey authenticity to visitors. Users are presented with a collection of live, but locally hosted, web pages and email messages. Subjects followed

---

a think-aloud protocol, verbalizing sources of doubt, concern, confidence, and confusion. Participants interacted with web pages using actual client software, however all hyperlinks were link redirected to a page asking them to describe how they expected the result to influence their evaluation.

While our experiment does *not* answer the question of the real impact of these indicators of trust, our experiment *does* indicate what typical computer users are able to detect when they are carefully watching for signs of phishing. This *security first* approach approximates a lower bound on vulnerability to phishing attacks.

The deceit component of phishing does not take advantage of technical vulnerabilities, nor can it always be properly addressed by technical countermeasures. As an example of this, it has been shown by Jakobsson and Ratkiewicz [6] that the use of IP addresses as the domains of phishing web pages is considered legitimate by a smaller portion of users than the use of relevant subdomains in conjunction with non-descriptive or related domains.

Dhamija, Tygar and Hearst [2] studied how computer users fall victim to phishing attacks based on a lack of understanding of how computer systems work; due to lack of attention; and because of visual deception practiced by the phishers. They also adopt the security first assumption, asking participants to evaluate a collection of web pages.

Downs, Holbrook, and Cranor [3] have performed an assessment of risk familiarity and phishing vulnerability in a lab setting. Their participants are given fictional identities to role play a suite of web and email interactions – a method focusing on task completion rather than security. They find that subjects are good at protecting themselves against known scams, but have difficulty adapting to new tactics.

In a study by Whalen and Inkpen [7], the authors used eye-tracking and surveys to determine that people see the lock icon but rarely interact with it. This indicates a vulnerability to forged lock icons, and potentially other trust indicators as well. Friedman et al. [4] studied user perceptions of secure connections and determined that the presence of a lock or key icon was the primary perceived indicator of "correct" evaluations of security, distantly followed by the presence of an "https" designation in the URL, and the type of information requested.

2   Experimental Design

Subject selection. Subjects were chosen both among college students and university staff and faculty. A total of 17 subjects participated in the study. Ages ranged from 18 to approximately 60. Computer science students and staff/faculty with computer science backgrounds were excluded, as was anybody who had taken one or more computer security classes.

Stimulus design. Subjects were shown both email stimuli and web page stimuli, some of which modeled phishing attempts while others represented authentic media. Most of the stimuli were based on actual emails or web pages, whether authentic or not; others were designed by the researchers. Minor modifications

were made to email stimuli: All emails were modified (or designed) to have the same apparent recipient.

A number of features were tested: legitimate endorsement logos (Verisign, BBB, and TrustE), made-up endorsements, cousin domain names, naked IP addresses, padlocks in various locations (favicon, content body, browser frame), spelling and grammatical irregularities, `https` and `http` hyperlinks, and personalization (salutations and account data).

Procedural overview. Participants sat in front the computer and display. The subjects were asked to rate 26 stimuli each stimulus in terms of their perceived phishiness/authenticity. The rating was done using a 5-point Likert scale [8]: *certainly phishing*, *probably phishing*, *no opinion*, *probably not phishing*, and *certainly not phishing*. Subjects were allowed to scroll stimuli windows up and down and perform mouse-over of hyperlinks, but *not* allowed to click on any hyperlinks. We required classification of a given stimulus before presenting the next one. As subjects observed and judged the stimuli, they were asked to verbalize their thoughts – whether relevant to the decision or not. Voice and screen actions were recorded using *Camtasia Studio* [1]. After completing the stimuli evaluation, participants discussed three questions in an exit interview: *Do you think you have been fooled by any of these stimuli in your daily computer usage?*, *What stimulus features inspired confidence in authenticity?*, and *What stimulus features generated suspicion in authenticity?*

3   Summary of Findings

The think-aloud protocol recorded several insightful comments:

- "If the URL looks hinky, I'm not going to trust it."
- "Well I hate to see a statement says 'this message is all authentic'."
- "When I see Verisign, I would probably go with it and say that it's certainly not phishing,"
- "There's no copyright thing which is usually there on other banks."
- "Probably any of these emails I got I would have just deleted ... I wouldn't read anything that looks not important to me."

The full version of this paper showing all the stimuli is available at: *http://www.indiana.edu/ phishing/papers/JakEtal.pdf*

When reading the conclusions, it is important to realize that these were made in a security first context; therefore, they describe the *abilities* of the subjects rather than the *habits* of the subjects.

The conclusions from our analysis are:

1. **Document layout matters.** Security awareness was generally well received when confined to a specific portion of a web page. The Chase *Security Center Highlights* and USBank *Online Security* area both use high resolution

padlocks to draw visitor attention; these two frequently evoked positive reactions from subjects. Some legitimate providers (such as Keybank) were given a low rating due to "unprofessional design." In the case of Keybank, subjects cited the absence of the institutional name on the login-page, along with the fact that the fields for user name and password were of different length. The presence of copyright information and legal disclaimers, typically at the bottom of the stimulus in small print, enhanced trust for many subjects.

2. **Too much emphasis on security can backfire.** Some stimuli, in particular the (legitimate) IUCU (Indiana University Credit Union) website with its blinking phishing banner, were criticized for their overwrought concerns about online security. Subjects did not like that the IUCU website said "phishing attack in progress" in three different locations. Some commented that "phishing" is too obscure a term for a financial institution to use in their communications – the phrase "identity theft" was offered as a plausible substitute. Explicit assertions of safety, such as "This message is authentic," and "Phishing Protected," sounded implausible to many subjects.

3. **People look at URLs.** It was found that subjects looked carefully at URLs of web pages, and on the URLs obtained by mouse-over in emails. Subjects were good at detecting IP addresses as being illegitimate, but were not highly suspicious of URLs that were well-formed, such as `www.chase-alerts.com`. On the other hand, subjects were good at detecting syntactically peculiar addresses, such as `www-chase.com`.

4. **Third party endorsements depend on brand recognition.** The stimuli deployed a range of third party endorsements, from well established brands like Verisign to made-up endorsements like *Safe Site*. We found that endorsements from Verisign were taken with the most gravity. Almost every subject mentioned Verisign by name as a positive factor in their trust evaluation. BBBOnLine and TRUST-e endorsements had no significant effect. On the other hand, the three made-up endorsements evoked consistent criticism.

   Some subjects noticed third party endorsements on stimuli they clearly believed to be phishing, and deduced that the graphics could be rendered on any page. One subject observed "Probably now that I see all these [stimuli], I should not believe in Verisign," but later dismissed a web page because "it's not Verisign protected, but it says something which I've never seen, 'TRUST-e'. I don't know, so probably I wouldn't go in this account."

5. **People judge relevance before authenticity.** Subjects often decided whether a stimulus was legitimate or not based on the content, as opposed to the signs of authenticity. In particular, any stimuli that offered a monetary reward was considered phishy, independently of whether it was authentic or not. Likewise, emails that requested passwords up-front were considered phishy, whereas emails that only appeared to contain information were considered safe. This is a problem, as users could be drawn to a site by an email that appears to be for information only, and once at the site asked for credentials.

6. **Faux-personalization creates trust.** Personalization, even with well-known data, increases the trustworthiness of stimuli, whether email or web pages. One subject said that presentation of ZIP code and mother's maiden name would enhance trust in an email message. Yet, this data could be gathered by an attacker using IP to zip code mapping software and publicly available databases [5]. Some expressed comfort with an email stating it was intended for a user whose account starts with 4546; account number prefixes are easily guessed because they come from a small pool of possibilities when the institution is known. While many subjects insisted that the last four digits are a more trustworthy marker, they did not explicitly penalize the message for using the first four digits.

7. **Emails are very phishy, web pages a bit, phone calls are not.** Overall, email stimuli were considered more phishy than web stimuli to participants in the study. Many subjects said that following links from email was a risky activity, and consciously avoid the practice. Since very few admit to following links given in phishy emails, their exposure to phishy web pages is inherently more limited. Many participants said that they would try to independently verify email contents by calling the institution directly. Few participants specified how they would obtain the correct phone number and therefore could expose themselves to fraudulent customer service numbers; most systems prompt users to dial in their account number and zip code prior to speaking with a representative. Several participants also said that email is an inappropriate alert medium for urgent matters, such as password changes and account lock-outs, and expected a phone call from the institution.

8. **Padlock icons have limited direct effects.** Large padlock graphics were effective at drawing attention to specific portions of the stimulus. By themselves, they did not cause any subject to express an improvement in trust. Small padlock icons in the content body were never commented on by subjects. Their ineffectiveness was supported by the nearly identical rating distributions of two Chase web pages that differ only by the presence of the SSL-post padlock icon in the login area. The SSL padlock at the bottom of browser frame enhanced trust in many subjects, however two subjects lost trust when mouse-over revealed a made-up certification authority, *Trust Inc.*

9. **Independent channels create trust.** If a stimulus suggested that the subject could call to verify the authenticity of the email or web page, then the very existence of this possibility strengthened the trust the subjects had in this stimuli. Subjects stated that they would not call the number to verify the authenticity, but *someone else would.*

Acknowledgments

References

1. Camtasia Studio Screen Recorder for Demos, Presentations and Training, Camtasia Studio 4, TechSmith Corportation (2006),
   http://www.techsmith.com/camtasia.asp
2. Dhamija, R., Tygar, J.D., Hearst, M.: Why phishing works. In: CHI 2006. Proceedings of the SIGCHI conference on Human Factors in computing systems, pp. 581–590. ACM Press, New York (2006)
3. Downs, J.S., Holbrook, M.B., Cranor, L.F.: Decision strategies and susceptibility to phishing. In: SOUPS 2006. Proceedings of the second symposium on Usable privacy and security, pp. 79–90. ACM Press, New York (2006)
4. Friedman, B., Hurley, D., Howe, D.C., Felten, E., Nissenbaum, H.: Users' conceptions of web security: A comparative study. In: CHI 2002. extended abstracts on Human factors in computing systems, pp. 746–747. ACM Press, New York (2002)
5. Griffith, V., Jakobsson, M.: Messin' with Texas, Deriving Mother's Maiden Names Using Public Records. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, Springer, Heidelberg (2005)
6. Jakobsson, M., Ratkiewicz, J.: Designing Ethical Phishing Experiments: A Study of (ROT13) rOnl Query Features. In: WWW 2006 (2006)
7. Whalen, T., Inkpen, K.M.: Gathering evidence: Use of visual security cues in web browsers. In: Graphics Interface 2005. ACM International Conference Proceeding Series, vol. 112. pp. 137–144. Canadian Human-Computer Communications Society, Waterloo (2005)
8. Likert, Rensis.: A technique for the measurement of attitudes. Archives of Psychology 140 (June 1932)

Phishing IQ Tests Measure Fear, Not Ability

Vivek Anandpara, Andrew Dingman, Markus Jakobsson, Debin Liu,
and Heather Roinestad

Abstract. We argue that phishing IQ tests fail to measure susceptibility to phishing attacks. We conducted a study where 40 subjects were asked to answer a selection of questions from existing phishing IQ tests in which we varied the portion (from 25% to 100%) of the questions that corresponded to phishing emails. We did not find any correlation between the *actual* number of phishing emails and the number of emails that the subjects indicated were phishing. Therefore, the tests did not measure the ability of the subjects. To further confirm this, we exposed all the subjects to existing phishing education after they had taken the test, after which each subject was asked to take a second phishing test, with the same design as the first one, but with different questions. The number of stimuli that were indicated as being phishing in the second test was, again, independent of the *actual* number of phishing stimuli in the test. However, a substantially larger portion of stimuli was indicated as being phishing in the second test, suggesting that the only measurable effect of the phishing education (from the point of view of the phishing IQ test) was an increased concern—not an increased ability.

Keywords: phishing, phishing education, phishing IQ test.

1 Introduction

Popular media routinely covers the mounting problem of phishing. Financial institutions frequently alert clients of the risks of identity theft, and many provide detailed descriptions of common attacks and how to avoid falling victim to these. With this popular focus on the problem, we must ask ourselves why the recent trends show an increase in the number of people that fall victim to phishing. Furthermore, we must pose the question whether current educational efforts are meaningful and whether ways in which vulnerabilities are assessed work.

To be able to ask these questions, it is important first to understand why phishing works. This question has been asked by several researchers recently [5,6,7,10,21], and a collection of insightful conclusions have been found. One reason that phishing works is that most people do not have a detailed understanding of all the guises a given threat might take, but only react to situations that he or she has already identified as being dangerous. Another reason is that many users do not possess the technical sophistication sufficient to verify whether a given email or webpage corresponds to an attempt to defraud them. The most important reason of all, though, might be that to most people, security is a *secondary* goal. In other words, the average person may very well ignore signs of risk, since he or she is not actively looking for these.

There are publicly available 'phishing IQ tests' published to help individuals assess their likely vulnerability to phishing scams. Examples of these can be found at Mailfrontier/Sonicwall [12,15]; Mailfrontier also has UK and German versions [14,13]. These tests typically take the form of a sequence of e-mail screen shots depicting messages of the sort phishers tend to emulate. Users identify the depicted message as either a legitimate message or a phishing scam, and receive a score based on the percentage of correct answers. We argue that such phishing IQ tests are flawed on several levels. Due to their delivery format, a static file, many actual security indicators are not available. By their nature, these tests also lack context present in real attacks and which can aid in making accurate decisions. In addition, and more importantly to our study, while the natural context is lacking, an artificial context may skew the test taker's judgment. This may create a false sense of security among test takers who are receive a high score on the tests. As we will show, *obtaining a high score is not an indication of ability to recognize phishing attempts.*

Since test takers know that they are being tested on their ability to identify phishing emails, test takers may be suspicious beyond the level they normally would upon seeing the original email in their inbox. This could mean that they correctly label some examples as phishing that they might normally be susceptible to, and/or would incorrectly identify legitimate emails as phishing. This is a well understood fact, but does not in any way affect our methodology. Quite to the contrary, we are to some extent able to quantify the effects of this type of bias; this is done by comparing the average ratings given by subjects before and after the educational step of the experiment.

We are also able to show that traditional forms of education [8] increase the level of fear or concern among users, but that they became no better at passing the phishing IQ tests.

2   Previous Work

Much of the existing literature related to phishing deals with people's perception of website credibility and not with how people judge the emails which lure them to the fake websites. There has been a substantial amount of work in the direction of what makes a website credible or phishy to the people.

Fogg et al [6,7] conducted a study with over 2500 subjects and investigated how different elements of websites affect people's perception of credibility and laid down guidelines for the credible perception of websites. On similar lines, Dhamija et al [5] worked towards establishing what makes phishing websites credible and why phishing works. In another user study which dealt with the effectiveness of anti-phishing measures such as phishing toolbars, Wu et al [21]. examined the impact of anti-phishing toolbars in preventing phishing attacks.

Several other studies have recently shed light on the problem of phishing [1,5,9,11,17,10] and several have proposed countermeasures [2,4,11,16,17,18,21]. Moreover, researchers have become increasingly interested in the role of malware in the context of phishing [3,19,20].

Many of these papers fail to recognize that although it is not commonly happening today, the various indicators of security which are emphasized can be spoofed by phishers (see, e.g., [5]). Thus, sophisticated phishing attacks may be difficult to detect, even to people who are reasonably aware of what to look for. It is also worth mentioning that the multitude of studies which perform general evaluations of phishing vulnerabilities largely neglect the subject-expectancy effect.The subject-expectancy effect is a cognitive bias that occurs when a subject expects a given result and therefore unconsciously manipulates an experiment or reports the expected result, partly to avoid embarrassment. This effect is applicable to phishing IQ tests (e.g., [12]), where the cognizance of being involved in a phishing IQ test makes the subjects unusually suspicious and this can substantially skew the results of the test.

There have been no previous attempts to gather any empirical data on the effectiveness of these phishing IQ tests.

3   Experiment and Results

We performed a test on 40 subjects for this data analysis. As our study asks about how an average user performs at the task, we excluded subjects with an unusual knowledge of computer science or security, and asked that subjects be people who either use or would consider using online shopping, banking, or bill paying.

A first phishing IQ test. For the first part of the experiment, we gave subjects a short sample test with five screenshots in which we varied the number of those screenshots corresponding to actual phishing emails. Our hypothesis predicted that the portion of screenshots subjects labeled as suspicious would be roughly the same regardless of the group of the subject (i.e., independent of how many screenshots *were* "bad").

Phishing education. After the first sample test, we gave the subjects time to read over an example of phishing education intended for the layman. We used the education material available at ftc.gov [8]

A second phishing IQ test, and analysis. After the educational step, a second phishing IQ test was administered. This contained 5 stimuli diferent from what the subject saw in the first phishing IQ test.

Figure 1 shows, among other things, how many of the examples they were shown that subjects labeled as phishing, and the number required to get a perfect score. It can be seen that the experiments support the hypotheses described above.

Consistent with our first hypothesis, the number of times subjects labeled an example as phishing appears to have no correlation with the number of actual phishing examples they were shown. The individual responses have a practically zero linear correlation coeffecient with the number of actual phishing examples, suggesting that the number of times a subject labels an example as phishing does not depend on the number that actually *are* phishing.
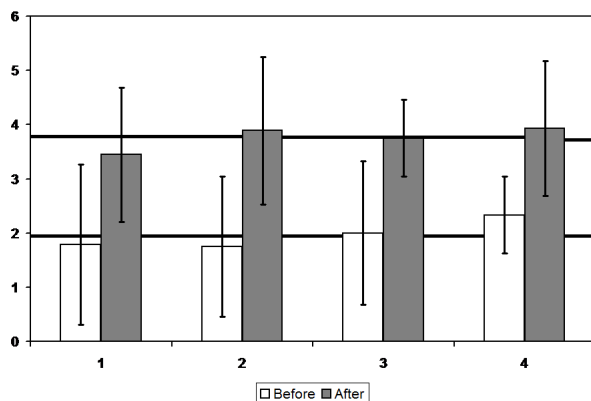
Fig. 1. Subject responses before and after education. On the x-axis, the number of stimuli that were *actually* phishy are shown; the diagram reads the number of stimuli that subjects indicated as phishy. The left bars show the experimental results of the first phishing IQ test, whereas the right bars show the results of the second test. The standard deviation is indicated for each measurement. For each pair of bars, we include a horizontal line showing what the results should have been if subjects were truly able to distinguish authentic stimuli from phishing attempts. It is evident that this is not the case. Note also the effect of the education: Subjects were not better at taking the second test than the first; however, they were more suspicious of all stimuli shown during the second test and hence falsely labeled many legitimate stimuli as phish.

Figure 1 also shows how many of the stimuli subjects labeled as phishing in the post-education test, compared to the actual number shown. This time, we see the overestimation we predicted in our hypothesis; after reading about how to identify phishing, subjects started seeing more instances of phishing than were necessarily there.

In the short term at least, the education does appear to affect the subjects' judgment, but more in terms of making them more suspicious than in improving their ability to distinguish phishing from legitimate emails. The number of times that subjects labeled a stimulus as phishing increased from the first to the second test for most subjects, even though two of the four groups were actually shown fewer instances in the second round than in the first; however, the number of instances labeled as phishing still did not correlate with the number which were phishing.

References

1. Anti-Phishing Working Group: Phishing Activity Trends Report November 2005 (2005)
2. Chou, N., Ledesma, R., Teraguchi, Y., Boneh, D., Mitchell, J.C.: Client Side Defense Against Web-based Identity Theft, http://crypto.stanford.edu/SpoofGuard/#publications
3. The Crimeware Landscape: Malware, Phishing, Identity Theft and Beyond, http://www.antiphishing.org/reports/APWG_CrimewareReport.pdf

4. Dhamija, R., Tygar, J.D.: The Battle Against Phishing: Dynamic Security Skins, In: Proc. SOUPS (2005)
5. Dhamija, R., Tygar, J.D., Hearst, M.: Why Phishing Works. In: CHI 2006. Proceedings of the Conference on Human Factors in Computing Systems (to appear)
6. Fogg, B.J., Soohoo, C., Danielson, D.R., Marable, L., Stanford, J., Tauber, E.R.: How Do Users Evaluate the Credibility of Web Sites?: A Study with Over 2,500 Participants. In: Proc. DUX (2003)
7. Fogg, B.J., Marshall, J., Laraki, O., Osipovich, A., Varma, C., Fang, N., Paul, J., Rangnekar, A., Shon, J., Swani, P., Treinen, M.: What Makes Web Sites Credible?: A Report on a Large Quantitative Study. In: Proc. CHI, pp. 61–68 (2001)
8. FTC.gov Alert, http://www.ftc.gov/bcp/edu/pubs/consumer/alerts/alt127.htm
9. Jakobsson, M.: Modeling and Preventing Phishing Attacks. In: Patrick, A.S., Yung, M. (eds.) FC 2005. LNCS, vol. 3570, Springer, Heidelberg (2005)
10. Jakobsson, M.: The Human Factor in Phishing. Privacy & Security of Consumer Information (2007)
11. Jakobsson, M., Myers, S.A. (eds.): Phishing and Countermeasures: Understanding the Increasing Problem of Electronic Identity Theft, p. 739 (December 2006) ISBN 0-471-78245-9
12. MailFrontier Phishing IQ Test II, http://survey.mailfrontier.com/forms/msft_iq_test.html
13. MailFrontier Phishing IQ Test – Deutsche Edition, http://german.mailfrontier.com/survey/phishing_de.jsp
14. MailFrontier Phishing IQ Test – UK Edition, http://survey.mailfrontier.com/survey/phishing_uk.html
15. MailFrontier/Sonicwall Phishing, http://www.sonicwall.com/phishing/
16. PassMark Security: Protecting Your Customers from Phishing Attacks - An Introduction to PassMarks, http://www.passmarksecurity.com/
17. The Phishing Guide - Understanding & Preventing Phishing Attacks, http://www.ngssoftware.com/papers/NISR-WP-Phishing.pdf
18. RSA Security: Protecting Against Phishing by Implementing Strong Two-Factor Authentication (2004), https://www.rsasecurity.com/products/securid/whitepapers/ PHISH_WP_0904.pdf
19. Stamm, S., Ramzan, Z., Jakobsson, M.: Drive-By Pharming. Technical Report TR641, Indiana University (December 2006)
20. Tsow, A., Jakobsson, M., Yang, L., Wetzel, S.: Warkitting: the Drive-by Subversion of Wireless Home Routers. Anti-Phishing and Online Fraud, Part II. Journal of Digital Forensic Practice (Special Issue) 1(3) (November 2006)
21. Wu, M., Miller, R., Garfinkel, S.: Do Security Toolbars Actually Prevent Phishing Attacks? In: Proc. CHI (2006)

Mental Models of Security Risks

Farzaneh Asgharpour, Debin Liu, and L. Jean Camp

School of Informatics, Indiana University
{fasgharp, deliu, ljcamp}@indiana.edu

Abstract. In computer security, risk communication refers to informing computer users about the likelihood and magnitude of a threat. Efficacy of risk communication depends not only on the nature of the risk, but also on the alignment between the conceptual model embedded in the risk communication and the user's mental model of the risk. The gap between the mental models of security experts and non-experts could lead to ineffective risk communication. Our research shows that for a variety of the security risks self-identified security experts and non-experts have different mental models. We propose that the design of the risk communication methods should be based on the non-expert mental models.

Keywords: Mental model, Card sorting, risk communication.

1 Introduction

The mental models approach to risk communication is a method based on the conceptual models of recipients of the communication. A mental model is an internal conception for how something works in the real world [1]. This notion can be very case specific and is subject to change due to experience, stigmatization, perception, and problem-solving strategies.

The mental models approach in risk communication has effectively been used to enhance environmental [2] as well as medical [3] risk communication. While a mental models approach has been used to examine privacy perspectives [4] it has not been introduced to information security. This work is grounded in mental models as it has been developed in environmental risk communication. The goal of mental models in environmental research is to enhance risk communication about household toxics [1]. Like computer security, environmental risks can be much more problematic at home than at the work place. For instance, paint stripper and other chemical hazards are, like computers, more easily regulated in the work place than home.

As mental models have not been investigated in security, we begin with a quantitative approach to evaluate the five mental models introduced in security literature: physical security, medical infections, criminal behavior, warfare and economic failure [5].

Risk communication typically consists of a message formulated by security experts to warn a community of non-experts against a set of threats. The difference between the mental model of the experts and non-experts with regard to the risk

---

can decrease the efficacy of the risk communication. This difference is often a consequence of two different levels of knowledge about the subject matter. One may think that since the experts have access to the technical definition of the risks, know the catalysts and understand the consequences of each threat, their mental model is more reliable for designing risk communication instruments. The key point is that the purpose of risk communication is not conveying the perfect "truth" to the users, but rather prompting them to take an appropriate action to defend their system against a certain danger. Even though mitigation of a specific risk requires knowledge of the nature of the risk, efficacy of the risk communication requires the experts to understand their target group.

In this work, we define a distance measure between each mental model and security risk. Using our proposed measure we estimate the mental models of the security experts' and non-experts' with regard to each security risk. The details of our experiment design are explained in Section 2. Section 3 covers the data analysis. Section 4 concludes the paper.

2   Experiment Design

Due to the complexity of human knowledge acquisition and psychology, discovering mental models is normally a very difficult task to achieve. This task could be done using different elicitation techniques such as Teachback Interviews, Repertory Grid, Goal-Oriented Approach, and Card Sort [6]. Card sorting [7], is a structured elicitation technique done by having a subject sorting a pile of cards, with some specific items typed on them, into different piles. There are two kinds of card sort: closed and open. In closed card sort one must choose the label of each card from a group of given labels. In open card sort no labels are given and one can sort the words into arbitrary piles according to one's perception. Considering the five mental models enumerated in Section 1, we applied a closed card sort experiment to estimate the mental models of lay users and experts with regard to various security risks. The benefit of card sort technique is that it is easy and natural to perform for people.

We considered two levels of expertise in security: expert (E) and non-expert (NE). By E we mean someone who knows all the technical definitions of the security-related words. We defined NE as someone who does not know the technical definition of security terms and at most knows some practical aspects of the risks. If the set $\mathbf{R} = \{r_1, r_2, \cdots, r_n\}$ presents all the security risks given in our experiment, the main purpose of the experiment is to estimate and compare the experts' and non-experts' mental models for each member of $\mathbf{R}$. To classify our participants as experts and non-experts we provided the definition of the expert and non-expert in the instruction section of the experiment and asked the participants to declare their level of expertise. We recruited 74 participants, consisting of faculty and students of various disciplines. These participants had varying levels of knowledge in computer security. Out of 74 participants, 25 were self-declared experts and 49 were self-declared non-experts. The participants were from 18 to 50 years old.

We provided a set of 66 words (Appendix-Table 1) to the participants and asked them to cluster the words into groups by marking similar words with the same colors. Appendix-Figure 1 shows a snapshot of the card sorting experiment.

The wordlist (Appendix-Table 1) contained the name of various risks, some common security related words and some words directly related to each of the following mental models: physical security, medical infections, criminal behavior, economic failure, and warfare. The words related to each mental model are all driven from Webster's Thesaurus. Given that there is always the possibility that a certain word might be unknown to the participant, we also specified a color for the words which might not be familiar to the participant. Finally, to leave enough room for other possible mental models, we gave one color for words which in a participant's view might not belong to any of the above categories.

We used the following correspondence between mental models and colors: physical security:green, medical infection:blue, criminal behavior:orange, warfare:red, economic failure:yellow. Gray was the color for words which, according to the participants' perception, did not match with any of the above mental models. Finally, we provided the color purple for the words not familiar to the participant. To be able to keep track of the participants' mental models and to maintain consistency in associating colors with different mental models, we provided instructions on how to associate colors with words. Due to various cultural color interpretations, we decided not to follow any specific pattern in associating colors, as for instance color green is associated with peace by some people and with the environment by others. The nonintuitive and arbitrary color selection made the participants refer to the instructions more frequently and therefore to be more careful in assigning colors to words.

We used macromedia Flash and PHP to present the card sorting experiment as an online experiment.

3   Data Analysis

Multidimensional scaling (MDS) method [8] is used to find structure in a set of distance, or dissimilarity, measures between objects. MDS assigns objects to specific points in a conceptual space such that the distances between points in the space match the given dissimilarities as closely as possible. Since MDS considers either relative distance or similarity between the observations, one can equally map the observations either using a similarity or a distance matrix. Applying this method we can map the words into a two dimensional space and then, considering relative distances between the words, assign mental models to each security risk.

To apply the MDS method, the original data were first tabulated and interpreted as proposed in [6]. Every time a participant marked a pair of words with the same color, we counted that as a vote for similarity between the two words. Therefore, as an example, if most of the participants marked the words "trade" and "stock" with the same color, then we say these two words are highly similar in people's perception. In contrast, if only a few participants assigned the words

"war" and "fever" with the same color, we interpret this result as these two words are not very similar. This way, we had two 66×66 matrices, one for experts and one for non-experts. We name these two matrices as Expert's Choice Matrix and Non-expert's Choice Matrix, and show them by respectively ECM and NCM. In order to reveal underlying perceptual dimensions that participants use to distinguish among these words, we present the symmetric matrix via multidimensional scaling map and locate the expert's and non-expert's choice matrices into a two dimensional space.

Before applying the MDS method to map words in a two dimensional space, we define a function to measure the distance between each pair of words in our wordlist. We also use this function to measure the distance between security risks and mental models and finally to assign a mental model to each risk. Considering matrix ECM, the distance $d_E(w_i, w_j)$ between two words w_i and w_j is defined as

$$d_E(w_i, w_j) = 1 - \frac{\mathsf{ECM}_{i,j}}{n} \tag{1}$$

in which, n is the number of data entries in the expert data set and $\mathsf{ECM}_{i,j}$ is the element of the matrix ECM located in the row i and the column j. The non-expert distance between the two words w_i and w_j, $d_{NE}(w_i, w_j)$, is defined similarly. One can easily prove that d_E and d_{NE} are distance measures.

Having all the above distances calculated, we replaced the elements of the two matrices with the corresponding distance matrices. In Appendix, Figures 2 and 3 show the map of the multidimensional scaling of the distance matrices ECM and NCM.

For each risk r we have five distances, one for each mental model. The corresponding mental model of r is the one with the smallest distance from r.

In Appendix, Table 1 shows a list of three words under each mental model marked as Obvious Words. For each of these words around 75% of our participants have grouped the word with the other related words. We call these words *obvious words* and refer to each set of obvious words under a certain mental model as an *obvious mental model*. For a given risk r, we define the expert-distance between r and an obvious mental model $M = \{w_1, w_2, w_3\}$ as

$$D_E(M, r) = \frac{1}{3} \sum_{1 \leqslant i \leqslant 3} d_E(w_i, r) \tag{2}$$

where $d_E(w_i, r)$ is the expert-distance between w_i and r. Similarly one can define $D_{NE}(M, r)$ as the non-expert-distance between the mental model M and the risk r. To each risk r we assign at least one expert, EM_r, and one non-expert, NM_r, mental model according to the previous definition. EM_r (NM_r) is the mental model corresponding to the obvious mental model with minimum expert-distance (non-expert-distance) from r.

In Appendix, Tables 2 and 3 show respectively the distance between each risk and each mental model. These tables are the basis for our conclusions. Appendix-Table 4, shows the expert and non-expert mental models indicated for each risk.

As shown in Appendix-Table 4, some of the probabilities are very low. The reason is that in average 34.3% of non-experts, and 40.5% of the expert participants considered an arbitrary mental model, other than our five suggested mental models. In other words, more than one third of the participants found computer security risks were not consistent with the mental models the previous work had found in the computer security literature. This implies that naive users and computer security experts may be even further apart than suggested by this work. This fact also suggests the need for qualitative study, such as interviews, to find other possible mental models.

Our methodology shows that for 10 out of 29 risks, E and NE have different mental models (risks corresponding to the checked boxes in the last column of the Appendix-Table 4). We are currently studying if changing the definition of the expert to the "security specialists who have been either teaching or studying in computer security for at least 5 years" changes our final results.

One can also see that the medical mental model is chosen by experts four times whereas just once by the non-experts. On the other hand physical security is selected 7 times by the non-experts but only 4 times by experts. This suggests that the medical mental model is not an appropriate mental model communicating non-experts. However, physical security could be an appropriate mental model for this purpose.

4   Conclusion

This paper reports an initial experiment to verify the mental models of the security experts and non-experts with regard to security risks. Previously these models had been implicit in security risk communication. This work uses card sorting to test both the similarity between experts and non-experts and the coverage of those implicit mental models. Our experiment illustrates that for 70% of the security risks non-experts have either physical security or criminal mental model. We also show that computer security risks are more distant from medical threats for non-experts than for experts.

Our study suggests that none of the mental models implicit in the security literature fit the understanding or the impression of the related risk. We propose that the efficacy of the security risk communication could be increased by adjusting the risk communications with the mental models of non-expert community.

Further research includes conducting qualitative interviews to extract mental models of security risks. Also, narrowing down the definition of the expert to "security specialists" one can repeat the experiment and estimate the mental model of each group. We expect to receive even more difference between the security specialists' and non-experts' mental models.

References

1. Morgan, M.G., Fischhoff, B., Bostrom, A., Atman, C.J.: Risk Communication: A Mental Models Approach. Cambridge University Press, Cambridge (2001)
2. Ronnfeldt, C.F.: Three Generations of Environment and Security. Journal of Peace Research 34(4), 473–482 (1997)
3. Jungermann, H., Schutz, H., Thuring, M.: Mental models in risk assessment: Informing people about drugs. In: Risk Analysis, Blackwell, Oxford (1981)
4. Kumaraguru, P., Cranor, L.F., Newton, E.: Privacy Perceptions in India and the United States: An Interview Study. In: 33rd Research Conference on Communication, The National Center for Technology & Law, George Mason University School of Law, USA (September 2005)
5. Camp, L.J.: Mental Models of Security. IEEE Computer Society Press, Los Alamitos (2006)
6. Byrd, T.A., Cossick, K.L., Zumd, R.W.: A Synthesis of Research on Requirements Analysis and Knowledge Acquisition Techniques. MIS Quarterly 16(1), 117–138 (1992)
7. Hudson, W.: Playing Your Cards Right, Getting the Most from Card Sorting for Navigation Design. HCI & Higher Education Column: People: HCI & the web 12(5), 56–58 (2005)
8. Kruskal, J., Wish, M.: Multidimensional Scaling. Sage Publication, Thousand Oaks (1978)

Appendix: Tables and Figures
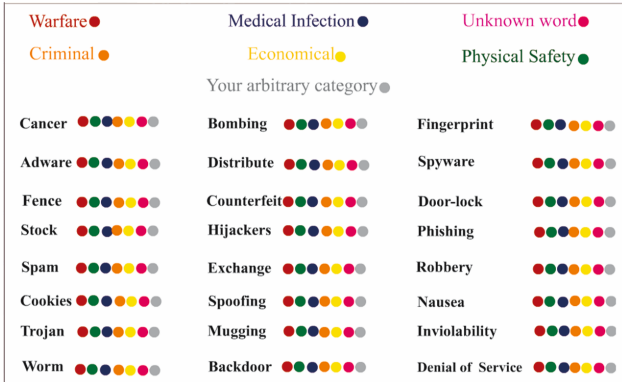
Fig. 1. Snapshot of the Card Sorting Experiment

Table 1. List of words given in the card sorting experiment (the first three words under each mental model are the *Obvious Words*)

Crime	Medical	Physical	Warfare	Economic	Security
Theft	*Epidemic*	*Fence*	*War*	*Trade*	Identity theft
Housebreaking	*Fever*	*Door-lock*	*Bombing*	*Export*	Hijackers
Kidnapping	*Illness*	*Shield*	*Destroy*	*Stock*	Cookies
Fingerprint	Cancer	Inviolability	Terror	Distribute	Adware
Counterfeit	Detoxification	Invulnerability	Attack	Exchange	Spyware
Robbery	Nausea		Suicide	Endorse	Phishing
Mugging	Inflammation			Advertise	Spam
Vandalism	Contagious			Risk	DoS attack
Injection	Sore				Drive-by-download
					Trojan
					Keystroke logger
					Junk mail
					Virus
					Worm
					Hacking
					Binder
					Exploit
					Zombie
					Authentication
					Click fraud
					Password
					UserID
					Firewall
					Backdoor
					Blacklist
					Spoofing
					Dropper
					Address book
					Honeypot

Table 2. Expert distances $(D_E(M, r))$ between security risks and mental models

Security Risk	Criminal	Physical	Medical	Economic	Warfare
Adware	0.9333	0.9600	0.9600	0.6267	0.9600
Spyware	0.6533	0.9600	0.9333	0.8933	1.0000
Phishing	0.7075	0.8435	0.9592	0.8844	0.8912
Identity theft	0.3467	0.9467	0.9333	0.9733	0.9467
Spam	0.8933	0.9733	0.9733	0.7067	0.9200
Hijackers	0.4133	1.0000	0.9467	1.0000	0.7600
Cookies	0.9333	0.9467	0.9067	0.7067	0.9733
DoS attack	0.7200	0.9200	0.9067	0.9067	0.7867
Drive-by-download	0.7733	0.9467	0.9733	0.8267	0.9200
Trojan	0.7467	0.9733	0.8400	0.9067	0.7867
Keystroke logger	0.7333	0.8400	0.9200	0.8800	0.9733
Junk mail	0.9200	0.9600	1.0000	0.6133	0.9067
Virus	0.8800	0.9733	0.5733	0.8933	0.8667
Worm	0.8267	0.9333	0.8133	0.8933	0.9067
Hacking	0.6400	0.9733	0.9867	0.9733	0.7067
Binder	0.9467	0.8800	0.9067	0.6667	0.9467
Exploit	0.7200	0.9333	0.9467	0.7067	0.8800
Zombie	0.8933	0.9733	0.7600	0.8667	0.9067
Authentication	0.9733	0.6533	0.9067	0.8800	1.0000
Click fraud	0.4533	0.9733	0.8933	0.9733	0.9600
Password	0.9733	0.7333	0.9333	0.8800	0.9733
UserID	0.9867	0.7867	0.9600	0.8000	0.9733
Firewall	0.9733	0.5600	0.9867	0.8667	0.9467
Backdoor	0.7200	0.7333	0.9600	0.9200	0.9067
Blacklist	0.8533	0.8267	1.0000	0.8000	0.9733
Spoofing	0.5600	0.9200	0.9467	0.9867	0.9467
Dropper	0.8400	0.9467	0.7867	0.8667	0.9067
Address book	0.9333	0.8933	0.9333	0.6800	1.0000
Honeypot	0.9200	0.8933	0.9867	0.7600	0.9733

Table 3. Non-Expert distances ($D_{NE}(M,r)$) between security risks and five mental models

Security Risk	Criminal	Physical	Medical	Economic	Warfare
Adware	0.8980	0.7143	0.9524	0.8571	0.9048
Spyware	0.7483	0.7891	0.9524	0.8844	0.9184
Phishing	0.7075	0.8435	0.9592	0.8844	0.8912
Identity theft	0.3537	0.8912	0.9660	0.9456	0.8367
Spam	0.7279	0.8571	0.9184	0.8231	0.8639
Hijackers	0.4286	0.9048	0.9524	0.9660	0.6327
Cookies	0.8844	0.7823	0.9660	0.7279	0.8844
DoS attack	0.8163	0.8163	0.8912	0.8095	0.8571
Drive-by-download	0.8027	0.8707	0.9524	0.8299	0.9320
Trojan	0.6803	0.8503	0.9388	0.9048	0.6939
Keystroke logger	0.7075	0.6667	0.9524	0.8299	0.8639
Junk mail	0.7483	0.8299	0.9388	0.8639	0.8435
Virus	0.7755	0.9116	0.5646	0.9592	0.8231
Worm	0.6735	0.8503	0.8231	0.8844	0.8095
Hacking	0.4830	0.8367	0.8980	0.8844	0.7279
Binder	0.9252	0.8231	0.9864	0.7347	0.9524
Exploit	0.6735	0.9592	0.9388	0.7415	0.8435
Zombie	0.8367	0.8980	0.8571	0.7891	0.7823
Authentication	0.9184	0.5510	0.9796	0.8980	0.9524
Click fraud	0.5578	0.8231	0.9252	0.8571	0.8980
Password	0.9456	0.5646	0.9932	0.8639	0.9388
UserID	0.9524	0.6259	1.0000	0.7891	0.9524
Firewall	0.8844	0.5102	0.9660	0.8707	0.8707
Backdoor	0.7143	0.7347	0.9660	0.8571	0.8571
Blacklist	0.8027	0.6735	0.9592	0.8571	0.9116
Spoofing	0.6463	0.8912	0.9592	0.8231	0.8844
Dropper	0.8231	0.9524	0.8231	0.7823	0.9184
Address book	0.9796	0.8503	0.9660	0.6939	0.9592
Honeypot	0.9388	0.8776	0.9524	0.7211	0.9592

Table 4. *Non-expert and expert mental models.* $P(NE, M, r)$ *is the probability that a non-expert selects M as its mental model for the risk r. $P(E, M, r)$ is defined similarly. These probabilities are calculated based on the original data entries for the non-expert and experts participants. For instance the probability of having "physical security" as the non-experts' mental model for the risk "Adware" is 0.28. The reason for this is that 28% of non-experts have assigned this mental model to the risk "Adware".*

Risk	NM_r	$P(NE, M, r)$	EM_r	$P(E, M, r)$	different MM
Adware	Physical	28.57%	Economic	28%	√
Spyware	Criminal	26.53%	Criminal	36%	
Phishing	Criminal	20.41%	Criminal	56%	
Identity theft	Criminal	59.18%	Criminal	72%	
Spam	Criminal	22.45%	Economic	16%	√
Hijackers	Criminal	48.98%	Criminal	72%	
Cookies	Economic	6.12%	Economic	16%	
DoS attack	Economic	14.29%	Criminal	32%	√
Drive-by-download	Criminal	10.20%	Criminal	24%	
Trojan	Criminal	22.45%	Criminal	28%	
Keystroke logger	Physical	28.57%	Criminal	24%	√
Junk mail	Criminal	16.33%	Economic	12%	√
Virus	Medical	38.78%	Medical	48%	
Worm	Criminal	26.53%	Medical	20%	√
Hacking	Criminal	36.73%	Criminal	40%	
Binder	Economic	8.16%	Economic	12%	
Exploit	Criminal	28.57%	Economic	20%	√
Zombie	Warfare	18.37%	Medical	28%	√
Authentication	Physical	51.02%	Physical	36%	
Click fraud	Criminal	42.86%	Criminal	60%	
Password	Physical	48.98%	Physical	28%	
UserID	Physical	38.78%	Physical	28%	
Firewall	Physical	46.94%	Physical	24%	
Backdoor	Criminal	20.41%	Criminal	28%	
Blacklist	Physical	36.73%	Economic	12%	√
Spoofing	Criminal	32.65%	Criminal	52%	
Dropper	Economic	10.20%	Medical	20%	√
Address book	Economic	4.08%	Economic	12%	
Honeypot	Economic	12.24%	Economic	20%	
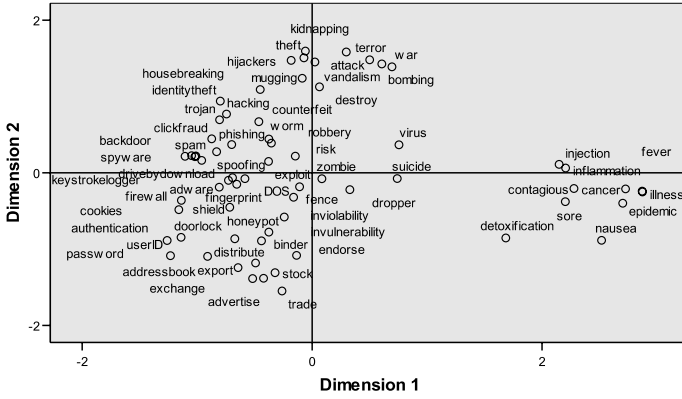
Fig. 2. Multidimensional Scaling Map for Non-Expert Choice Matrix
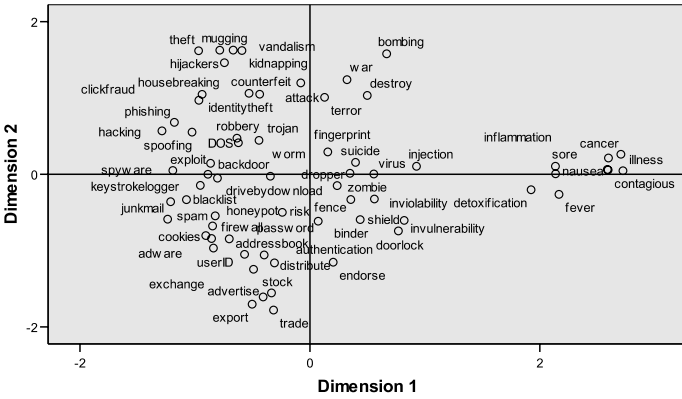
Fig. 3. Multidimensional Scaling Map for Expert Choice Matrix

Improving Usability by Adding Security
to Video Conferencing Systems

April Slayden Mitchell and Alan H. Karp

Hewlett-Packard Laboratories
{april.mitchell, alan.karp}@hp.com

Abstract. Many video conferencing solutions exist in the market today
and many new ones are being introduced. In striving to provide an ex-
perience to users as close to "being there" as possible, two major design
issues must be considered: security and ease of use. In this paper, we
describe a method for designing a "One Space" video conferencing user
interface that reveals security information to the users while reducing
the complexity of the user experience.

1   Motivation

Many desktop video conferencing systems are being sold as a way to reduce the
need for business trips. Shortcomings of these systems are apparent to many
consumers; we'll focus on two of them here. The first drawback is a confusing
interface that makes many conferencing applications too complicated to use.
The second is that private discussions cannot be shared via the public internet
for fear of leaks and vulnerability. While encryption protects the data being
transmitted, users are also concerned about off-camera "watchers" and over the
phone "listeners". In this paper, we describe a user interface that addresses both
of these issues.

In comprehensive video conferencing solutions, such as HP's Halo[1] and Cisco's
TelePresence[2], it is important that users are able to operate comfortably while
in the room and are able to make the same assumptions about security as they
are in a standard conference room.

At the center of the complexity and confusion for users of standard video
conferencing applications and conferencing equipment is the user interface. Too
often the interface consists of menus within menus. While this pattern is familiar
from desktop computing, it is nevertheless confusing for occasional users, leading
them to ask questions such as: "Which menu do I look in to find the phone?"
and "Where's the option for sharing my display?"

Nested menus are also detrimental to security. Without extensive customiza-
tion, users are often presented with many options that the security policy pre-
vents them from using, thus revealing protected information and introducing

---

[1] HP Halo: http://www.hp.com/halo/index.html
[2] Cisco Telepresence: http://www.cisco.com/en/US/netsol/ns669/networking_solutions_
solution_segment_home.html

confusion (see principle of expected ability in [2]). In order for a video conferencing application to be a success, the user interface must be understandable by a novice, warrant repetitive use, and reveal the aspects of security relevant to the participants. User studies showed that knowing who was listening was the top concern. Clearly, the introduction of cameras and microphones into a space invalidates the "assumption" of private communication. We took privacy strongly into consideration in developing our model for a secure video conferencing solution interface. The theme of our solution can be summarized by the term "Revelation", revealing the presence of "watchers" and "listeners" as well as making any action taken visible to all participants.

2   The "One Space" Metaphor

A video conferencing system consists of one or more physical locations (rooms containing the video conferencing equipment) connected over a network. In a "One Space" video conferencing system, both the interface and room design encourage people to act as if the physical room is a single location in which all participants are present. All rooms consist of the same equipment and physical layout, including a center table, chairs, lighting, and color scheme. The rooms are not customized based on company or location, thus adding to the illusion of a shared environment.

In addition to various microphones and speakers, each room contains video display screens for showing other attendees and an additional display for showing the shared interface (see Figure 1 for an example from HP Halo). This display contains means for controlling all of the physical devices available in each room, including the cameras, PCs, etc. This shared display is considered an extension of the table desktop and is visible to all attendees in every room.

Fig. 1. This image shows the physical layout of an HP Halo video conferencing room

In the One Space metaphor, it is very important that all rooms always view the same user interface. Attendees in any room can control the interface by using a device such as a mouse, which allows full control of the interface and its actions are visible to everyone, regardless of room. All participants, including those dialing in over the phone, and all resources, such as overhead cameras, are

represented by icons in a virtual room. By presenting a schematic representation of the rooms and functions involved in the meeting, we enforce the One Space metaphor.

Encryption and the use of private networks can address many security concerns. They can not answer questions such as: "Who is on the phone?", "Is there a help-assistant listening?", "Is the door to the room open?", and "Are there people in the room who are off-camera?" Our One Space user interface addresses these questions. There is no way through the interface to connect to and "listen in" to a conference uninvited or unannounced. All video links must be explicitly accepted or declined. The audio doesn't start until the video is displayed. All attendees who dial in to a conference line are represented by a uniquely identified icon. Similarly, the audio connection with the help-assistant is always represented on the user interface by a representation of an occupied or unoccupied desk.

3   Implementation

Our proposed One Space interface, shown in Figure 2, appears on the shared display in each room.[3] It consists of a schematic representation of the physical space and shows virtual devices, such as help information, which are available. Dial-in participants can access this view via a web browser. The numbered callouts in the figure do not appear on the actual interface; they appear only to facilitate the description.

The spatial view of the One Space interface, in which all equipment, devices, etc. are represented as icons, removes the need for traditional nested menus. All icons are pictorial and represent common objects such as a table, a phone, or a camera. Mouse tips can provide alternative denotations. The shared control of the space is an essential element in making the interface more secure and easier to use because there are no hidden actions by any connected parties, and there is no need to worry about the learning curve due to using a room in a different location. All video conference rooms connected will be represented on the One Space interface. Figure 2 represents an event in which two rooms are connected; however, the table as indicated by callout 1 can be expanded to show an event with more rooms or reduced to show only one side of the table when the room is being used locally. As shown in callout 6, the location and company information for each room is displayed on the interface. The public information appearing on the interface can be configured during booking: such as, the attendee list, the conference telephone number, and each room's LAN access (internal or external). If there are network and/or phone connections in a room, then their existence and activity are viewable via the interface. In Figure 2, a laptop is shown on the table (callout 10), which denotes that it has been connected to the network within Company B's room.

---

[3] The ideas contain herein are not details of any video conferencing product by HP or any other company. They are proposals for designing a more usable, more secure video conferencing interface.
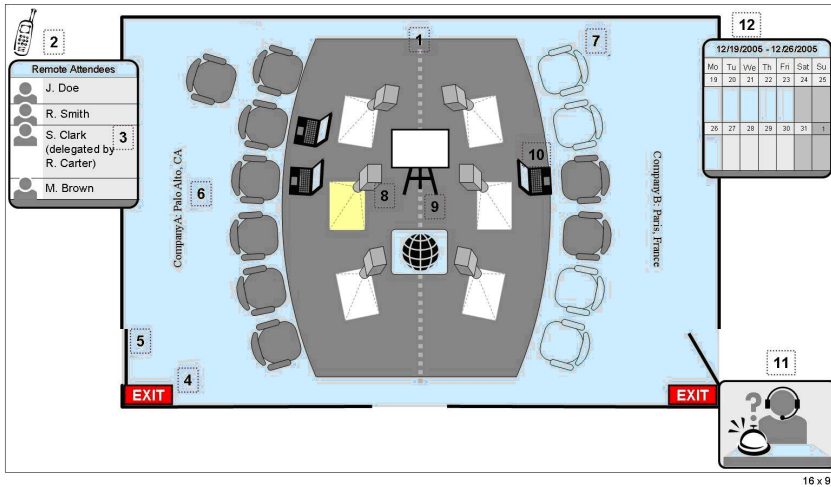
Fig. 2. This is an example of a One Space user interface for a video conference connection between two physical locations with remote participants dialed in via a conference line

In callout 2, remote attendees who are dialed in over the phone are recognized on the interface. This information is tracked through a per-meeting/per-person pass code which each invited member receives. Callers can enter their names via keypad or using speech recognition software. Each name is compared with the name of the responsible party. If the name is different from the name associated with the PIN used, the system recognizes that this person is a delegate of the responsible party and lists the joining party's name as well as the responsible party's name (callout 3).

Every physical seat in each room is shown on the interface as a chair icon colored based on whether or not the seat is currently occupied. Additionally, on-camera seats are distinguished from off-camera seats. Callout 7 shows an off-camera, unoccupied seat. This detection can be done by weight sensors placed on the chair to determine occupancy as well as by camera tracking tools which could identify chairs brought into the room or extra people standing off-camera [1].

In addition to seats in the room, an icon representing the door to the room (callout 5) will also appear on the interface. This icon changes to indicate whether or not the actual physical door is open or closed in order to reveal if persons located outside of the room can easily hear or see participants in the video conference. In addition to icons on the interface which reveal attendance and connection information, there are other icons which allow users in either room to control conference actions. For example, callout 4 indicates an Exit sign which is how a room leaves the conference. All other rooms will still stay connected and the table on the One Space interface will adjust to reflect that one room has left the event.

Several icons representing overhead cameras are shown on the interface. The interface also reveals the area on the table that will be illuminated if it a particular camera is activated. Once a camera or device is activated, the area on the One Space interface table (callout 8) will be shaded a different color to reflect the portion of the physical table that is on camera.

Icons that appear in the center of the table (callout 9) represent virtual devices which can be accessed. These may include a shared web browser or whiteboard. The icon denoted by callout 11 represents the help desk. It should only show the desk as occupied when the help help-assistant is online and can interact with the meeting attendees. When the help-assistant is not connected, the icon should appear as an unoccupied desk with a bell. Any user may click on the bell to contact the help-assistant.

A calendar, as indicated by callout 12, should be accessible when in a video conference. This calendar shows the availability of all rooms currently connected. Through this interface, attendees are able to schedule a follow-up event with the same rooms and the same attendees, but not access any other room's calendars or change the attendee list.

4   Conclusion

Utilizing the One Space metaphor for the user interface both enhances meeting attendees' feelings of occupying a shared physical space and provides confidentiality assurances, such as knowing at all times who else is present in the connected rooms. The schematic of the One Space interface makes using the various devices in connected rooms intuitive while enhancing security by revealing physically absent listeners at all times. Similarly, the pictorial representation of all devices, attendees, etc. and the consistency of the interface regardless of room locale makes repetitive use simple. HP has utilized portions of the One Space interface guidelines in designing the interface for Halo, and internal studies have shown successful use by novices and overall improved user experience. Utilizing the One Space metaphor when designing video conferencing interfaces will provide more visible security assurances and support repetitive use, all while enabling even a novice to use the system to its fullest extent.

References

1. Harville, M., Li, D.: Fast, Integrated Tracking and Activity Recognition with Plan-View Templates from a Single Stereo Camera. In: Proceedings of the IEEE Computer Vision and Pattern Recognition Conference, Washington, DC (2004)
2. Yee, K.: User Interaction Design for Secure Systems. In: Proceedings of the Fourth Intl. Conference on Information and Communications Security, Singapore (2002)

A Sense of Security in Pervasive Computing⋆—Is the Light on When the Refrigerator Door Is Closed?

Jakob Illeborg Pagter[1] and Marianne Graves Petersen[2]

[1] Centre for IT Security, The Alexandra Institutet Ltd., Denmark
[2] Department of Computer Science, University of Aarhus, Denmark

Abstract. In this paper, we investigate how existing theoretical contributions on usable security can serve to guide the design of a specific system. We illustrate how going through this theoretically informed, concrete design process, also provides the basis for complementing existing theoretical contributions. The system we have designed is a system taking advantage of pervasive computing technology to offer hotel guests access to their personal, digital materials while in a hotel room. The design is based on two ideas novel to usable security, namely falsification and the singleton invariant..

1 Introduction

New infrastructures and interaction technologies potentially provide new means for accessing and integrating personal, digital materials in everyday life. While on a business trip, you may access your personal music, film and movie collection stored on a home-server from anywhere in the world. This opens up for new opportunities, at the same time as it raises a need for dealing with security in a way, which makes it feel safe to exploit these new opportunities. We are particularly interested in investigating new contexts and situations where secure access to personal, digital materials contribute to the growing experience economy (Pine and Gilmore [13]). At the same time, we wish to explore such situations in order to contribute to the emerging area of usable security.

Recently, there has been a growing activity in the area of usable security—or HCI-SEC [4], and we take this as our starting point. In line with others, our starting point is that systems need to be designed in a way which integrates qualities and that neither usability nor security can successfully be designed in isolation or as an add-on ([14,11]). Different strategies for such integration have been presented. One of the key ideas is to seamlessly integrate security and user goals and let user goals be what drives the interaction and making the security state somehow visible to the user [14]. We take this approach as our starting point, but we also illustrate how it is sometimes beneficial to make security

---

aspects more visible. This idea is inspired by recent trends in usability research, e.g. designing visible and remarkable computing [12].

We also share the starting point of Grinter and Smetters [9] that "security cannot be considered in the abstract, separate from a particular application and context of use". We have investigated how to design for secure access to personal digital materials in a hotel room. This setting is chosen as it exemplifies how pervasive computing can contribute new experiences in everyday life. Also, we find that the semi public nature of a hotel room is an interesting site for exploring the nature of usable security in everyday life. Inspired by Dourish et al. [7] we have learned from how people organize themselves around physical materials. We have used this as a basis for designing a system which allows secure access to personal materials in a hotel room.

2   Designing for Usable Security

The existing literature on HCI-SEC has concentrated on two main areas: A) studies of the usability of existing security systems (e.g. [1,15]), and B) studies on how to design systems with usable security (e.g. [6,7,14]). We have focused on the latter area as our interest is to design usable security systems, but they are of course intertwined.

Based on [2,5,6,7,9,10,14,16,17] we identify the following overall principles for designing for usable security: 1) Design for a specific context; 2) Establish coherence between "normal" actions and security actions; 3) Make security states visible; 4) Implicit infer security actions from user actions; and 5) Use explicit security actions when users need to act in response to significant security risks.

As investigated in the following, we hypothesise that explicit security actions can be used strategically to design for new engaging experiences, offering security as a major part of the experience. In the following, we investigate how this can be realised in a hotel room context.

Due to the private nature of life in hotel rooms, and therefore the challenges in studying this, we have embarked on a number of resources in understanding how costumers behave. We have used cultural probes materials [8], and in particular the untraditional, but highly relevant empirical material collected by the artist Sophie Calles [3]. Based on her observations and experiences as a maid during a three week stint at a hotel in Vienna, she carefully documented and reflected on how people live in hotel rooms. Further there is a potential in offering new experiences in a hotel room, exploiting pervasive computing technologies. In hotel rooms, people typically bring some personal technologies, but the hotel may offer enhanced experiences if a secure and easy transfer of personal media to other platforms, e.g. wide screen, high quality sound systems etc. is possible

Dourish et al. [7], have illustrated how it is important to learn from how people currently organize themselves in terms of physical arrangement of space and objects in designing future interactive systems for the same context. We interpret the observations of Calles [3] onto a set of characteristics of life in a hotel room in the following ways.

- People creatively inhabit the room with personal materials they have brought from home as well as new materials they acquire on the visit.
- Guests exercise different levels of control of access to the room, using available means, i.e. do not disturb sign on door and lock on suitcase.
- A feeling of control over access to private situations and materials is crucial to shaping a sense of home in a hotel room.

Pervasive computing technology may potentially provide new means for "making home" in a hotel room. In Calles' examples, people brought physical photos and placed them in the room in order to make home. Using available technology, it is possible for guests to access and organize themselves with all their personal digital materials while in a hotel room. But according to the above observations, we propose that this is only an interesting opportunity, if the users, in an understandable way, are able to delicately control the access to these materials.

Above, we identified a number of principles for designing for usable security. Interpreting these in the hotel context, we see how the principle of designing for visible security states is important in this context too. E.g. the do not disturb sign is very visible both to the person placing it on the door and to the maid responding to it. In particular, also placing the sign on the door is a very explicit action, which is not something happening implicitly as part of a naturally occurring action. Thus this is somewhat conflicting with the principle of designing implicit security actions. This context suggests that explicit security actions must be designed for strategically, with respect to the aspects that the user is most concerned with. In this case including: A) controlling who enters and leave the room and when, and B) controlling access to private materials in one's presence and absence from the room. In the following we discuss how to design for B) when exploiting the possibilities for making home with digital materials in a hotel room.

3 Usable Security in a Hotel Room

The basic technical setup is that the user has a cell phone either containing or providing access to a variety of digital objects. Further we have a hotel, where each room has a comptatible net-enabled entertainment system.

Based on the above requirements for establishing a sense of security we propose a singleton invariant solution. The overall idea is that at any point in time, contents or activity the user engages in, is only accessible from a single device, i.e. either from the user's cell phone or from the system in the hotel room. We call this the singleton-invariant and it is directly inspired from the properties of physical objects which can only be at one place at a time.

A customer/user in a hotel will experience the following sequence of actions: 1) The customer checks into the hotel at night and his cell phone is associated to the hotel room. 2) Upon entering the hotel room she can transfer objects from the cell phone to the hotel room system. For instance the user might place photos on the walls and open a conversation with her family at home for the

purpose of, participate in saying goodnight to her kids. 3) The next morning the user leaves the hotel room, goes to a meeting and all personal objects are automatically transferred back to her cell phone. In the cab on the way to the meeting venue, she might continue last night's conversation with the family at home. 4) When later that day she returns to the hotel room, the room is again automatically setup as when she left it in the morning. 5) The next morning she checks out from the hotel, and her cell phone is disassociated from the hotel room system. We see that as specified by the singleton invariant, any digital object is (logically) always either on the phone or on the hotel room system.

From a security point of view, the two primary weak spots are 1) trusting the hotel and its personel, and 2) protecting your cell phone. The first is a necessary assumption. Conversely it could be argued that if we trust the hotel and its personel fully, there is also no need for a security solution. We address the middle road where we may believe the hotel as an organization, but might not trust hotel personnel or others who might enter the room when the guest is not there.

As for the cell phone, the immediate solution is of course the built-in authentication mechanism based on PIN-codes. But as shown by Dourish et al. [7] this mode of access control on cell phones is rarely adopted by users; also the PIN only protects the SIM-card, not all data on the phone. An alternative solution could be to use the physical fact that cell phones are typically in the immediate proximity of their owner. One could for instance build a solution based on Bluetooth enabled watches that when paired to another device will provide a warning when this device is out-of-range. As for usability, we believe that the above solution is indeed designed according to the five guidelines identified above.

4   A Sense of Security?

A central topic in establishing a sense of security is how to convey the basic security properties to the user, in particular the singleton-invariant. According to Dourish et al. [7], this property must be highly visible and available for inspection and examination. The problem is of course, that in the above solution, when a guest leaves the room, his private documents are automatically removed and when he re-enters they are automatically displayed again. How can he know what is displayed in the hotel room when he is outside the room? Is the light on when the refrigerator is closed? Dourish et al. emphasize that security should not be transparent but that it should be "highly visible—available for inspection and examination". The point is that visibility does not imply availability for inspection and examination, but that these exact properties may be of high importance in some cases.

It is a well-known and wide-spread scientific principle that any hypothesis must be falsifiable, i.e. it must be possible to construct an experiment which potentially disproves the hypothesis. Experiments which fail to falsify the hypothesis, increase your trust in the hypothesis. We propose to use falsifiability as an instrument in realising inspection and examination. Falsifiable security makes a non-visible security property (i.e. a part of the security state which is

not directly invisible, but not observed by the user) visible through an explicit action of the user. A good everyday example is locking the door when you leave your home to go for work; it is reassuring to check that the door is indeed locked.

In this light, the hotel solution presented above in section 4 is in fact too automatic in inferring security actions from the user's actions. Even though the inferred actions are correct, they inhibit the user from attempting to falsify the hypothesis that the singleton-invariant actually works. One way to achieve falsifiability is to change the solution so that objects are no longer automatically moved from the cell phone to the hotel room system as the user enters. Instead we reserve some special command or gesture for moving all objects back to the hotel room system. Note that we might instead have disabled the automatic disengage or both. We do not disable both because disabling one is sufficient and we aim for maximal implicitness and strategically chosen explicit security actions. We choose to disable disengage rather than engage because security is more critical when the user leaves than when he re-enters.

When the user leaves the hotel room he can not only see that objects are moved back to his cell phone, he may also physically go back into the room and verify that all objects have been removed. In this way, falsifiability is ensured.

The change in the interaction is small and subtle, but we are convinced that using the concept of falsifiability in this "strategic" manner can significantly increase the user's sense of security.

5   Conclusions and Future Work

The hotel case illustrates in a concrete situation how security is not just a "necessary evil" which needs to be dealt with in one way or another. On the contrary, using falsifiable security, security can in fact become a visible, enabling factor; it can be a significant part of the experience which is sold to people.

Realised in this way, it can give people true control over their most sensitive documents. Our main contributions include falsifiable security as an extension of existing work on how to design systems that are secure and usable. Further we have demonstrated how it is possible to work with these theoretical ideas in the practical design of a system based on the singleton invariant principle. A minor contribution is that our work emphasizes the fact that usable security for commercial applications is a domain far wider than home banking and similar web-based solutions.

To further mature and test our theoretical and practical concept, we plan to implement aspects of the system in the future and to conduct an evaluation of this implementation. Future work also includes addressing the common scenario of multiple guests in the same hotel room.

In the present work we have articulated how falsifiable security can be implemented in a pervasive computing system, which has strong physical aspects, however, falsifiable security can be further matured through investigating how this can be designed for in contexts with less physical aspects, e.g. can it contribute to prevent phishing.

References

1. Adams, A., Sasse, M.: Users are not the enemy. Communications of the ACM 42, 12 (1999)
2. Bardram, J.E., Kjr, R., Pedersen, M.: Context-aware user authentication - supporting proximity-based login in pervasive computing. In: UbiComp 2003 (2003)
3. Calles, S.: Double Game. Violette Editions (2000)
4. Cranor, L.F., Garfinkel, S.: Secure or usable? IEEE Security and Privacy 2,5 (2004)
5. de Paula, R., Ding, X., Dourish, P., Nies, K., Pillet, B., Redmiles, D., Ren, J., Rode, J., Filho, R.: In the eye of the beholder: A visualization-based approach to information systems security. International Journal of Human Computer Studies Special Issue on HCI Research in Privacy and Security (2005)
6. de Paula, R., Ding, X., Dourish, P., Nies, K., Pillet, B., Redmiles, D., Ren, J., Rode, J., Filho, R.: Two experiences designing for effective security. In: Symposium on Usable Privacy and Security (SOUPS) (2005)
7. Dourish, P., Grinter, R.E., de la Flor, J.D., Joseph, M.: Security in the wild: User strategies for managing security as an everyday, practical problem. Personal and Ubiquitous Computing 8 (2004)
8. Gaver, W., Dunne, T., Pacenti, E.: Cultural probes. In: Interactions, vol. 6,1, pp. 21–29. ACM Press, New York (1999)
9. Grinter, R.E., Smetters, D.K.: Three challenges for embedding security into applications. In: Proceedings of CHI 2003 Workshop on HCI and Security Systems (2003)
10. Newman, R., Gavette, S., Yonge, L., Anderson, R.: Protecting dometisc powerline communications. In: Symposium on Usable Privacy and Security (SOUPS) (2006)
11. Palen, L., Dourish, P.: Unpacking "privacy" for a networked world. In: Proceedings of the ACM Conference on Human Factors in Computing Systems, CHI 2003, pp. 129–136. ACM, New York (2003)
12. Petersen, M.G.: Remarkable computing—the challenge of designing for the home. In: Proceedings of CHI 2004, pp. 1445–1449. ACM Press, New York (2004)
13. Pine, B.J., Gilmore, J.H.: The Experience Economy: Work Is Theater & Every Business a Stage. Harvard Business School Press, Boston (1999)
14. Smetters, D.K., Grinter, R.E.: Moving from the design of usable security technologies to the design of useful secure applications. In: New Security Paradigms Workshop 2002 (2002)
15. Whitten, A., Tygar, J.: Why jonny can't encrypt: A usability evaluation of pgp 5.0. In: Proceedings of the Ninth USENIX Security Symposium (1999)
16. Wu, M., Miller, R., Little, G.: Web wallet: Preventing phishing attacks by revealing user intentions. In: Symposium on Usable Privacy and Security (SOUPS) (2006)
17. Yee, K.-P.: Aligning usability and security. IEEE Security and Privacy 5,2 (2004)

Erratum to: Financial Cryptography and Data Security

Sven Dietrich[1] and Rachna Dhamija[2]

[1] Stevens Institute of Technology, Computer Science Department, Castle Point on Hudson, Hoboken, NJ 07030, USA
spock@cs.stevens.edu
[2] Harvard University,Center for Research on Computation and Society, DEAS, Maxwell Dworkin 110, 33 Oxford Street, Cambridge, MA 02138, USA
rachna@deas.harvard.edu

Erratum to:
S. Dietrich and R. Dhamija (Eds.)
Financial Cryptography and Data Security
DOI: 10.1007/978-3-540-77366-5

The book was inadvertently published with an incorrect name of the copyright holder. The name of the copyright holder for this book is: © Springer-Verlag Berlin Heidelberg. The book has been updated with the changes.

Author Index