LNCS 3570

Andrew S. Patrick
Moti Yung (Eds.)

# Financial Cryptography and Data Security

## 9th International Conference, FC 2005
Roseau, The Commonwealth of Dominica
February/March, 2005, Revised Papers

Springer

# Lecture Notes in Computer Science 3570

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

Editorial Board

David Hutchison
*Lancaster University, UK*

Takeo Kanade
*Carnegie Mellon University, Pittsburgh, PA, USA*

Josef Kittler
*University of Surrey, Guildford, UK*

Jon M. Kleinberg
*Cornell University, Ithaca, NY, USA*

Friedemann Mattern
*ETH Zurich, Switzerland*

John C. Mitchell
*Stanford University, CA, USA*

Moni Naor
*Weizmann Institute of Science, Rehovot, Israel*

Oscar Nierstrasz
*University of Bern, Switzerland*

C. Pandu Rangan
*Indian Institute of Technology, Madras, India*

Bernhard Steffen
*University of Dortmund, Germany*

Madhu Sudan
*Massachusetts Institute of Technology, MA, USA*

Demetri Terzopoulos
*New York University, NY, USA*

Doug Tygar
*University of California, Berkeley, CA, USA*

Moshe Y. Vardi
*Rice University, Houston, TX, USA*

Gerhard Weikum
*Max-Planck Institute of Computer Science, Saarbruecken, Germany*

Andrew S. Patrick Moti Yung (Eds.)

# Financial Cryptography and Data Security

9th International Conference, FC 2005
Roseau, The Commonwealth of Dominica
February 28 – March 3, 2005
Revised Papers

Springer

Volume Editors

Andrew S. Patrick
National Research Council of Canada
1200 Montreal Road, Ottawa, ON, Canada K1A 0R6
E-mail: Andrew.Patrick@nrc-cnrc.gc.ca

Moti Yung
RSA Laboratories and Columbia University
Computer Science, 1214 Amsterdam Ave., New York, NY, USA
E-mail: moti@cs.columbia.edu

Library of Congress Control Number: 2005928164

CR Subject Classification (1998): E.3, D.4.6, K.6.5, K.4.4, C.2, J.1, F.2.1-2

| | |
|---|---|
| ISSN | 0302-9743 |
| ISBN-10 | 3-540-26656-9 Springer Berlin Heidelberg New York |
| ISBN-13 | 978-3-540-26656-3 Springer Berlin Heidelberg New York |

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

springeronline.com

© Springer-Verlag Berlin Heidelberg 2005
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11507840 06/3142 5 4 3 2 1 0

# Preface

The 9th International Conference on Financial Cryptography and Data Security (FC 2005) was held in the Commonwealth of Dominica from February 28 to March 3, 2005. This conference, organized by the International Financial Cryptography Association (IFCA), continues to be the premier international forum for research, exploration, and debate regarding security in the context of finance and commerce. The conference title and scope was expanded this year to cover all aspects of securing transactions and systems. The goal is to build an interdisciplinary meeting, bringing together cryptographers, data-security specialists, business and economy researchers, as well as economists, IT professionals, implementers, and policy makers.

We think that this goal was met this year. The conference received 90 submissions and 24 papers were accepted, 22 in the Research track and 2 in the Systems and Applications track. In addition, the conference featured two distinguished invited speakers, Bezalel Gavish and Lynne Coventry, and two interesting panel sessions, one on phishing and the other on economics and information security. Also, for the first time, some of the papers that were judged to be very strong but did not make the final program were selected for special invitation to our Works in Progress (Rump) Session that took place on Wednesday evening. Three papers were highlighted in this forum this year, and short versions of the papers are included here. As always, other conference attendees were also invited to make presentations during the rump session, and the evening lived up to its colorful reputation.

Putting together such a strong program would not be possible without the hard work of the Program Committee, whose members are listed on a separate page. In addition, a large number of external reviewers were recruited because of their special expertise in particular areas, and their names are also listed in these proceedings. Each of the submissions was reviewed by at least three experts, who then engaged in vigorous online discussions. The selection process was difficult because there were many excellent papers that could not be fit into the program. We want to thank all the authors who submitted papers, and we hope that the feedback they received was useful for continuing to develop their work, whether their papers were accepted or not.

We also want to thank this year's General Chair, Stuart Schechter, for valuable assistance and for handling the arrangements in Dominica, and Ari Juels for moderating the rump session. Special thanks also go out to Aggelos Kiayias for setting up and operating the Web-based reviewing system that was essential for handling such a large number of submissions and reviewers.

We hope that this year's program was in the spirit of the conference goals as envisioned, and that the conference continues its colorful tradition as an interdisciplinary, high diversity meeting that helps foster cooperation and the fruitful exchange of ideas among its international participants.

April 2005 Andrew Patrick and Moti Yung

# Financial Cryptography and Data Security 2005

**Program Chairs:** Andrew Patrick and Moti Yung
**General Chair:** Stuart Schechter

## Program Committee

| | |
|---|---|
| Colin Boyd | Queensland University of Technology |
| Suresh Chari | IBM |
| Liqun Chen | HP Labs |
| Lynne Coventry | NCR |
| Yvo Desmedt | University College London |
| Giovanni Di Crescenzo | Telcordia Technologies |
| Roger Dingledine | Moria Research Labs |
| Scott Flinn | National Research Council of Canada |
| Juan Garay | Bell Labs, Lucent Technologies |
| Dan Geer | Geer Risk Services |
| Craig Gentry | DoCoMo Labs USA |
| Mike Just | Treasury Board of Canada |
| Aggelos Kiayias | University of Connecticut |
| Helger Lipmaa | Helsinki University of Technology |
| David M'Raihi | Verisign |
| Kobbi Nissim | Microsoft |
| Satoshi Obana | Columbia University and NEC |
| Andrew Odlyzko | University Minnesota |
| Pascal Paillier | Gemplus |
| David Pointcheval | Ecole Normale Supérieure |
| Bart Preneel | Katholieke Universiteit Leuven |
| Angela Sasse | University College London |
| Berry Schoenmakers | Technische Universiteit Eindhoven |
| Sean Smith | Dartmouth College |
| Jessica Staddon | Palo Alto Research Center (PARC) |
| Michael Szydlo | RSA Laboratories |
| Jacques Traore | France Télécom |
| Gene Tsudik | University of California, Irvine |
| Alma Whitten | Google |
| Adam Young | Cigital |
| Bill Yurcik | NCSA |

## Sponsors

| | |
|---|---|
| Gold Sponsor: | Interactive Investor (www.iii.co.uk) |
| Bronze Sponsors: | RSA Security<br>France Télécom |
| In-Kind Sponsor: | Bibit Global Payment Services |

## External Reviewers

Michel Abdalla
Gildas Avoine
Alexandra Boldyreva
Calude Castelluccia
George Danezis
Alex Deacon
Glenn Durfee
Renwei Ge
Rosario Gennaro
Henri Gilbert
Marc Girault
David Goldberg
Philippe Golle
Juan Gonzalez
Rachel Greenstadt
Shai Halevi
Helena Handschuh
Yvonne Hitchcock
Kevin Soo Hoo
Markus Jakobsson
Stas Jarecki
Charanjit Jutla
Sébastien Kunz-Jacques
Joseph Lano
John Malone-Lee
Nick Mathewson
Sean Murphy
Steve Myers
Gregory Neven
Jean-Claude Pailles
Valeria de Paiva
Duong Hieu Phan
Tal Rabin
Yona Raekow
Zulfikar Ramzan
Josyula R. Rao
Jason Reid
Pankaj Rohatgi
Markku-Juhani O. Saarinen
Marius Schilder
Umesh Shankar
Vitaly Shmatikov
Paul Syverson
Jun'ichi Takeuchi
Yiannis Tsiounis
Yunlei Zhao
Hong-Sheng Zhou

# Table of Contents

## Threat and Attacks

## Invited Speaker

## Digital Signing Methods

## Panel

## Privacy

## Hardware Oriented Mechanisms

## Supporting Financial Transactions

## Systems, Applications, and Experiences

## Message Authentication

## Exchanges and Contracts

## Auctions and Voting

## Works in Progress

## User Authentication

# Fraud Within Asymmetric Multi-hop Cellular Networks

Gildas Avoine

EPFL, Lausanne, Switzerland

**Abstract.** At *Financial Cryptography 2003*, Jakobsson, Hubaux, and Buttyán suggested a lightweight micro-payment scheme aimed at encouraging routing collaboration in asymmetric multi-hop cellular networks. We will show in this paper that this scheme suffers from some weaknesses. Firstly, we will describe an attack which enables two adversaries in the same cell to communicate freely without being challenged by the operator center. We will put forward a solution to fix this protocol. Then we will describe another method that allows an attacker to determine the secret keys of the other users. This attack thwarts the micro-payment scheme's purpose because an attacker can thus communicate without being charged. Finally we will suggest some solutions to counteract this attack.

**Keywords:** Micro-payment, multi-hop cellular networks, cryptanalysis.

## 1 Introduction

Nowadays, architectures for wireless communication are mostly based on *single-hop cellular networks*, e.g., the Global System for Mobile communications (GSM) [1]. Within this framework, mobile stations can access the infrastructure with a single hop and base stations can also reach each mobile station in its cell with one hop. However, such infrastructures require multiple fixed base stations to encompass the service area, which can lead to numerous problems. Conversely, *multi-hop networks*, also called *ad-hoc networks*, do not rely on a fixed infrastructure; mobile stations communicate amongst themselves using multi-hop routing. Though these networks have some advantages, mainly their low cost, they bring with them several problems, chiefly related to the routing process (congestion, selfishness, etc.). *Multi-hop cellular networks* [3] mitigate these problems by combining conventional single-hop cellular networks and multi-hop networks. Here, the multi-hop routing is only used in order to reach the closest base station and to link the destination base station with the recipient user. A variant of this kind of network, introduced by Jakobsson, Hubaux, and Buttyán [2] consists of a multi-hop uplink, i.e., the link from the mobile station to the base station, and a single-hop downlink, i.e., the link from the base station to the mobile station. Such a network, called an *asymmetric multi-hop cellular network*, aims to reduce the energy consumption of the mobile stations.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 1–15, 2005.
© Springer-Verlag Berlin Heidelberg 2005

When a routing protocol is based on multi-hop links, incentives must be used to encourage cooperation between the parties — called *nodes* in this case. Micro-payments are one way to treat the problem. In this paper, we analyze the lightweight micro-payment scheme suggested by Jakobsson, Hubaux, and Buttyán [2] at *Financial Cryptography 2003*, which aims to encourage cooperation in asymmetric multi-hop cellular networks. In this scheme, the cost paid by the packets' originators covers on average the routing cost, which includes the (probabilistic) gain of the intermediaries along the packet route. We will show in this paper that the proposed scheme suffers from some weaknesses which compromise its security. In Section 2, we will recap the main principles of the analyzed micro-payment scheme. In Section 3, we will firstly describe a method which allows two attackers in the same cell to communicate freely; we will then suggest a lightweight patch in order to fix the scheme. In Section 4, we will describe another threat, which enables an attacker to determine the secret keys of the nodes. This attack thwarts the micro-payment scheme's purpose because, with these keys, an attacker can communicate without being charged; the owners of the stolen keys are charged instead. Finally, we will also suggest mechanisms to counteract this using keyed-hash functions.

## 2 Description of the Scheme

### 2.1 Entities

The micro-payment scheme suggested by Jakobsson *et al.* [2] consists of three classes of entities: the *users*, the *base stations* and the *operator centers*. Among the users, we distinguish between the *originators* of the packets, the *intermediaries* on the path from the originator to the base station and the *recipients* of the packets. We also recognize the *base stations of the home network* of a user, i.e., the network where the user is registered and the *base stations of the foreign networks*. There is an *operator center* per network, which is simultaneously an accounting, auditing and registration center.

### 2.2 Principle

Before sending a packet, an originator has to send a forward request including a reward level $L$ to his neighbors, one after another, until one of them agrees to forward the packet. The reward expected by the participating neighbor is related to $L$. Increasing the reward level allows users with particularly low battery power to obtain service in a neighborhood populated with low battery resources. The authors of [2] suggested a system in which all packet originators attach a payment token to each packet they send. Each intermediary on the packet's path to a base station then verifies whether this token is a *winning ticket* for him. This outline is based on the probabilistic payments suggested by Rivest [4]. Intermediaries with winning tickets can send a *reward claim* to their accounting center in order to be rewarded for their work. The cost paid by the originator covers — on average — the cost of routing and other network maintenance. Therefore, base stations receive two kinds of packet: reward claims that they send to the accounting centers and packets with payment tokens. In the latter, the base stations send the packets (without the token) to the expected destination and the tokens are sent to the accounting centers. Packets with invalid tokens are dropped, as the transmission cannot be charged to anybody. The packet transmission procedure is detailed in Section 2.4 and the reward protocol is described in Section 2.5.

### 2.3 Setup

When a user registers for access to the home network, he is assigned an identity $u$ and a symmetric key $K_u$. The pair $(u, K_u)$ is stored by both the user and the user's home network. From a routing point of view, each user $u$ manages a list $\lambda_u = ((u_i, d_i, L_i))_i$ where $u_i$ is the identity of a neighbor, $d_i$ its path length (in terms of hops) to the closest base station and $L_i$ its threshold for forwarding packets as explained later. $\lambda_u$ is increasingly sorted according to $d_i$ and then $L_i$.

### 2.4 Packet Transmission Protocol

**Origination.** The originator $u_o$ of the packet $p$ performs the following procedure.

1. Selects the reward level $L \in [0, \max_L]$.
2. Computes $\mu = \mathrm{MAC}_{K_{u_o}}(p, L)$ where MAC is a keyed-hash function.
3. Sends the tuple $P = (L, p, u_o, \mu)$ according to the Transmission procedure.

**Transmission.** In order to send a tuple $P = (L, p, u_o, \mu)$, a user $u$ (originator or intermediary) performs the following procedure.

1. If the base station can be reached in a single hop then $u$ sends $P$ directly to it. If not, he goes to Step 2.
2. $u$ selects the first entry $(u_i, d_i, L_i)$ from $\lambda_u$ for which $L_i \leq L$. If such an entry does not exist then $u$ drops the packet.
3. $u$ sends a *forward request* to $u_i$ containing the reward level $L$.
4. If $u$ receives an acknowledgment from $u_i$ before a timeout $\delta$, then he sends $P$ to $u_i$. If not, he goes back to Step 2 to the next entry in $\lambda_u$.
5. If $u$ is not the originator of the packet, he carries out the Reward protocol.

**Acceptance by an Intermediary.** When a user $u'$ receives a forward request from a user $u$ with a reward level $L$, he agrees to forward the packet if and only if $L_{u'} \leq L$. If this is the case, he sends an acknowledgment to $u$ and waits for the packet. He then carries out the Transmission procedure.

**Acceptance by a Base Station**

1. When a tuple $P = (L, p, u_o, \mu)$ is received by a base station in the originator's home network, the base station checks whether $\mu = \mathrm{MAC}_{K_{u_o}}(p, L)$ with the stored secret key $K_{u_o}$. If the check fails the packet is dropped; if not, $\mu$ is sent to the accounting center and $p$ is sent to the closest base station to the recipient user. This base station broadcasts the packet to the recipient user.

2. When a tuple $P = (L, p, u_o, \mu)$ is received by a foreign base station, the latter forwards it to the registration center of the originator's home network. This center performs the tasks described in the first step of this procedure.

### 2.5 Reward Protocol

**Recording.** After a user $u$ has forwarded a tuple $P = (L, p, u_o, \mu)$, he verifies whether $f(\mu, K_u) = 1$ where $f$ is a given function described in Section 2.6. If the check succeeds, we can say that the user has a *winning ticket* and can claim a reward for this ticket. In this case, he records $(u_1, u_2, \mu, L)$ where $u_1$ is the identity of the user from whom he received the packet and $u_2$ is the identity of the user (or base station) to whom he sent the packet. Let $M$ be the list of recorded reward 4-tuples.

**Sending.** When the user is able to reach the base station with only one hop, he sends the claim $(u, M, m)$ directly to it, where $m = \text{MAC}_{K_u}(\text{hash}(M))$. If not, the claim is sent to the base station using the same procedure as a usual packet. Note that the list $M$ is encrypted with the key of the user in both cases.

An example of packet transmission and reward claims is given on Fig. 1.

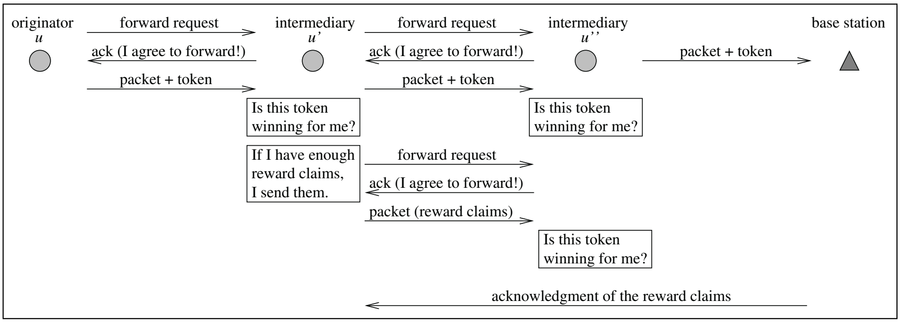

originator
u
forward request
ack (I agree to forward!)
packet + token
intermediary
u'
forward request
ack (I agree to forward!)
packet + token
intermediary
u''
packet + token
base station
Is this token winning for me?
Is this token winning for me?
If I have enough reward claims, I send them.
forward request
ack (I agree to forward!)
packet (reward claims)
Is this token winning for me?
acknowledgment of the reward claims

**Fig. 1.** Example of packet forwarding

$u$ sends a packet. $u'$ and then $u''$ agree to forward it. The token is winning for $u'$ and he has enough reward claims to send them to the base station. $u''$ agrees to forward the reward claims. The base station acknowledges the reception of the reward claims.

### 2.6 Winning Function

The winning function $f$ determines whether a ticket $\mu$ is winning for a user $u$. Let $K_u$ be the secret key of $u$. $\mu$ is a winning ticket for $u$ if and only if $f(\mu, K_u) = 1$. Since the attack described in Section 4 exploits this function, its design should be defined with care. Jakobsson *et al.* suggest that this function could be a one-way hash function, but they say that such a function is too costly. Instead, they suggest choosing $f$ such that $f(\mu, K_u) = 1$ if and only if the Hamming distance between $\mu$ and $K_u$ is less than or equal to a threshold $h$, because this function is very lightweight. The authors of [2] note that if the list of recorded reward 4-tuples $M$ is not encrypted, then an attack could be possible. We describe such an attack in Section 4.1 that results in the discovery of all secret keys if $M$ is not encrypted, with only $\eta$ requests to an oracle, where $152 \leq \eta \leq 339$ in practice. We then show in Section 4.3 that such an attack remains possible even when $M$ is encrypted. In this case, the complexity of the attack depends on the implementation and the victim's environment, but it remains proportional to $\eta$.

### 2.7 Accounting and Auditing

The scheme described in [2] relies on an accounting and auditing center. We assume for the sake of simplicity that these two entities are one and the same, along with the registration center. We call it the *operator center*. Note that there is only one operator center per network. The accounting center receives both user claims and transmission transcripts, both forwarded by the base stations. The accounting center periodically verifies all received user claims concerning all the recorded reward tuples it has received from base stations. All recorded originators are charged a usage fee according to their service contract. Moreover, the accounting center credits all parties (except the originator and the base station) whose identity appears in the accepted reward claim. Here, a reward claim is said to have been accepted if it is *correct*, i.e., if $f(\mu, K_u) = 1$ and a base station has reported the packet associated with the ticket $\mu$ as having been transmitted. The goal of the auditing center is to detect attacks in the network using statistical methods. According to Jakobsson *et al.*, the following attacks can be detected using the auditing techniques (except the *tampering with claims* attack which is prevented by the used of authentication methods); *selective acceptance*: the user agrees to receive (with the intent to forward it) a packet if and only if it contains a *winning* ticket; *packet dropping*: the user agrees to receive packets but does not forward them — whether he claims credit for winning tickets or not; *ticket sniffing*: a user claims credit for a packet he intercepted, but neither agrees to forward nor actually forward it. A more serious attack consists of users along a fake path submitting claims as if they had routed the packet; *crediting a friend*: a user with a winning ticket claims to have received the packet from (or have send it to) a user other than the true one; *greedy ticket collection*: a user claims credits in excess of those specified by the protocol, by collecting and sharing tickets with colluders; *tampering with claims*: a user modifies or drops the reward claim filed by somebody else in order to increase his profits or to remove harmful auditing information; *reward level tampering*: a packet carries an exaggerated reward level along its path, but the reward level is reduced before it is transmitted to the base station; *circular routing*: the packet transits through a circular routing in order to increase the benefit to the intermediaries; *unnecessary long path routing*: the packet transits through an unnecessary long path within a particular neighborhood in order to increase the benefit to the intermediaries since they have a valid ticket.

Our goal in this paper is not to discuss this technique. We assume that in the outcome, this statistical method fulfills the claims of the authors.

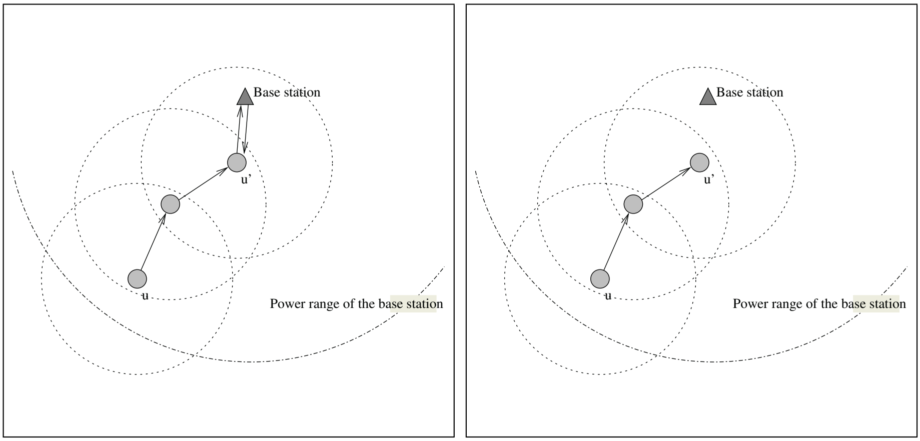
## 3 Communicating Freely in a Cell

In this section, we describe an attack which allows two misbehaving users in the same cell to communicate freely. We will show that this attack can be put into practice rather easily. We will then suggest a lightweight solution to counteract the threat.

### 3.1 Description of the Attack

This attack consists of two users in the same cell communicating freely using fake identities, thus their neighbors will not be rewarded for their work. Firstly we recap that if a user $u$ sends a message to a user $u'$ who is not in his neighborhood, then the packet is sent to the base station through other users. Note that if $u'$ is on the path from $u$ to the base station, then he should not keep the packet when he receives it, but should forward it to the base station and wait for the packet to come back from the base station (see Fig. 2a). Unfortunately, there is no mechanism to protect against adversary wanting to take the packet on the uplink, as represented on Fig. 2b. Such cheaters would not be punished since they are not registered to the accounting center; the weak point being that there is no authentication between the users on the packet path.

Note that it is rather easy for $u'$ to be on the packet path, by claiming a fake distance from the base station and a fake reward level. In particular, if two hops


Base station
u'
u
Power range of the base station
Base station
u'
u
Power range of the base station

(a) Well-behavior: $u'$ forwards the packet to the base station and waits for it to come back.

(b) Misbehavior: $u'$ keeps the packet when he receives it instead of forwarding it to the base station.

**Fig. 2.** $u'$ is the final recipient of the packet are enough to link $u$ and $u'$, the attack will definitely succeed: $u'$ announces in his neighborhood that he is able to reach the base station with only one hop even if it is untrue. Due to this unfair competition, his neighbors will choose him to route the packets[1]. Better still, the recipient attacker can be "near" the path (i.e., she can eavesdrop the transmitted data without being on the routing path) or "near" the base station and she can then hijack the packet without even being on the packet routing path. Even if this latter case is not scalable, it is realistic because the attack is easy to put into practice. For instance, if one of the attackers lives close to a base station, she can communicate freely in her cell, hijacking the packets which are intended to her. Punishing her is not straightforward because her identity does not appear in the packet and she participates only passively in the attack. Note, however, that the attack is possible on the uplink, but not on the downlink.

### 3.2 Fixing the Scheme

Fixing the scheme without requiring heavy cryptographic functions — which the authors sought to avoid — is a difficult task because the attack relies on the fact that there is no authentication between the users. One way to fix the scheme is to oblige the packet to pass through the base station in order to be usable by the recipient. This can be done if each node on the uplink encrypts the packets that it forwards — with a key also known by the base station — using symmetric encryption which is much less expensive than asymmetric encryption. Thus, each node can be sure that the packet will have to be decrypted by the base station otherwise it will be rendered unusable for the recipient.

However, such a solution is quite costly. We suggest instead relaxing the security requirements. Indeed, since [2] is based on the fact that while a small amount of fraud is acceptable, large-scale fraud has to be avoided, we suggest reducing the number of computations by introducing a probabilistic mechanism: each user encrypts the packet with a probability $\rho$. If $n$ is the number of intermediaries between the two attackers, $\rho$ is the probability that an intermediary encrypts the packet, and $\tau$ is the probability that the attack succeeds, then we have: $\tau = (1-\rho)^n$. Taking, for example $n = 5$, which seems a realistic value and $\rho = \frac{1}{2}$, we have $\tau \approx \frac{3}{100}$. We may even determine a threshold at which the attack is no longer an attractive proposition[2] and consequently decrease $\rho$ until it reaches this threshold. This technique substantially reduces the computations performed by the nodes. If a node decides to reduce $\rho$ in order to save its battery power, it will be detected by the auditing center, since its rate of forwarded encrypted packets over the total number of forwarded packets will be abnormally low.

[1] This misbehavior could also be used to set up a "famine" attack against a node.

[2] The cheater can repeat his attack until it succeeds but if $\rho$ is small, the attack will no longer be attractive due to excessive battery consumption and the delay caused by repeated attempts.

## 4 Recovering Secret Keys Using Side Channel Attack

As discussed in Section 2, the goal of the secret keys stored by the nodes is twofold. Firstly, these keys aim to encrypt the reward claims. Secondly, they are used to charge the originator of packets: the originator's secret key is used to compute a MAC on the packet, which is used by the accounting center in order to charge the owner of the key. In other words, if an attacker is able to steal a secret key, he is able to communicate freely and the charged node is the owner of the stolen key. We will show in this section how an attacker can carry out such an attack. For the sake of simplicity, we will first give a theoretical overview of the attack, showing that if an attacker can access an oracle, defined below, then he can recover the 128 bit keys using only approximately a few hundred oracle requests. We will then show in Section 4.3 that such an oracle is available in practice. Finally we fix the scheme using a keyed-hash function.

### 4.1 Description of the Attack: Theoretical Approach

Firstly, we will recap the principle of the winning tickets. A user sends a tuple $P = (L, p, u_o, \mu)$ to a user $u$ where $\mu = \text{MAC}_{K_{u_o}}(p, L)$; $L$, $p$, $u_o$, and $K_{u_o}$ have already been defined in Section 2.4. $u$ checks whether $f(\mu, K_u) = 1$ that is $d_{\mathcal{H}}(\mu, K_u) \leq h$ where $d_{\mathcal{H}}$ represents the Hamming distance, $h$ is a given threshold, and $K_u$ is the secret key of $u$.

We assume in this theoretical approach that if the test succeeds then $u$ sends the claim $(u_1, u_2, \mu, L)$ to the accounting center[3]; if the test fails, $u$ sends nothing (see Fig. 3). Obviously, the intermediary nodes do not know $K_{u_o}$, therefore they are not able to check whether $\text{MAC}_{K_{u_o}}(p, L)$ is valid. Thus, a node can be seen as an Oracle $\mathcal{O}$, such that for a request $\mu \in \{0,1\}^\ell$ where $\ell$ is the size of the secret key, $\mathcal{O}$ returns *true* if $d_{\mathcal{H}}(\mu, K_u) \leq h$ otherwise we consider that it returns *false*.

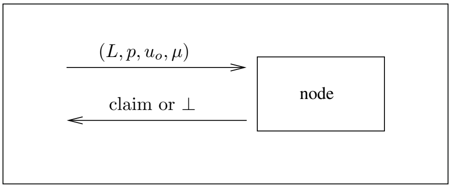


**Fig. 3.** The node can be seen as an oracle

We will now show that some information on the secret key leaks from the oracle. In other words, by sending some forward requests to a node and by spying on

[3] In practice, a claim is not sent as soon as a winning ticket is received, but they are recorded and then encrypted to be sent to the accounting center. We will consider the practical aspects in Section 4.3.

its reward claims, an attacker can determine its secret key. The attack consists of two steps:

1. the first step aims to find a value $\hat{\mu} \in \{0,1\}^\ell$ such that $d_{\mathcal{H}}(\hat{\mu}, K_u) = h$ or $h+1$;
2. the second step aims to recover $K_u$ by sending requests to $\mathcal{O}$ with slight variations of $\hat{\mu}$.

Let us denote $\mathcal{O}(\mu)$ the Boolean answer from the oracle for the request $\mu$; so $\mathcal{O}(\mu)$ means that the answer is *true* and $\neg\mathcal{O}(\mu)$ means that the answer is *false.* Let $\mathcal{F}_{\text{in}} := \{\mu \in \{0,1\}^\ell \mid d_{\mathcal{H}}(\mu, K) = h\}$, $\mathcal{C}_{\text{in}} := \{\mu \in \{0,1\}^\ell \mid d_{\mathcal{H}}(\mu, K) \leq h\}$, $\mathcal{F}_{\text{out}} := \{\mu \in \{0,1\}^\ell \mid d_{\mathcal{H}}(\mu, K) = h+1\}$, and $\mathcal{C}_{\text{out}} := \{\mu \in \{0,1\}^\ell \mid d_{\mathcal{H}}(\mu, K) \geq h+1\}$. Let $\mu_i$ be the $i$-th bit of $\mu$ and $\mu^{(i)}$ be equal to $\mu$ except $\mu_i$ which is flipped. We assume in the sequel that $0 < h < \ell$.

**Step 1.** In order to solve the first step of the attack, we supply a Las Vegas algorithm (see Alg. 1), with parameters $s$, $t$, and $\mu$, which allows to find a value $\mu$ on the border $\mathcal{F}_{\text{in}}$ or $\mathcal{F}_{\text{out}}$. Its principle is the following: given a random value $\mu$, it puts a request to the oracle with this value and then it chooses (possibly randomly) $s$ bits of $\mu$ if $\mathcal{O}(\mu)$ (resp. $t$ bits of $\mu$ if $\neg\mathcal{O}(\mu)$), $r_1, r_2, \ldots, r_{s \text{ or } t}$, and sends $\mu^{(r_1)}, \mu^{(r_2)}, \ldots, \mu^{(r_{s \text{ or } t})}$ to the oracle. We assume that the parameters $s$ and $t$ are such that $(s,t) \neq (0,0)$. Let $\xi(\ell,h,s,t)$ be the probability that Alg. 1 answers. We have:

$$\xi(\ell,h,s,t) := A(\ell,h,s)\Pr(\mu \in \mathcal{C}_{\text{in}}) + B(\ell,h,t)\Pr(\mu \in \mathcal{C}_{\text{out}}) \tag{1}$$

where

$$\begin{aligned} A(\ell,h,s) &= \Pr(\text{Alg. 1 answers} \mid \mu \in \mathcal{C}_{\text{in}}) \\ &= \Pr(\text{Alg. 1 answers} \mid \mu \in \mathcal{F}_{\text{in}}) \\ &= 1 - \binom{h}{s} / \binom{\ell}{s} \text{ if } s \leq h \text{ and 1 otherwise,} \\ B(\ell,h,t) &= \Pr(\text{Alg. 1 answers} \mid \mu \in \mathcal{C}_{\text{out}}) \\ &= \Pr(\text{Alg. 1 answers} \mid \mu \in \mathcal{F}_{\text{out}}) \\ &= 1 - \binom{\ell-h}{t} / \binom{\ell}{t} \text{ if } t \leq \ell - h \text{ and 1 otherwise.} \end{aligned}$$

**Lemma 1.** *Given a random $\mu \in \{0,1\}^\ell$, the probability that $\mu \in \mathcal{F}_{in}$ is $\frac{1}{2^\ell}\binom{\ell}{h}$ and the probability that $\mu \in \mathcal{F}_{out}$ is $\frac{1}{2^\ell}\binom{\ell}{h+1}$.*

*Proof.* The proof is straightforward since $\mid \mathcal{F}_{\text{in}} \mid = \binom{\ell}{h}$, $\mid \mathcal{F}_{\text{out}} \mid = \binom{\ell}{h+1}$, and $\mid \{0,1\}^\ell \mid = 2^\ell$.

□

From Lemma 1, we deduce that the probability that Alg. 1 answers is

$$\xi(\ell,h,s,t) = \frac{\binom{\ell}{h}}{2^\ell} A(\ell,h,s) + \frac{\binom{\ell}{h+1}}{2^\ell} B(\ell,h,t). \tag{2}$$

From the Las Vegas algorithm, it is straightforward to design a Monte Carlo algorithm as represented on Alg. 2. Let $C(\ell,h,s,t)$ be the number of rounds of Alg. 2 in order to find a value on the border; we have:

$$\Pr(C(\ell,h,s,t)=c)=\xi(1-\xi)^{c-1} \text{ if } c>0 \text{ and } \Pr(C(\ell,h,s,t)\leq 0)=0.$$

```
Alg. 1: Find-Border-Las-Vegas($s$,$t$,$\mu$)

send $\mu$ to the oracle $\mathcal{O}$
if $\mathcal{O}(\mu)$ then $b \leftarrow s$
else $b \leftarrow t$
end
pick $b$ distinct random $r_i$ in $[1,\ell]$
send $\mu, \mu^{(r_1)}, \mu^{(r_2)}, \dots, \mu^{(r_b)}$ to $\mathcal{O}$
if $\mathcal{O}(\mu) \wedge \neg \left(\bigwedge_{i=1}^{i=b} \mathcal{O}(\mu^{(r_i)})\right)$
    then return "$\mu$ is in $\mathcal{F}_{\text{in}}$"
else
   if $\neg\mathcal{O}(\mu) \wedge \left(\bigwedge_{i=1}^{i=b} \mathcal{O}(\mu^{(r_i)})\right)$
      then return "$\mu$ is in $\mathcal{F}_{\text{out}}$"
   else return $\perp$
   end
end
```

```
Alg. 2: Find-Border-Monte-Carlo($s$,$t$)

pick a random value $\mu \in \{0,1\}^\ell$
if Find-Border-Las-Vegas($s,t,\mu$) $\neq \perp$
   then return $\mu$
else
   iterate Find-Border-Monte-Carlo(s,t)
end
```

We compute the average number of rounds of Alg. 2, $\tilde{C}(\ell,h,s,t)$, in order to complete the first step of the attack.

$$\tilde{C}(\ell,h,s,t)=\lim_{k\to\infty}\sum_{c=1}^{k} c\,\xi(1-\xi)^{c-1}=\frac{\xi}{(1-(1-\xi))^2}=\frac{1}{\xi}. \tag{3}$$

Given that each round of Alg. 2 requires either $t+1$ (with probability $\sigma$) or $s+1$ (with probability $1-\sigma$) calls to the oracle, and that $(s,t)\neq(0,0)$, we compute from (2) and (3) the average number of requests to the oracle in order to complete the first step of the attack:

$$\frac{2^\ell(1+s\sigma+t(1-\sigma))}{\binom{\ell}{h}A(\ell,h,s)+\binom{\ell}{h+1}B(\ell,h,t)}. \tag{4}$$

**Step 2.** We now consider the second step of the attack, whose complexity is *a priori* $\ell$ according to Lemma 2.

**Lemma 2.** *Given $\mu \in \mathcal{F}_{in}$ (or given $\mu \in \mathcal{F}_{out}$), we can recover the key $K$ with only $\ell$ requests to the oracle $\mathcal{O}$.*

*Proof.* We assume that we have $\mu \in \mathcal{F}_{\text{in}}$; we know therefore that $\mathcal{O}(\mu)$ is true. For every $i$ $(1 \leq i \leq \ell)$ we ask the oracle the question $\mathcal{O}(\mu^{(i)})$. We have $\mu_i = K_i$ if and only if $\mathcal{O}(\mu^{(i)})$ is false, that is flipping $\mu_i$ moves away $\mu$ from the key $K$. The same track is used to prove the lemma with $\mu \in \mathcal{F}_{\text{out}}$. □

Practically, the number of requests to the oracle in Step 2 can be reduced by (a) exploiting the values already checked in the first step of the attack, (b) using the fact that the second step can be halted as soon as the $h$ or $h{+}1$ bits that differ from the key have been found (the other bits can thus be found by inference). We describe these two points in further detail below.

(a) We can re-use the requests of the last round of Alg. 1. in the second step of the attack. We thus have $s$ answers (resp. $t$ answers) from the oracle if $\mu \in \mathcal{C}_{\text{in}}$ (resp. $\mu \in \mathcal{C}_{\text{out}}$). $s\sigma + t(1-\sigma)$ answers from the oracle are thus already known on average.
(b) Since the second step flips the bits of $\mu$ independently, one after the other, the process can be halted as it has found the $h$ (resp. $h+1$) bits $\mu_i$ s.t. $\mu_i \neq K_i$ or the $\ell - h$ bits s.t. $\mu_i = K_i$ (resp. $\ell - h - 1$) when $\mathcal{O}(\mu)$ (resp. $\neg\mathcal{O}(\mu)$). We denote $\zeta(\ell, h)$ the average number of calls to the oracle that can be saved using this inference method. We compute $\zeta(\ell, h)$ below. We notice that the process stops at the round $i$ if and only if

$$\mathcal{O}(\mu^{(i)}) \neq \mathcal{O}(\mu^{(i+1)}) = \cdots = \mathcal{O}(\mu^{(\ell)}).$$

We have to consider the case where $\mu \in \mathcal{F}_{\text{in}}$ and the case where $\mu \in \mathcal{F}_{\text{out}}$. Let us begin with $\mu \in \mathcal{F}_{\text{in}}$. Let $Y_{\text{in}} := \ell - i$. We have

$$\begin{aligned}
\Pr(Y_{\text{in}} = 1) &= \frac{\ell-h}{\ell-1} \cdot \frac{h}{\ell} + \frac{h}{\ell-1} \cdot \frac{\ell-h}{\ell} \\
\vdots \qquad & \quad \vdots \qquad\qquad \vdots \\
\Pr(Y_{\text{in}} = i) &= \frac{\ell-h}{\ell-i} \cdot \prod_{\substack{j=0\\ j<\ell}}^{i-1} \frac{h-j}{\ell-j} + \frac{h}{\ell-i} \cdot \prod_{\substack{j=0\\ j<\ell}}^{i-1} \frac{\ell-h-j}{\ell-j}
\end{aligned}$$

So, the average number of requests that can be save if $\mu \in \mathcal{F}_{\text{in}}$ is:

$$\tilde{Y}_{\text{in}}(\ell, h) = \sum_{\substack{i=1\\ i<\ell}}^{\infty} i \cdot \left( \frac{\ell-h}{\ell-i} \cdot \prod_{j=0}^{i-1} \frac{h-j}{\ell-j} + \frac{h}{\ell-i} \cdot \prod_{j=0}^{i-1} \frac{\ell-h-j}{\ell-j} \right).$$

If $h \approx \frac{\ell}{2}$, we can estimate $Y_{\text{in}}(\ell, h)$ using a geometric probability law with parameter $\frac{1}{2}$, from which we obtain

$$\tilde{Y}_{\text{in}}(\ell, h) \approx 2. \tag{5}$$

We define $Y_{\text{out}}$ using the same method, and prove that $\tilde{Y}_{\text{out}}(\ell, h) \approx \tilde{Y}_{\text{in}}(\ell, h)$. Thus, the average complexity of the second step is $\ell - s\sigma - t(1-\sigma) - \zeta(\ell, h)$, which can be approximated by

$$\ell - s\sigma - t(1-\sigma) - 2. \tag{6}$$

From (4) and (6) we obtain when $(s,t) \neq (0,0)$ the complexity of the attack[4] in terms of requests to the oracle:

$$\frac{2^\ell(1+s\sigma+t(1-\sigma))}{\binom{\ell}{h}A(\ell,h,s)+\binom{\ell}{h+1}B(\ell,h,t)}+\ell-s\sigma-t(1-\sigma)-2. \qquad (7)$$

### 4.2 Interpretation

If $\ell$ denotes the size of $K_u$ in the usual binary representation, then the probability that $\mu$ is a winning ticket is $\sigma = \frac{1}{2^\ell}\sum_{i=0}^{h}\binom{\ell}{i}$. In order to envisage an intuitive representation of the probability that a ticket is winning, we draw on Fig. 4 the probability according to the threshold $h$ when $\ell = 128$. In practice, the probability of a ticket being winning should be greater than $\frac{1}{100}$ otherwise the nodes would not collaborate: this implies that $h$ should be greater than 51. Fig. 5 represents the corresponding complexity of the attack, given by (7), when $s$ and $t$ are optimal. In the range of the practical values of $h$, the attack requires between 152 and 339 requests to the oracle in order to recover a key 128 bits long! Optimal values for $s$ and $t$ depend on $h$. When $h = 51$, the optimal complexity is obtained where $s = 5$ and $t = 0$.

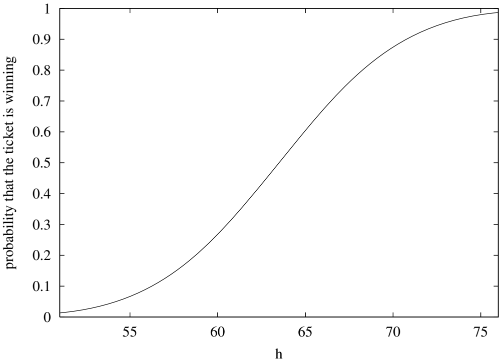


**Fig. 4.** Probability that a randomly chosen ticket is winning according to the value $h$

[4] Note that the algorithm which is given here aims to demonstrate that a practical attack is possible (if we can use oracle $\mathcal{O}$) but more sophisticated algorithms could further reduce the complexity. For instance, the attack can be improved if the calls to Alg. 1 are not independent. Indeed, consider the case where Alg. 1 is used with $t = 1$ and the two requests to the oracle are $\mu$ and $\mu^{(i)}$. If the protocol does not answer, we clearly have $\mathcal{O}(\mu) = \mathcal{O}(\mu^{(i)})$. We know, however, that if $\mathcal{O}(\mu)$ then $\neg\mathcal{O}(\overline{\mu})$ where $\overline{\mu}$ means that all the bits of $\mu$ have been flipped. We can now use this new value for the next call to Alg. 1, thus decreasing the number of calls to the oracle.

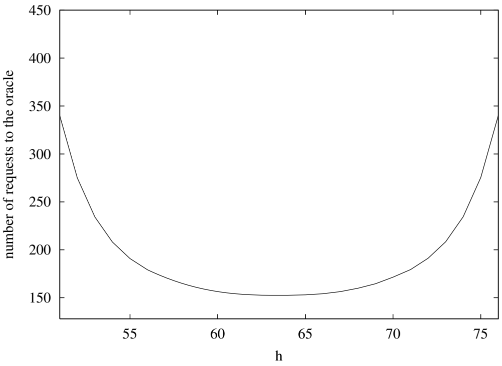


**Fig. 5.** Number of requests to $\mathcal{O}$ in order to recover the secret key of a given user

### 4.3 The Attack in Practice

In [2], the nodes do not send a reward claim as soon as they receive a winning ticket. Instead, they store them in a list $M$ which is encrypted before being sent to the accounting center. Consequently, an attacker is no longer able to match the values submitted to the node (requests to the oracle) with the sent claims (answers from the oracle). In other words, she no longer knows which of her value will generate a claim. Unfortunately, the attacker can still use an oracle even in this case; the attack consists of disturbing the input distribution of the node by sending beams of equal random values $\mu$ and then by analyzing whether the output distribution is disturbed, i.e., if the list of the reward claims is longer[5] or sent more frequently than usual. Indeed, if the random value $\mu$ forming the beam is such that $f(\mu, K) \neq 1$, then the output will not be disturbed. If not, the number of claims will be larger (length of packets larger than usual or packets more frequent than usual), meaning that the oracle answers true for this value. The length of the beam depends on the node environment and on the implementation of the protocol. Note that it is not necessary for the beam to fill the buffer; it merely has to sufficiently disturb the input distribution in such a way that the disturbance is detectable in the output distribution. Thus, the remainder of the buffer can be filled with random values. In this way, other (honest) users requesting the node to forward are helping the attacker by filling the buffer! Consequently, depending of the environment and implementation, the complexity of the attack remains proportional to the theoretical complexity given in Section 4.1.

### 4.4 Fixing the Scheme

Since the node has a buffer to store the winning tickets, one may think that in order to fix the scheme, it could reject random values that have already been

[5] [2] says that, even if an attacker cannot distinguish which ticket generates a reward claim, she can determine how many reward claims are sent.

stored. Indeed, the attacker is no longer able to send beams of equal random value. Unfortunately, it is possible to perform our attack in another way as follows. We assume w.l.o.g that the buffer is empty at the beginning of the attack[6]; for the sake of simplicity, we also assume that there are no other users making requests to the victim during the attack[7]. The attacker sends the victim the following beam:

$$\alpha^1, \alpha^2, \ldots, \alpha^{n-1}, \alpha^n$$

where the $\alpha^i$ are independent random values, until the node sends its list of claims (i.e., the oracle responds). In this case, the last sent value $\alpha^n$ is such that $f(\alpha^n, K) = 1$. The attacker wants now to check the value $\mu$ and sends for that the beam:

$$\alpha^1, \alpha^2, \ldots, \alpha^{n-1}, \mu$$

The $n-1$ first values fill the buffer, except the last space. Consequently either $f(\mu, K) = 1$ and therefore the node will send its claims or else $f(\mu, K) \neq 1$ which implies no answer from the node.

We feel that, whatever the patches applied, computing the Hamming distance between the secret key and another value in order to determine the winning tickets is not a good idea. The way that we suggest to fix the scheme consists of modifying the protocol such that the information that an attacker can obtain with the attack is rendered useless. Thus, we propose that a ticket is winning for $u$ if and only if:

$$d_{\mathcal{H}}(\mu, \text{hash}(K_u)) \leq h.$$

This technique has two advantages: when $K_u$ is kept in a tamperproof memory, only $\text{hash}(K_u)$ remains in the vulnerable memory; the attacker is able to obtain $\text{hash}(K_u)$, but the only thing that the attacker can do with this information is a *greedy ticket collection attack*, which is detected by the auditing center (see Section 2.7). Note that in [2], even if the key is encrypted with a password when the node is turned off, it has to remain permanently in clear in the non-tamperproof memory when the node is turned on. The second advantage is the lightweight of this solution because the hash value is computed only once instead of being computed for every packet. If the computational capabilities of the nodes allow a keyed-hash function to be carried out for each packet, then a more secure way would be to decide that a ticket is winning if and only if:

$$d_{\mathcal{H}}(\mu, \text{MAC}_{\text{hash}(K_u)}(\mu)) \leq h.$$

[6] The list is empty as soon as the user sends his list of claims. Note that even if the size of the buffer is not fixed, an attack is possible.

[7] This is not actually a problem since the attacker can accept the request instead of the victim, as explained in Section. 3.1, or if it is not possible, this disturbance will slightly increase the complexity of the attack, but the attack will still be possible — remember that the probability that the value of a "disturbing" requester generates a winning ticket is very low.

Note that again it is not the key itself which is in the vulnerable memory, but only the hash of the key. If one of these solutions is used to fix the protocol, the attacker can no longer use the node as an oracle.

## 5 Conclusion

In this paper, we have analyzed the security of the micro-payment scheme designed for asymmetric multi-hop cellular networks proposed by Jakobsson, Hubaux, and Buttyán. We have shown that the security of the scheme is compromised. Our contribution has mainly consisted of showing two attacks that entirely break the system in the sense that all the users' secret keys can be determined, with only a few hundred trials. This implies that an attacker can thus communicate freely, without being charged: the owners of the stolen keys are charged instead. We have suggested some lightweight but efficient modifications in order to repair the micro-payment scheme.

## Acknowledgments

I would like to thank M. Jakobsson, J.-P. Hubaux, and L. Buttyán, for their relevant comments on this work, as well as N. Ben Salem and S. Vaudenay for their helpful discussions. The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

## References

1. Thomas Haug. Overview of GSM: philosophy and results. *International Journal of Wireless Information Networks*, 1(1):7–16, 1994.
2. Markus Jakobsson, Jean-Pierre Hubaux, and Levente Buttyán. A micro-payment scheme encouraging collaboration in multi-hop cellular networks. In *Financial Cryptography – FC'03*, vol. 2742 of *LNCS*, pp. 15–33, Le Gosier, Guadeloupe, French West Indies, January 2003. IFCA, Springer.
3. Ying-Dar Lin and Yu-Ching Hsu. Multihop cellular: A new architecture for wireless communications. In *Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies – INFOCOM 2000*, vol. 3, pp. 1273–1282, Tel-Aviv, Israel, March 2000. IEEE.
4. Ronald Rivest. Electronic lottery tickets as micropayments. *Financial Cryptography – FC'97*, vol. 1318 of *LNCS*, pp. 307–314, Anguilla, British West Indies, February 1997. IFCA, Springer.

# Protecting Secret Data from Insider Attacks

David Dagon, Wenke Lee, and Richard Lipton

Georgia Institute of Technology
{dagon, wenke, rjl}@cc.gatech.edu

**Abstract.** We consider defenses against confidentiality and integrity attacks on data following break-ins, or so-called intrusion resistant storage technologies. We investigate the problem of protecting secret data, assuming an attacker is inside a target network or has compromised a system.

We give a definition of the problem area, and propose a solution, VAST, that uses large, structured files to improve the secure storage of valuable or secret data. Each secret has its multiple shares randomly distributed in an extremely large file. Random decoy shares and the lack of usable identification information prevent selective copying or analysis of the file. No single part of the file yields useful information in isolation from the rest. The file's size and structure therefore present an enormous additional hurdle to attackers attempting to transfer, steal or analyze the data. The system also has the remarkable property of healing itself after malicious corruption, thereby preserving both the confidentiality and integrity of the data.

## 1 Introduction

Security technologies have traditionally focused on perimeter defenses. By itself, this approach creates what has been called a lobster-model of security, or "a sort of crunchy shell around a soft, chewy center" [Che90]. If an attacker manages to get into the network, it becomes very difficult to detect or prevent further security compromises.

This has prompted the development of secure storage techniques that resist against successful attacks. This paper studies the problem of protecting secret data under the assumption that an attacker has already broken through the network perimeter (or is an "insider"). We give a formal definition of the problem, and present one solution called VAST. The key idea is to distribute secret data in an extremely large storage system without exploitable identification information. Our VAST storage system is orthogonal and complimentary to existing data protection techniques, such as encryption, in that it makes attacks much more difficult to succeed.

In this paper, we describe the design rationales, data structures and algorithms. We also describe an implementation of such a system, to demonstrate acceptable normal use. Specifically, we make the following contributions:

**Definition of Secure Storage Problem.** We formally describe the problem of secure storage of secrets in Section 3.1. We describe an abstract data type that is a large storage table composed of records. Operations include initialization, insertion and deletion. We also describe security properties that the table and operations must guarantee. This

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 16–30, 2005.
© Springer-Verlag Berlin Heidelberg 2005

general description of the problem formalizes intrusion resistant systems, and encourages further research into this general problem area.

**Storage Scheme for Secret Data.** Based on the abstract data type, we propose the VAST storage system, which uses extremely large (e.g., terabyte-sized) files to store secret information. In VAST, a secret is broken into shares distributed over a large file, so that no single portion of the file holds recoverable information.

## 2 Related Work

VAST of course fits into the larger field of fault-tolerant systems generally, and intrusion-tolerant systems specifically. There has been a considerable amount of work on tolerant and dependable file storage. Many works have used secret sharing as part of a resilient data storage system [WBS$^+$00, LAV01]. Of particular relevance is [FDP91, FDR92], and Rabin's work in [Rab89], which all used secret sharing as an information dispersal technique for security and redundancy. Our work is in a similar vein, but seeks to leverage tradeoffs between disk I/O speeds and memory on *local* file stores, without the need to distribute shares among hosts.

Many other intrusion resistant systems have used *fragmentation-and-scattering*, a technique similar to VAST's hashing store of secret shares. In [DFF$^+$88], the SATURNE research project describe the fragmentation-and-scattering scheme. Stored data was cut into insignificant fragments, and replicated over a network. The distribution of the fragments obliged attackers to compromise numerous resources before they could read, modify or destroy sensitive data. Instead of distributing resources over a network, VAST keeps all fragmented data in a single file, albeit usually spread over several drives.

The tremendous time difference between memory and drive I/O has motivated work in complexity analysis [AKL93, AV87]. The general goal of these works is to describe a lower bound on algorithms and demonstrate a minimal number of I/O operations. VAST works in the opposite direction, and seeks to maximize the number of required I/O operations to slow attackers.

Components of VAST were inspired by other results. For example, the large table in VAST is similar in principle to the solution in [Mau92], where a short (weak) key and a long plaintext were kept secure by using a publicly-accessible string of random bits whose length greatly exceeded that of the plain text. In [CFIJ99], the authors created a very similar model for memory storage, and generally described how to create a storage system that can forget any secret. Their solution assumed the existence of a small and fixed storage area that the adversary cannot read, which differs from VAST's large, unfixed, and readable storage tables.

Other areas of research have used techniques similar to VAST. VAST's distribution of shares over a table has a superficial resemblance to data hiding [KP00]. VAST's ability to recover and heal corrupted messages also resembles Byzantine storage systems [MR98], or even the larger field of error correction codes. VAST combines existing approaches in a new and interesting way.

## 3 Designing Large Files for Valuable Data

Below, we describe an abstract secure storage problem, and suggest relevant design considerations for any solution. We then propose the VAST storage system, and detail its operation.

### 3.1 Secure Storage Problem Statement

For this paper, we address the following specific scenario: Assuming an attacker has penetrated a storage system, what reasonable measures help prevent the compromise of stored secret data through brute-force analysis, such as key cracking, dictionary password guessing, and similar attacks?

We formally describe the secure storage of data in large tables as follows. A large table $T$ has parameters $(n, r, m, d, K)$. The table is used to store $n$ records of $r$ bits. The table itself is $m$ bits in size, where $m \geq nr$, and usually $m \gg nr$. The value $d$ determines a fraction of the table, $0 < d \leq 1$. The value $K$, described below, is a threshold used to measure security properties. The table supports the following operations.

1. **Initialize.** An $init()$ function iteratively initializes each of the $n$ records in $T$.
2. **Add.** An $add()$ operation inserts data into the table.
3. **Delete.** A $delete()$ operation removes entries from the table.
4. **Find.** A $find()$ operation retrieves information from the table.

The security property of the table is the following statement. Suppose we initialize the table and then perform a series of insertion operations. Next, suppose we use only $dm$ bits from the table. Given a value $x$, and using only $dm$ bits, the probability one can can correctly compute $find(x)$ is at most $2^{-K}$. In other words, if $dm$ bits are stolen or analyzed, there's only a small chance that $x$ can be recovered from the exposed portion of the table. We can also state a stronger security property for the table, so that it also provides semantic security. Again assuming only $dm$ bits are used, the semantic security property holds that one cannot compute $find(x)$ correctly, and further cannot obtain one bit of $x$ with any advantage over $\frac{1}{2} + 2^{-K}$.

It is not obvious that one can create a table with these properties. Reasoning about the problem points to one possible solution. To start, we know that our overall goal is to increase $K$, which minimizes the probability of a successful attack. One strategy to accomplish this is to encrypt the data, $x$, inserted into the table, since this makes linear scans of the table much more difficult, and forces the attackers to perform brute-force attacks on the encryption scheme. A second strategy is to not only increase $m$, but also to distribute $x$ in such a way that $d \approx 1$ before recovering $x$ becomes possible. In other words, we should store data in a large table such that analyzing a small $d$ fraction of the table cannot yield $x$.

An additional, practical benefit derives from using a large table size, $m$. If the table is large, and $x$ is stored such that $d$ must be near 1, then in practical terms this means analyzing the table's $m$ bits will require enormous resources. We know, for example, that I/O access is extremely slow compared to memory access [AKL93]. We therefore should design our table with a goal opposite of Vitter's work minimizing I/O operations in algorithms [AV87]. Instead, we wish to *maximize* the I/O operations (and therefore, the time) required for analysis.

The above discussion suggests making the table size large. One consequence is that an attack will take more time to succeed. With I/O operations *several* orders of magnitude slower than memory access [HP03], this means analysis will require repeated disk access.

### 3.2 Design Considerations for Secure Data Storage Problems

In most attacks on data confidentiality and integrity, the attacker first needs to get hold of the target data, usually by copying it offsite. In this attack set up stage, time is proportional to the size of data. For example, if the attacker needs to transfer data on a link with a capacity of $C$ data units per unit time, then the time it takes to transfer data with size $D$ will be $T = \frac{D}{C}$. If the target data is actually small in size, we better protect the data by dispersing it in a large storage file without any "exploitable" identification information. This will force the attacker to process the entire large storage to recover the target information. If the table size $m$ is tera-scale, the time needed to steal the file is potentially prohibitive.

In order to slow the attack, we need to force it to carry out more operations. For attacks on confidentiality and integrity, a simple protection scheme is to fragment the data and distribute the shares throughout the large file. Thus, for each attack (trial) that involves locating shares and guessing (brute-force analyzing the data), instead of spending time $T$ for one target, it now must spend time $kT$ if $k$ fragments are needed to reconstruct the data.

### 3.3 The VAST Storage System

We now describe the design of our large file scheme, using a credit card database storage system as a motivating example. User financial records are stored in a file, and retrieved using keys, passwords or PINs that hash to appropriate table entries. (Without significant modification, the system could be used in almost any password-based authentication system.) A readable metadata index file stores the relevant information for each user, including user name, *u*, and salts $s_1$, $s_2$, ..., $s_k$, each a random number. The metadata identification (or user identity) file does not need to be read-protected because it contains no secret. (In practice, of course, one might elect to restrict access to this file as well; however, our analysis presumes it has been accessed by an attacker.)

The data storage file is a very large table $T$ with $m$ entries in which multiple shares of data are randomly distributed. There are no empty entries because the table is initially filled with random bit strings that look like valid shares.

We next study the data structures and algorithms for the large table file. The main design goals are:

***Functional.*** From the functionality point of view, the table must store financial information reliably so that the data is retrievable only when a proper key is presented. This corresponds to the $add()$ and $find()$ operations noted in section 3.1.

***Secure.*** From the data security point of view, the design objective is to make it very difficult and slow for an attacker to steal the large information file and extract the information using brute-force key guessing or dictionary attacks. That is, it costs the attacker maximally (in time, or other resources) with each guess. This corresponds to the security principle noted in section 3.1.

Below, we discuss how to achieve these goals.

**Storing Unguessable Shares of Random.** In order to force the attacker to read all shares with each guess, VAST is based on secret sharing [Sha70]. Financial data for each user is stored under their unique name, $u$, in a large table. The data is accessed through the use of a key, $key$, and $k$ random salts, $s_1, s_2, \ldots, s_k$.

To add a user and data into the system (the $add()$ operation in Section 3.1), we first take the user's financial information (e.g., a credit card number) $M$, and add any needed padding to match the length of $X_2$, a large random number selected for each insertion of $M$. We will use $X_1$ to refer to the padded information $M$. Together, $X_1$ and $X_2$ may be considered as a message and Vernam's one-time pad [Bis03]. As will be seen below, portions of this cipher scheme are stored in the table. We selected a one-time pad because its provable security was attractive, and helps partially address problems found in hash-storage schemes, such as dictionary attacks on weak passwords. The use of the pad also avoids problems associated with storing message-derived hashes in the metadata table, e.g., theft of hashes, and offline guessing attacks against messages with similar structures, such as credit cards. (We discuss an attack model below.)

$X_1$ and $X_2$ are of equal length, on the order of 128 to 160 bits or more. The numbers are XOR'd together to produce a third value, $X = X_1 \oplus X_2$. The random number $X_2$ is then appended to the user's entry in the identity file, along with user name $u$ and a set of salts, $\{s_1, \ldots, s_{k'}, \ldots, s_k\}$, each a unique random number.

Instead of storing the padded message $X_1$ in the table, we first encrypt it with a symmetric encryption operation, $E_{key}(X_1)$. (Any symmetric encryption system can be used.) In addition to improving security, the encryption step also makes it easier to generate convincing initial (random) values for unused portions of the table.

Then, applying Shamir's secret sharing scheme [Sha70], two random polynomials are constructed to secret share $E_{key}(X_1)$ (the encrypted message) and $X$ (the cipher text):

$$f_1(x) = E(X_1) + \textstyle\sum_{j=1}^{k'-1} a_j x^j \pmod{q}, \quad f_2(x) = X + \textstyle\sum_{j=1}^{k'-1} b_j x^j \pmod{q} \quad (1)$$

We select $q$, a large prime number (greater than $k$, $X_1$ and $X_2$), and store it in the metadata file. The coefficients $a_j$ (and likewise $b_j$) $j = 1, 1, \ldots, k' - 1$, are random, independent numbers in the range of $[0, q-1]$. We use $k' \leq k$ to provide collision tolerance because $k'$ shares are sufficient to reconstruct the secret. Thus, for $k$ shares, the threshold of $k'$ shares must be present to recover the secret. For each $i = 1, 2, \ldots, k$, we store both $f_1(i)$ and $f_2(i)$ in the same table entry at $H(key||s_i) \bmod m$. These shares look just like any other random numbers in the range $[0, q-1]$. Therefore, at initialization (the $init()$ operation in Section 3.1), the table is filled with random numbers in the range of $[0, q-1]$. After the shares are inserted in the table, the coefficients of the two polynomials (Equations (1)) are discarded. Figure 1 provides an overview of the process.

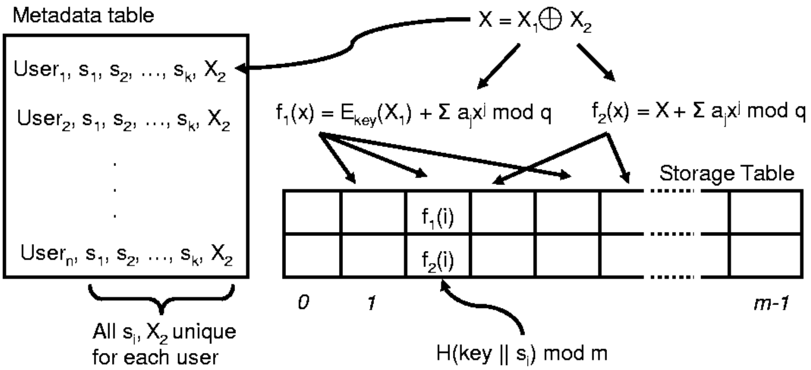

Metadata table
User1, s1, s2, ..., sk, X2
User2, s1, s2, ..., sk, X2
Usern, s1, s2, ..., sk, X2
All si, X2 unique for each user
X = X1⊕ X2
f1(x) = Ekey(X1) + Σ ajxj mod q
f2(x) = X + Σ ajxj mod q
Storage Table
f1(i)
f2(i)
0
1
m-1
H(key || si) mod m

**Fig. 1.** *Overview of VAST System* Information or a message, $X_1$, is $\oplus$-combined with a random number $X_2$ to form $X$. The random number $X_2$ is stored in a metadata table under the appropriate user's entry, along with random salts, $s_1, s_2, \ldots, s_k$, unique for each user. The values $E_{key}(X_1)$ and $X$ are Shamir-shared to derive $k$ shares, $f_1$ and $f_2$. Each $f_1(i)$ and $f_2(i)$, which are stored in the large table, based on a hash of the key and salt, at table entry $H(key||s_i) mod\, m$

To retrieve information for a user $u$ (the $find()$ operation in Section 3.1), we interact with the user to obtain the key, password or PIN, called $key'$, and look up the salts in the metadata file. Then, we retrieve the shares $f_1'(i)$ and $f_2'(i)$ in the table entry $H(key'||s_i) \bmod m$, for each $i = 1, 2, \ldots, k$. Given $k'$ shares, say $i = 1, 2, \ldots, k'$, the polynomial $f_1$ (and likewise $f_2$) can be reconstructed using Lagrange interpolation:

$$f_1'(x) = \sum_{i=1}^{k'} f_1'(i) \prod_{1 \le j \le k', j \ne i} \frac{x-j}{i-j} \pmod q \tag{2}$$

Thus, $X_1'$ (and likewise $X'$) can be computed:

$$X_1' = f_1'(0) = \textstyle\sum_{i=1}^{k'} c_i f_1'(i) \pmod q, \quad \text{where } c_i = \prod_{1 \le j \le k', j \ne i} \frac{j}{j-i} \tag{3}$$

We then perform decryption, $X_1' = E_{key}^{-1}(E_{key}(X_1'))$. If $X_1' \oplus X' = X_2$ (the value stored with $u$ in the metadata file), then the key was valid, and the correct message $X_1$ was recovered. If $X_1' \oplus X' \ne X_2$, this may be due to collisions (i.e., some shares overwritten by the shares of another user), and another $k'$ shares can be used to compute $X_1'$ and $X'$ as in Equation (3). In the worst case, one needs to try $\binom{k}{k'}$ times before the key is validated. However, since collisions are very rare, the probability of success (in validating a valid key) with the first $k'$ shares is very high.

Suppose an incorrect key $key'$ is supplied. Then $k'$ incorrect shares $f_1'(i)$ and $f_2'(i)$, $i = 1, 2, \ldots, k'$ are read to construct $X_1'$ and $X'$. The chance of $X_1' \oplus X' = X_2$, and thus validating the incorrect $key'$ , is very small, $2^{-128}$ if $X$ is a 128-bit random. This is because for the $X'$ value computed from the shares, $X_1'$ must happen to be exactly $X' \oplus X_2$, which in turn requires that one share, say the $k'$th share, for $X_1'$, must be $X \oplus X_2' - \sum_{i=1}^{k'-1} c_i f_1' i \pmod p$, a $2^{-128}$ chance. Thus, VAST meets the security property for storage tables stated in Section 3.1.

An attacker may attempt to search for the shadow keys in $T$, since every data element has at least $k'$ shares in the table. But searching for the correct $k'$ elements in $m$ is difficult, on the order of $\binom{m}{k} \geq (\frac{m}{k})^k$, where $m$ is enormous. (Recall, the table is tera-scale, often with $2^{40}$ or more entries, all initially random.) The attacker's best strategy is key guessing, since the search space is much smaller. Even if 8 character keys are composed from 95 possible characters, it is easier to guess out of $95^8 \leq 2^{56}$ combinations, compared, say, to $\binom{2^{40}}{8} \geq 2^{296}$, for $k = 8, m = 2^{40}$. So, the attacker can only perform key guessing.

Now consider an attacker attempting to guess the key of user $u$ to retrieve the financial data, $X_1$. If she can precompute the shares of $X_1$ and $X$, then for the guessed key *key'*, she might just check the shares in one entry, say $H(key'||s_1) \bmod m$, (or up to $k - k'$ entries) to learn that *key'* is incorrect. However, we can show that this is not possible. First, although she can read the identification file and hence the random $X_2$, she cannot figure out the values of message $X_1$ and cipher text $X$ because encryption using one-time pad offers perfect secrecy. Furthermore, the coefficients of the polynomials in Equations (1) are random and are discarded. Therefore, there is no way the attacker can precompute the shares. So, she has to read the shares from the table. If she reads fewer than $k'$ shares, according to Equation (3), she will not be able to compute $X_1'$ (and likewise $X'$). And without $X_1'$ and $X'$, she cannot test if $X_1' \oplus X_2 = X'$ to know if *key'* is correct. Based on the above analysis, and the strengths of the basic cryptographic primitives of one-time pad and secret sharing, we have the following claim:

*Property 1.* In order to retrieve a user's information or just to learn that the key is incorrect, at least $k'$ table entries must be read.

When collisions occur, and enough of the salts still point to valid shares, the system has detected some corruption of the table. In other words, some of the shares are invalid, but enough are still present to recover $X_1$ under the Shamir secret sharing scheme. This could be due to collisions as other data is added to the table, or because of malicious corruption of the file. In either case, if $X_1$ has been retrieved using only $k' < k$ salts, new random salts are generated, and written to the metadata file. The data is then rehashed and written to the table. This way, the table "heals" itself and corrects corruption detected during reads. Thus, when data collides with other entries, we eventually detect this problem, and relocate the shares. This movement may cause other collisions, but the chance is small. Eventually a steady state is obtained, and no user's shares collide with shares of any other. Section 4 discusses the reliability of this system, and the small probability of collisions occurring.

In order to completely corrupt a secret stored in the table, at least $k - k' + 1$ entries must be overwritten. The chance of this occurring with random writes is extremely small, on the order of $\frac{k-k'+1}{m}$, where $m$ is enormous. Section 4 provides a complete analysis of the reliability of the system. However, if any legitimate read access occurs prior to all $k-k'+1$ collisions, the corruption will be detected and repaired. (Recall that only $k'$ shares must be valid, so $k - k'$ corrupted shares can be detected and corrected in the course of legitimate use.) This property allows us to assert the following claim:

*Property 2.* Since reads from the table reveal any collisions, allowing for repair of the data's integrity, data is destroyed only if $k - k' + 1$ shares are corrupted between legitimate access attempts.

This is an important property for storage systems, since attackers unable to recover data from the file may nonetheless maliciously write bad information, in order to corrupt the file for normal use. (For example, they might randomly write zeros to the table.) With a large tera-scale file, however, successfully deleting all information would take an enormous number of writes, and may risk detection by other orthogonal detection systems.

The size of the financial data, $M$, stored as $X_1$ using the above scheme is of course limited [CSGV81]. We've used credit card information as a motivating example. However, there are many ways we can extend our scheme to store arbitrarily large files. One simple scheme is to treat each encrypted block of the whole encrypted message as an $M$ of user $i$. In other words, we could make as many users are there are blocks, so that a large $M$ is distributed or chained over many users.

No doubt other variations are possible. One can be creative about using pointers, indices, or set orders to store even large amounts of data. Therefore, while credit card number storage provides a real-world motivation for our work, our scheme can be extended to provide more general support for a wide range of applications. Tera-scale drives are now affordable, and we encourage others to examine how fragmentation-and-scattering schemes can be improved with large data stores.

**Table Tiers.** We also briefly note a possible variation of VAST using table tiers to efficiently use limited resources. While tera-scale drive storage is inexpensive, greater reliability may be obtained by dividing a quantity of storage into separate independent VAST tables.

Recall the important design goal of providing *reliable* storage for sensitive information. As will be discussed in 4.2, there is a small chance that collisions may occur when inserting new shares into a table. So, in addition to using a lower threshold for validating retrieved information, $k' \leq k$, one can simply make additional tables, each holding the same user information, but distributed with independent sets of salts. Thus, a 4-terabyte storage system can be broken into 4 1-terabyte storage systems, each with an independent chance of failure.

Using separate *independent* sets of salts over many separate tables is analogous to the practice of using drive backups from different manufacturers and models, in order to ensure that the hardware failure rates are truly independent. So, by adding tiers of tables, one can reduce an already small chance of failure into an infinitesimal risk.

## 4 Security Analysis

By storing secret data in a large, structured file, attackers are forced to copy and analyze the entire terabyte-sized file as a whole. No single portion of the file yields useful information in isolation. Below, we evaluate the improved security provided by VAST, and the reliability of the system.

### 4.1 Cost of Brute-Force Attacks

Below, we analyze the solutions VAST provides, namely (a) reliable and efficient retrieval of stored secrets, and (b) greater defense against key-cracking attacks.

**Attacks in General.** Broadly, attacks on storage files fall into two categories: on-line attacks and off-line analysis [PM99, Bis03]. The on-line analysis of keys is difficult in VAST for several reasons. First, scanning the hash file in a linear fashion does not provide the attacker with any information about which entries are valid hash stores. (Recall that unused entries are initialized with random bits, and data is stored in encrypted shares, which also appear random.) Interestingly, all of the $k$ Shamir secret keys are present in the same file; however, the attacker has $\binom{m}{k}$ possible combinations. Recall that $m$ is enormous, say in the range of $2^{40}$, and $k$ is not negligible, say in the range of 8-10. So $\binom{m}{k} \geq (\frac{2^{40}}{8})^8 \geq 2^{296}$, and the presence of all the shares on the table $T$ does not help the attacker more than guessing.

Since sequential or adjacent shares on disk may be read more quickly than shares distributed on random parts of the drive, an attacker may attempt to precompute numerous hashes for key guesses, and upload the sorted precomputed indices. That is, an attacker might compute, using a dictionary $D$, with $P$ permutations per word, some $\{|D| \cdot P \cdot nk\}$ hashes offline, and sort them by index value to improve drive access times, since many shares for many guesses will be adjacent, or at least within the same logical block on disk. (Recall, for example, that drive reads from adjacent locations on disk may be faster that reads from non-adjacent tracks and sectors [HP03].) However, if the VAST system is properly bandwidth limited, the attacker will find this slow going as well. The minimal space needed to request a single share is 8 bytes. Assuming a dictionary of just ten thousand words is used, with only a hundred permutations per word, the attacker would have to upload approximately 8 megs for each user *and* each salt. Because VAST systems are deployed on low-bandwidth links, this could potentially take a long time, and could easily be detected. Even if the attacker somehow uploaded the precomputed indices, they still have to obtain the $k$ shares and find if any $k'$ subset solves a polynomial to recover $X_1$ and $X_2$.

Without sufficient resources on-line, an attacker's preferred strategy would be to transfer the hash file for off-line for analysis. Assuming an attacker somehow transfers a tera-scale file offsite for analysis, the size of the file presents a second hurdle: repeated I/O operations.

Disk access takes on the order of 5 to 20 milliseconds, compared to 50 to 100 nanoseconds for DRAM. While drives are getting faster, they are on average 100,000 times slower than DRAM by most estimates [HP03], and are expected to remain relatively slow [Pat94].

Given this, Anderson's formula [Bis03] can be used to estimate the time it would take to check $N$ possible keys, with $P$ probability of success (of one key guess) and $G$ guesses performed in one time unit: $T = \frac{N}{PG}$. To perform an exhaustive key space search, an attacker might load some of the hash file into memory, $m'$, while the bulk of it, $m - m'$, must remain on disk. For those key guesses that hash to a memory store, the attacker would enjoy a fast lookup rate on par with existing cracking tools. But most of the time, the attacker would have to read from disk. Since VAST's indexing schema uses hash operations that provide uniform dispersion, the ratio of memory to disk is applied to the rates for drive and memory access. We assume that the time required for a disk-bound validation operation is a factor of $L$ of the time for memory-bound operation, and let $r = \frac{m'}{m}$. We can then modify the guess rate $G$ in Anderson's formula to reflect the rate for disk access, so it becomes $G(r + (1 - r)L)$. Since $k'$ shares must be read to validate a guessed key, the guess rate is further reduced to $\frac{G(r+(1-r)L)}{k'}$. We thus have the following claim:

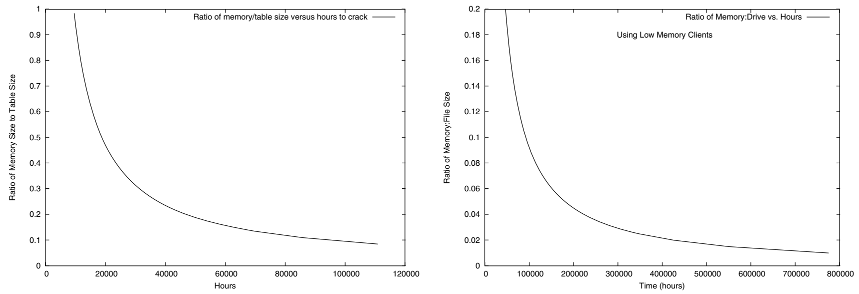
*Property 3.* In the VAST system, the time taken to successfully guess a key (with probability $P$) is:

$$T = k' \frac{N}{PG(r + (1 - r)L)} \tag{4}$$

In this light, existing encrypted file schemes are just a special case of the VAST system with $r = 1$ and $k' = 1$, and a much smaller $m$. Our objective is to make $T$ as high as possible. If we make the table very large, $r$ is close to zero, then Equation (4) is close to $T = k' \frac{N}{PGL}$. This means then the deciding factor is $L$, or the time required for disk access.

Our implementation of a single-CPU cracker resulted in a rate for memory-bound operations of just over 108,000 hash operations per second, while the disk-bound guessing yielded approximately 238 hash operations per second. No doubt, different hardware will produce different results. But on the whole, systems designers note that disk access is at least 100,000 times slower than accessing memory [HP03], i.e., $L = \frac{1}{100,000}$, so one might expect the ratio of $L$ to improve only slightly [Pat94].

Using the modified Anderson's formula, we can estimate progress on a single machine, making the conservative assumption of a key alphabet of 95 printable characters,



(a) Guess Rate (b) Low Memory Client Guess Rate

**Fig. 2.** Figure (a) shows how the ratio of memory to table size affects guess rates for key cracking. The graph assumes 6 character keys selected from 95 printable characters, and 5 salts per user, and $m = 2^{40}$ entries. Reasonable progress is only possible when memory size is large enough to hold the entire table. Figure (b) shows the guess rate when low-memory clients are used, effectively zooming in on a portion of figure (a). With less memory, the guess rate is consistently slow. Administrators can force attackers into a low-performance portion of the curve just by adding inexpensive additional drives

merely 6 character keys, and only five salts per data item. Figure 2(a) plots the time it takes to guess a key, as a function of the ratio of memory to disk size. If one has a 1:1 memory disk ratio (i.e., a terabyte of memory, approximately \$1.6 million [FM03]), the cracking time still requires over 9,500 hours–about 13 months. We presume that most attackers will have less than a terabyte of memory available. In such a case, their rate of progress is significantly worse–on the order of hundreds of thousands of hours.

Administrators worried about distributed cracking tools have a simple and effective defense: just grow the hash file. Space does not permit a complete discussion of table growth, but an intuitive approach is to place the old table within the larger table, and rehash each user into the larger space when they access their stored message.

Note that there are several orders of magnitude in price difference between drives and memory. This means that if adversaries attempt to match the size of the storage table with more memory, an administrator merely needs to buy more disk space. For a few thousand dollars, administrators force the attackers to spend millions to match the size of the table. This is an arms race attackers cannot easily afford.

### 4.2 Reliability Analysis

When shares are written to the table, there exists a chance that valid entries may be overwritten by shares for another data item. The probability of no collision *whatsoever* when inserting a total of $n$ items, each with $k$ shares, is computed as:

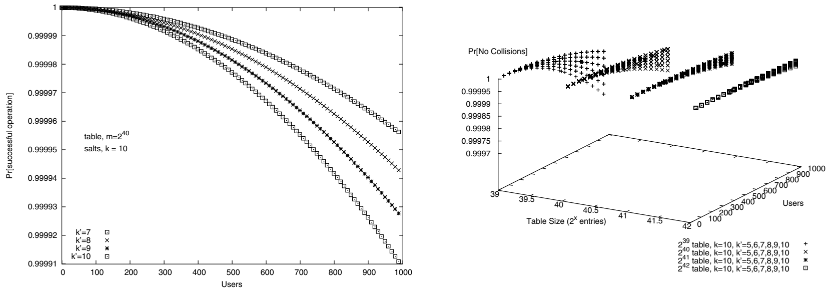
$$P_0 = \prod_{i=0}^{nk-1} \left(1 - \frac{i}{m}\right) \tag{5}$$

For practical purposes, we assume hash values are independent for a good-enough secure hash function. We can use the Equation (5) to compute for a desired probability, say $99.9999\%$, how many data elements (each with some $k$ hashes) can be stored in a table with size $m$.

We can relax the matching requirement a bit, as long as the data has $k' \leq k$ shares in the table, the data can be retrieved. That is, for each element, we allow at most $l = k - k'$ of its shares to be overwritten by other write operations. Intuitively, we can then accommodate more data using the same table while achieving the same desired (low) probability of rejecting a valid key. The exact calculation of $P_l$, the probability that each data item has at least $k'$ valid shares (i.e., no more than $l$ shares are overwritten), is very complicated. For simplicity's sake, we can compute the lower bound of $P_l$. We use the following:

$$P_l' = \prod_{i=0}^{n-1} \prod_{j=0}^{k-1} \left(1 - \frac{ik' + j}{m}\right) \tag{6}$$

This can be interpreted as: when inserting the $k$ shares for the $i$th data item, avoid the first $k'$ valid shares for each of the ($i$-1)th items already in the table, and the $k$ shares of the $i$th item themselves do not overwrite each other, (i.e., there is no self-collision.) It is easy to see that this calculation does not include other possible ways



(a) Collisions

(b) Table Size

**Fig. 3.** a) The number of user data entries in a table versus the chance that no collisions occur, for a table with $m = 2^{40}$ entries, and ten salts per data item. By tolerating a few collisions, $k' < k$, higher reliability is achieved. b) The relationship between table size, item count, and successful operation. For small tables, variations in $k'$ may be necessary to improve reliability. More tables can also be added cheaply to improve performance. Alternatively, one can restructure the table into tiers

that can lead to the condition where each item has at least $k'$ valid shares. Therefore, $P_l'$ is a lower bound of $P_l$, i.e., $P_l \geq P_l'$. It is obvious that $P_l' \geq P_0$. Therefore, we have $P_l \geq P_0$. For large $m$ and small $l$, $P_l'$ is very close to $P_l$. We thus use this simple estimation.

Figure 3(a) shows the benefit of allowing some collisions (up to $k = k'$ to occur). As more data is added, there's an increasing chance that one item will suffer more than $k - k'$ collisions. At some point, the risk of such failure becomes unacceptable, and larger tables or table tiers must be used. One may be tempted to lower $k'$ even further. However, recall that if $k'$ is too low, an adversary has a greater probability of stealing a portion of the file and obtaining all of the required shares. Specifically, if only $z$ bytes are stolen, there is a $(\frac{z}{m})^{k'}$ chance of all an item's shares are exposed.

Conceptually, it is best to fix an error rate, estimate the maximum number of data items, and design the file size accordingly. Figure 3(b) shows the flexibility of each parameter. To obtain a fixed error rate, one can increase the size of the table. One can also adjust $k$ and (to a lesser extent) $k'$ to achieve the desired error rate. If one is constrained by a drive budget, and cannot find a configuration with an acceptable reliability, then table tiers provide a solution.

The VAST system also addresses the problem of malicious data corruption. If an attacker does not attempt to read the secret data, but merely tries to delete it, VAST provides two defenses. First, the attacker does not know where the secret shares are stored, so the attack must corrupt nearly a terabyte to have a chance of success. Second, if the attacker merely corrupts a fraction of the storage table, subsequent reads can detect the errors, and create new salts, thereby "healing" the table with each read. In a normal secret storage system (e.g., a password-protected file), the attacker merely has to change as little as one byte to damage the file.

### 4.3 Efficient Legitimate Use

To fully evaluate a security enhancement, the increased cost of an attack should be balanced against the costs imposed on legitimate use. An implementation and testing of VAST shows that it can efficiently handle many data retrieval operations per second. Each operation involves a hash computation, a seek and a read from disk. Even though retrieving information may require up to $k$ disk reads, in practice the number of salts is small enough to make this efficient. In our tests, when all table operations require drive access, the number of operations is limited to around 250 per second per drive, using a slow (5400 rpm) IDE drive. Thus, when using low-end equipment there is an upper limit to how many records can be retrieved at a time. If one anticipates more than $250/k$ simultaneous reads, then the hash store may use faster drives, or could be distributed over a RAID system.

An important observation is that, once completely I/O bound, the performance of VAST does not decrease with larger tables. Figure 4 shows that with small tables (unacceptable from a security point of view), a good portion of the file can be cached by an operating system's I/O buffers. As a result, reads are quick, and hundreds of thousands of hash validations can be performed per second. As tables grow in size, particularly at around $2^{25}$ entries an above, the majority of the hash file then resides only on disk, and performance degrades. With large files, I/O comes to dominate the $find()$ operation time (which includes both I/O and memory operations for decryption and share recovery). Thus, the performance does not degrade further. Eventually, a steady rate is reached as the OS block cache becomes dominated by the drive seek time. So, one may add more terabytes to a hash store without lowering performance further. In fact, in our

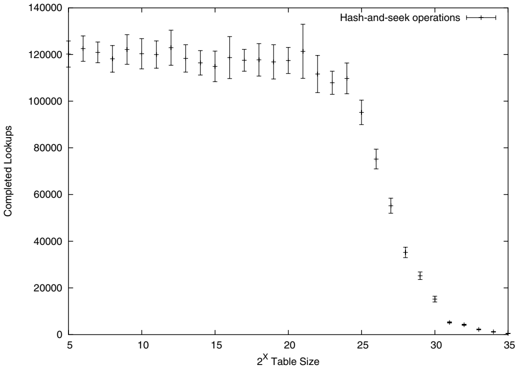


**Fig. 4.** Performance of a VAST system deployed on FreeBSD, retrieving random records. Due to the unpredictable location of shares on disk, and coincidental proximity of some hashes in single (8K) blocks, performance varied. Plots show the mean number of hash-seek-read operations, with standard error, compared to table size. In practice, one would use a terabyte-sized file. But the output for smaller-sized files is included to show how memory greatly speeds up performance. Significantly, even though performance degrades for larger files, it reaches a minimum of no less than 250 operations per second. Thus, one may add more terabytes to an I/O-bound VAST system, and expect no further performance degradation

testing we observed a very slight increase in performance with the addition of each new drive since each spindle provides its own independent service rate.

One might be concerned about the efficiency of reading any $k'$ subset of $k$ shares. That is, if the authentication phase must find the right $k'$ of $k$, it could potentially take $\binom{k}{k'}$ operations. In practice, however, the first $k'$ of the $k$ shares will almost always provide a correct match. Even under considerable load, the system may be designed to perform with 99.9999% success. And since $k$ and $k'$ do not differ much and are small, around 10-15, the rare worst case scenarios will not take an unreasonable amount of work to complete.

## 5 Conclusion

Despite the best efforts of systems administrators, storage systems will become vulnerable, and attackers will sometimes succeed. The VAST system provides a way to store information that resists successful penetrations. In addressing this problem, this paper contributed the following points.

First, we studied the problem of protecting secret data storage against insider attacks, and formally defined it as the problem: How to store data in a table such that no fraction of the table yields useful information? Reasoning about this problem suggested the use of large storage systems to minimize the attacker's chance of success, and to increase the cost of attack.

We then proposed the VAST system as one possible solution to the secure data storage problem. Each secret has its multiple shares randomly distributed in an extremely large file. Random decoy shares and the lack of usable identification information prevent selective copying or analysis of the file. No single part of the file yields useful information in isolation from the rest. The file's size and structure therefore present an enormous additional hurdle to attackers attempting to transfer, steal or analyze the data.

Finally, we implemented the VAST system, and demonstrated that it performs reasonably well for normal use. Experiments show that breaking VAST requires an enormous amount of time and resources. Under our security model, VAST greatly improves the security of data storage as well, since attacks are likely to trigger an alert and response. Unlike previous work, e.g., [FDP91, FDR92, DFF+88], VAST requires only a single host, and presumes an attacker may access the protected file.

Using large files to safely store data is a counter-intuitive approach to security. VAST demonstrates how algorithms that maximize the number of I/O operations can be used to improve security, similar to traditional fragmentation-and-scattering schemes. With affordable tera-scale storage devices, we believe solutions to the table storage problem now have many practical applications.

## References

[AKL93] Lars Arge, Mikael Knudseni, and Kirsten Larsent. A general lower bound on the complexity of comparison-based algorithm. In *Proceedings of the 3d Workshop on Algorithms and Data Structures*, volume 709, pages 83–94, 1993.

[AV87] A. Aggarwal and J.S. Vitter. The i/o complexity of sorting and related problems. In *Proc. 14th ICALP*, 1987.

[Bis03] Matt Bishop. *Computer Security: Art and Science*. Addison-Wesley-Longman, 2003.

[CFIJ99] Giovanni Di Crescenzo, Niels Ferguson, Russell Impagliazzo, and Markus Jakobsson. How to forget a secret. In *Proc. of STACS 99*, 1999.

[Che90] B. Cheswick. The design of a secure internet gateway. In *Proc. of Usenix Summer Conference*, 1990.

[CSGV81] R.M. Capocelli, A. De Santis, L. Gargano, and U. Vaccaro. On the size of shares for secret sharing schemes. *Lecture Notes in Computer Science*, 576, 1981.

[DFF$^+$88] Y. Deswarte, J.C. Fabre, J.M. Fray, D. Powell, and P.G. Ranea. Saturne: a distributed computing system which tolerates faults and intrusions. In *Workshop on the Future Trends of Distributed Computing Systems in the 1990s*, pages 329–338, September 1988.

[FDP91] J.-M. Fray, Y. Deswarte, and D. Powell. Intrusion tolerance using fine-grain fragmentation-scattering. In *Proc. IEEE Symp. on Security and Privacy*, pages 194–201, 1991.

[FDR92] Jean-Charles Fabre, Yves Deswarte, and Brian Randall. Designing secure and reliable applications using fragmentation-redundancy-scattering: an object-oriented approach. In *PDCS 2*, 1992.

[FM03] Holly Frost and Aaron Martz. The storage performance dilemma. `http://www.texmemsys.com/files/f000160.pdf`, 2003.

[HP03] John L. Hennessy and David A. Patterson. *Computer Organization and Design*. Morgan Kaufman Publishers, 2003.

[KP00] Stefan Katzenbeisser and Fabien A. P. Petitcolas, editors. *Information Hiding Techniques for Steganography and Digital Watermarking*. Artech House Books, 2000.

[LAV01] Subramanian Lakshmanan, Mustaque Ahamad, and H. Venkateswaran. A secure and highly available distributed store for meeting diverse data storage needs. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN'01)*, 2001.

[Mau92] Ueli M. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. 1992.

[MR98] Dahlia Malkhi and Michael Reiter. Byzantine quorum systems. *Distributed Computing*, 11:203–213, 1998.

[Pat94] N.P. Patt. The I/O subsystem: A candidate for improvement. *IEEE Computer: Special Issue*, 24, 1994.

[PM99] Niels Provos and David Mazieres. A future-adaptable password scheme. `http://www.openbsd.org/papers/bcrypt-paper.ps`, 1999.

[Rab89] Michael O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *Journal of the ACM*, 36, April 1989.

[Sha70] A. Shamir. How to share a secret. *Comm. of ACM*, 13(7):422–426, 1970.

[WBS$^+$00] Jay Wylie, Michael Bigrigg, John Strunk, Gregory Ganger, Han Kiliccote, and Pradeep KhoslaComputer. Survivable information storage systems. In *IEEE Computer*, volume 33, pages 61–68, August 2000.

# Countering Identity Theft Through Digital Uniqueness, Location Cross-Checking, and Funneling*

Paul C. van Oorschot[1] and S. Stubblebine[2]

[1] School of Computer Science, Carleton University, Ottawa, Canada
[2] Stubblebine Research Labs, Madison, NJ, USA

**Abstract.** One of today's fastest growing crimes is identity theft – the unauthorized use and exploitation of another individual's identity-corroborating information. It is exacerbated by the availability of personal information on the Internet. Published research proposing technical solutions is sparse. In this paper, we identify some underlying problems facilitating identity theft. To address the problem of identity theft and the use of stolen or forged credentials, we propose an authentication architecture and system combining a physical location cross-check, a method for assuring uniqueness of location claims, and a centralized verification process. We suggest that this system merits consideration for practical use, and hope it serves to stimulate within the security research community, further discussion of technical solutions to the problem of identity theft.

## 1 Introduction and Motivation

Identity theft is the unauthorized use and exploitation of another individual's identity-corroborating information (e.g. name, home address, phone number, social security number, bank account numbers, etc.). Such information allows criminal activities such as fraudulently obtaining new identity credentials, credit cards or loans; opening new bank accounts in the stolen name; and taking over existing accounts. It is one of today's fastest growing crimes. In one Canadian incident reported in April 2004 [13], a single identity theft involving real estate lead to a $540,000 loss. In 2002, reportedly 3.2 million Americans suffered an identity theft which resulted in new bank accounts or loans [1]. The severity of the problem has resulted in a recent U.S. law – the "Identity Theft Penalty Enhancement Act" – boosting criminal penalties for phishing (see below) and other identity fraud ([29]; see also [26]).

* Author addresses: paulv@scs.carleton.ca, stuart@stubblebine.com

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 31–43, 2005.
© Springer-Verlag Berlin Heidelberg 2005

Despite growing media attention and numerous web sites (government-sponsored[1] and other) discussing the problem, its seriousnous continues to be under-estimated by most people other than those who have been victimized. In the research literature to date, there appear to be few effective technical solutions or practical proposals (see below and in §2), none of which to our knowledge have been adopted successfully to the point of decreasing identity thefts in practice.

"Activity profiling" by credit card companies – a form of anomaly detection in customer usage of a credit card – partially addresses the problem of stolen or fraudulent credit cards, but not that of identity theft itself. While consumers have limited liability on use of fraudulent credit cards in their name, protection by credit card companies is limited to the realm of credit cards (see next paragraph). Regarding protection afforded by banks, in the U.S. (but reportedly not Canada), when one major bank puts an alert on a name, a common clearinghouse (limited to banks) allows all major banks to share that warning [17].

Unfortunately, identity theft appears to be a system-level problem that no one really "owns", and thus it is unclear whose responsibility it is to solve. Sadly, individual citizens are poorly positioned to solve this problem on their own, despite being the victims suffering the most in terms of disrupted lives, frustration and lost time to undo the damage – especially when stolen identity information is used to mint new forms of identity-corroborating information (or e.g. new credit cards) unbeknownst to the legitimate name-owner. According to one 2003 report [1], victims averaged 60 hours "to resolve the problem" of an identity theft, e.g. getting government and commercial organizations to stop recognizing stolen identification information, and to re-issue new identity information.

Among those perhaps in the best position to address identity theft are the national consumer credit reporting agencies – e.g. in the U.S., Equifax, Experian, and Trans Union. Among other things, the credit bureaus can when necessary post alerts on credit files of individuals whom they suspect are subjects of identity theft [17]. However, it is unclear how strongly the business models of credit bureaus motivate them to aggressively address the problem, and surprisingly some have reportedly opposed certain measures which aid in identity theft prevention (e.g. see [1]). Moreover, at least one such organization[2] was itself exploited by criminals in an incident raising fears of large-scale identity theft.

*Phishing*[3] is a relatively new Internet-based attack used to carry out identity theft. "Phishing kits" now available on the Internet allow even amateurs to create bogus websites and use spamming software to defraud users [32]. A typical phishing attack involves email sent to a list of target victims, encouraging users to visit a major online banking site. By chance a fraction of targeted users actually hold an account at the legitimate site. However the advertised link is to a spoofed site, which prompts users to enter a userid and password. Many legitimate users

[1] For example, see http://www.consumer.gov/idtheft/

[2] Equifax Canada recently confirmed that in February 2004, 1400 consumer credit reports were "accessed by criminals posing as legitimate credit grantors" [16, 17].

[3] See http://www.ftc.gov/bcp/conline/pubs/alerts/phishingalrt.htm

do so immediately, thereby falling victim. This is a variation of an attack long-known to computer scientists, whereby malicious software planted on a user's machine puts up a fraudulent login interface to obtain the user's userid and login password to an account or application.

*Key logging* attacks now rival phishing attacks as a serious concern related to online identity and sensitive personal information theft [19]. A recent example involved a trojan program *Bankhook.A* which spread without human interaction beyond web browsing, involved a (non-graphic) file named *img1big.gif*, and exploited a vulnerability in a very widely used web browser. Upon detecting attempted connections to any of about 50 major online banks,[4] it recorded sensitive information (e.g. account userid and password) prior to SSL encryption, and mailed that data to a remote computer [28, 22].

**Our Contributions.** We identify underlying problems facilitating identity theft, and propose a general authentication architecture and system we believe will significantly reduce identity theft in practice. The system combines a physical location cross-check, a method for assuring uniqueness of location claims, and a centralized verification process. We outline how the system prevents a number of potential attacks. We propose an extension addressing the problem of acquiring fraudulent new identity credentials from stolen credentials. A major objective is to stimulate further research and discussion of technical solutions to the "whole" problem of identity theft (rather than subsets thereof – e.g. phishing and key-logging).

**Organization.** The sequel is organized as follows. §2 discusses further background and related work. §3 presents an overview of our proposed authentication system and architecture for addressing identity theft, a security analysis considering some potential attacks, and a discussion of preventing privacy loss due to location-tracking. §4 gives concluding remarks.

## 2 Fundamentals and Related Work

We first discuss credentials, then identify what we see as the fundamental issues facilitating identity theft, thereafter mention a relationship to issues arising in PKI systems, and finally review related work.

**Credentials.** We define *identity credentials* (*credentials*) rather loosely as "things" generally accepted by verifiers to corroborate another individual's identity. By this definition, a credential may be digital (such as userid-password, or public-key certificate and matching private key) or physical (e.g. physical driver's license, plastic credit card, hardware token including secret key). The looseness arises from situations like the following: the secret key within a hardware token

[4] Text string searches were made for https connection attempts to URLs containing any of 50 target substrings. See Handler's log (June 29, 2004) at http://isc.sans.org/presentations/banking_malware.pdf.

is extracted, and as the key itself is then digital, essentially the important component of the physical token in now available in digital form – which we also call *credential information.* A further looseness is that unfortunately some pieces of information, such as (U.S.) Social Security Number, are used by some parties as identity-corroborating data, even if provided verbally (rather than physical inspection of a paper or plastic card) – even though they are not generally treated as secret.

**Fundamental Underlying Problems.** There are numerous reasons why personal identities and credential information are so easily stolen, and why this is so difficult to resolve. We believe the fundamental problems facilitating identity theft include the following.

F1: *ease of duplication:* the ease of duplicating personal data and credentials;
F2: *difficulty of detecting duplication:* the difficulty of detecting when a copy of a credential or credential information is made or exists (cf. [18]);[5] and
F3: *independence of new credentials:* if existing credential information is used by an impersonator to obtain new credentials, the latter are in one sense "owned" by the impersonator, and usually no information flows back to the original credential owner immediately.

In particular due to F3, we see identity theft as a *systemic* problem, which cannot be solved by any single credential-granting organization in isolation. Regarding F2, a *copy* of a cryptographic key is digital data; a copy of a physical credential is another physical object which a verifier might accept as the original.

Identity theft is also facilitated by the availability of personal information (and even full credentials, e.g. stored at servers) on the Internet; and the ease with which many merchants grant credit to new customers without proper verification of identification. While we focus on the theft of credential *information*, the theft of actual physical credentials (e.g. authentic credit cards) is also a concern – but one more easily detected.

**Relationship to PKI Systems.** We note there are similarities between detecting the theft and usage of password-based credentials and that of signature private keys; indeed, passwords and signature private keys are both secrets, and ideally in both cases, some form of theft checkpoint would exist at the time of verification. More generally, issues similar to those arising in identity theft arise in certificate validation within public key infrastructure (PKI) systems – most specifically, the revocation of private keys. There is much debate in practice and in academic research about revocation mechanisms, and which are best or even adequate. This has lead to several *online status checking* proposals (e.g. OCSP [27] and SCVP [25]), to counter latency concerns in offline models. This suggests looking to recent PKI research for ideas useful in addressing identity

[5] Thus one cannot tell when an identity theft occurs. Often copies of identity information are made, used elsewhere, and detected later only after considerable damage has occurred.

theft (and vice versa). As a related result, we cite the *CAP principle* [8, 10]: a large-scale distributed system can essentially have at most two of the following three properties: high service availability; strong data consistency; and tolerance of network partitions.

**Related Work.** The U.S. Federal Communications Commission (FCC) requires[6] that by Dec. 31 2005, wireless carriers report precise location information (e.g. within 100 meters) of wireless emergency 911 callers, allowing automatic display of address information on 911 call center phones, as presently occurs for wireline phones. Companies must either use GPS in 95% of their cell phones by Dec. 31 2005, or deploy other location-tracking technology (e.g. triangulation or location determination based on distance and direction from base stations); thereafter emergency call centers must deploy related technology to physically locate callers. As of Feb. 2004, 18% of U.S. call centers have this technology [30].

While many technologies and systems exist for determining the physical location of objects, these generally are not designed to operate in a malicious environment – e.g. see the survey by Hightower and Borriello [14]. Sastry et al. [31] propose a solution to the *in-region verification problem* of a verifier checking that a claimant is within the claimed specified region. This differs from the more difficult *secure location determination problem* involving a verifier determining the physical location of a claimant. Gabber and Wool [9] discuss four schemes, all based on available infrastructure, for detecting the movement of user equipment; they include discussion of attacks on these systems, and note that successful cloning, if carried out, would defeat all four. All of the above references address a problem other than identity theft per se, where complicating matters include the minting of new credentials (see F3 above) and uniqueness of a claimant with the claimed identity; the binding of location information to a claimed identity is also critical.

Physical location has long been proposed as a fourth basis on which to build authentication mechanisms, beyond the standard "something you know, something you have, something you are". In 1996, Denning and MacDoran [6] outlined a commercial location-based authentication system using the Global Positioning System (GPS), notwithstanding standard GPS signals being subject to spoofing (e.g. see [9, 33]. Their system did not seek to address the identity theft problem – for example regarding F2, note that in general, location information alone does not guarantee uniqueness (e.g. a cloned object may claim a different physical location than the original object); F3 is also not addressed.

One real-world system-level technique to ameliorate identity-theft is the *credit-check freeze* solution [1],[7] now available in many U.S. states. An individual can place a "fraud alert" on their credit reports, thereby blocking access to it by others for a fixed period of time, or until the individual contacts the credit bureaus and provides previously agreed information (e.g. a PIN). Another option is selective access, whereby a frozen report can be accessed only by a specifically

[6] See http://www.fcc.gov/911/enhanced/ (see also [9]).

[7] See also http://www.ftc.gov/bcp/conline/pubs/general/idtheftfact.htm

named inquirer. These methods apparently prevent identity thieves from getting (new) credit in a victim's name, or opening new accounts thereunder, but again do not solve the problem of identity theft (e.g. recall F3 above).

Corner and Noble [3] propose a mechanism involving a cryptographic token which communicates over a short-range wireless link, providing access control (e.g. authentication or decryption capabilities) to a local computing device without user interaction. While not proposed as a solution to identity theft per se, this type of solution offers an innovative alternative to easily replicated digital authentication credentials – simultaneously increasing security and decreasing user interaction (e.g. vs. standard password login).

Chou et al. [2] proposed a client-side software plug-in and various heuristics for detecting online phishing scams. Lu and Ali [24] discuss using network smart cards to encrypt sensitive data for remote nodes prior to its availability to local key-logging software.

## 3 Authentication Based on Uniqueness, Location and Funneling

A high-level overview of our proposed authentication system is given in §3.1 . A partial security analysis is given in §3.2. Privacy refinements are discussed in §3.3.

### 3.1 High-Level Overview of New System

Our goal is a system which prevents, or significantly reduces, occurrences of identity theft in practice. Our design is as follows. Every system user has a hardware-based *personal device*,[8] e.g. cell phone or wireless personal digital assistant (PDA), kept on or near their person, and which can be used to securely detect their location[9] and securely map the person to a location, ideally on a continuous basis. We call this a *heartbeat locator*, perhaps initially simply based on existing infrastructure such as emergency wireless 911 technology (see §2).

Note that in many cases, if someone has your identification credentials, or a reasonable copy thereof, for all intents and purposes they *are* you from the viewpoint of a verifier. We therefore must address both credential theft and cloning. To address cloning, one general solution is to perform a check (providing reasonably high confidence) that the personal device does in fact remain unique; we call this an *entity uniqueness* mechanism. To aid in this, we require that all identity verifications be *funneled* through a centralized point, allowing a check

[8] Here "personal" implies that the device be able to identify (or can be associated with) a unique individual.

[9] By *securely detecting location* we mean: the detected location cannot easily be spoofed. In particular, if person $P_A$ is factually at location $L_A$, then it must be very difficult (ideally infeasible in practice) for an attacker to arrange that a signal is sent indicating that $P_A$ is at a different location $L_B \neq L_A$.

to be made that no "irregularities" have occurred (based on ongoing device monitoring) for the personal device in question. For discussion of irregularities and more about theft and cloning, see §3.2.

In the process of a transaction being executed/processed, when an identity[10] is simply asserted (or ideally, confirmed by a first means), a secondary confirmation occurs based on the location of the transaction (e.g. merchant's point of sale location) matching the location the central service last recorded for the personal device corresponding to the asserted identity. This can thus be employed as a second-factor authentication system,[11] with the features of (1) combining location determination with continuous location tracking; and (2) funeling all transactions through a single point. This effectively turns an offline or distributed verification system into an online one (cf. §2).

**Extension Addressing Minting of New Credentials.** We now present a proposal to address issue F3 above (note that *some* such proposal is necessary to fully address identity theft). An extension of the above system is to require that a name-owner give explicit approval before certain actions specifically based on existing identity information – such as the minting of new credential information *not tied to the personal device* – are taken. In practice, a solution might be most effectively put in place by the national credit bureaus as a new service offering, to complement that of freezing access to credit records (see §2). Incoming queries regarding a consumer credit file could be required, by policy, to specify if the inquiry was being used to mint credentials which might reasonably be used as identity credentials by other responsible parties. The major credit bureaus might provide (in a coordinated manner) a central alert-centre to check if such credential minting was currently "allowed" by the legitimate name-owner (e.g. as indicated by a *minting bit* in the existing credit file). Reputable (participating) organizations which created any form of personal credential would agree[12] to create new credentials only if the response from the centralized service indicated the minting bit was on. In this way, a cautious individual, even without prior identity theft problems, could have minting of new credentials disabled the majority of the time, as a pre-emptive measure.

### 3.2 Security Analysis and Discussion

In this section we provide a partial security analysis of the new proposal, and discuss necessary checks regarding the personal device. While we offer no rigorous

[10] An identity per se is not required – e.g. pseudonyms could be used, to enhance privacy (see §3.3).

[11] Again, this is a systemic (multi-application) authentication system addressing identity theft, rather than a second-factor point solution limited to a particular application, such as credit card authorization.

[12] We recognize that this would require a significant change in behaviour by many organizations, over a long period of time (which legislation might shorten). However, we expect that nothing less will solve the difficult problem of identity theft.

security arguments here,[13] we discuss a number of attack scenarios and how the system addresses these. We do not "prove" that the proposed system is "secure" in a general practical setting, and believe this would be quite difficult, as "proofs" of security are at best relative to a particular model and assumptions, with increased confidence in the relevance and suitability of these generally gained only over time. However we encourage further analysis to allow the proposal to be iteratively improved.

We begin by referring back to the three fundamental problems of §2. The system proposed in §3.1 addresses these as follows. The ease of credential duplication (F1) is reduced by the use of a hardware device; the capability to detect credential duplication (F2) is provided by the funneling mechanism and ongoing device monitoring (heartbeat mechanism); and the minting of new (fraudulent) credentials based on stolen authentic credentials (F3) is partially[14] addressed by the "minting bit" extension.

**Device Irregularities, Theft and Cloning.** Fraud mitigation strategies depend on users reporting stolen personal devices in a timely matter.[15] However, some heuristics may also be effective to detect both theft and cloning. Examples of heuristic predictors of cloning include the same personal device appearing multiple times (two heartbeats asserting the same identity, whether at the same or distinct locations), or in two different locations within an unreasonably short period of time (taking into account usual modes of travel). A heuristic indicator of device theft is a user unable to correctly authenticate even though the location is verifiable (e.g. within range). These are all examples of *irregularities*. In this case, authentication attempts using the device within a short time thereafter may be suspect.

Personal devices flagged as having experienced sufficient irregularities should be disallowed from participating in transactions, or subject to additional checks. As suspicion arises regarding a device (cloning, theft or other misuse), extensions to the basic techniques are possible. For example, the personal device holder might be requested to provide an additional authentication factor to confirm a transaction. In essence, known techniques used for credit card activity profiling, which by system design are currently used only to mitigate credit card fraud, could be adapted to mitigate identity theft in the new system.

Note that a theft deterrent in this system is the risk of physical discovery – device possession allows location-tracking of the thief. Related to this, the deactivation (if featured) and re-activiation of the device's location-tracking feature should also require some means of user authentication, so that a thief cannot

[13] A more complete security analysis will be given in the full paper.

[14] Our proposal does not prevent an attacker from himself forging new credentials; but can prevent the use of stolen credentials to obtain new credentials from an authentic credential-generating organization.

[15] Loaning a personal device to a non-malicious user (e.g. a relative or friend) does not necessarily cause an increase in fraud since those users generally are trusted not to commit fraud using the device.

disable this feature easily, and if already disabled, the device is unusable for authentication.

**Device Uniqueness.** While ideally the personal device would be difficult to physically duplicate, our proposal only partially relies on this, as duplicate heartbeats will lead to a failed verification check. To enforce device uniqueness, ideally both (1) each device is tracked continuously since registration; and (2) it can be verified that the user originally registering a device remains associated with the tracked device. We may consider the latter issue under the category of theft, and the former under cloning. In practice, monitoring could at best be roughly continuous, e.g. within discrete windows of time, say from sub-second to a minute; we expect this would not pose a significant problem. However there are practical contraints in even roughly monitoring devices – for example, wireless devices are sometimes out of range (e.g. in tunnels, or on airplanes) or turned off. Thus the system must address the situation in which for at least some devices, location-tracking is temporarily disabled. It may be an acceptable risk to allow a device to be "off-air" for a short period of time (e.g. seconds or minutes), provided that it reappears in a reasonably plausible geographic location. Devices "off-air" for a longer period could be required to be re-activated by a user-to-system authentication means (i.e. not user-to-device). Personal devices which have gone "off-air" recently might be given a higher irregularity score, or not be allowed to participate in higher-value transactions (absent additional assurance) for some period of time.

**Threats and Potential Attacks.** The class of threats we are intending to protect against is essentially the practical world, or more precisely, any plausible real-world attack of "reasonable" cost (relative to the financial gain of the identity theft to the attacker). We consider here a number of potential attacks, and discuss how the system fares against them.

1. *Theft.* If the personal device is stolen or lost, the loss should be reported leading to all further verification checks failing; effectively this is credential revocation. Since often a theft is not immediately noticed or reported, the device should require some explicit user authentication mechanism (such as a user-entered PIN or biometric) as part of any transaction; the device should be shut down upon a small number of incorrect entries (possibly allowing a longer "unblocking PIN" for re-activation).[16]
2. *Cloning.* There can be no absolute certainty that the personal device has not been cloned or mimicked. If a clone exists, either it has a continuous heartbeat (case A), or no heartbeat (case B). In case A, assuming the original device also still has a heartbeat, the system will be receiving two heartbeats with the same device identifier, and flag an irregularity. In case B, if and when

[16] Although a motivated and well-armed attacker can generally defeat user-to-device authentication mechanisms (cf. [9]), we aim to significantly reduce, rather than totally eliminate, occurrences of identity theft. We believe a 100% solution will be not only too expensive or user-unfriendly, but also non-existent.

the cloned device is used for a transaction, its location will be inconsistent with previous heartbeats (from the legitimate device), and thus the cloned device will be unable to successfully participate in transactions.

3. *Theft, clone, return.* Another potential attack is for a thief to steal a device, clone it (in a tracking de-activated state), then "simultaneously" activate the clone and deactivate the original, and finally return the stolen device. The idea is then to carry out a transaction before the original device owner reactivates or reports the theft. Such an attack, if possible, would nonetheless make identity thefts significantly more difficult than today (and thus our goal would be achieved). A variation has the attacker inject unauthorized software in the original device, to completely control it (including the capability to remotely power it on and off), before returning it. Then at the instance of carrying out a transaction, the attacker remotely powers down the original before powering up the clone, to prevent detection of two heartbeats. However a geographic irregularity would arise (as the clone's location would differ from that of the last heartbeat of the real device).
4. *Same-location attack.* An attacker, without possessing a target victim's personal device, might attempt to carry out a transaction at the same physical location (e.g. retail store) as the target victim and that victim's personal device. This attack should be prevented by a requirement that a user take some physical action to commit a transaction (e.g. press a designated key, enter a PIN, or respond to an SMS message). A further refinement is an attacker attempting to carry out a transaction at the same place and the same instant as a legitimate user (and also possessing any other credentials necessary to impersonate the user in the transaction). Here the attacker would be at some physical risk of discovery, and one of the two transactions would go through. While this attack requires further consideration, it appears to be less feasible.

### 3.3 Privacy Enhancement

The proposal of §3.1 is a starting point towards a technical system-level approach to addressing identity theft. We acknowledge that it leaves many opportunities for enhancement, and contains some features which some may find unacceptable. Among these is the loss of privacy as a result of continual location-tracking. While there is always a price to pay for increased security, for some users this loss of privacy will clearly be above the acceptable threshold. Thus it is important to explore means to address this privacy issue (cf. [9, 23]).

A user can choose a *trusted third party* (TTP) he is willing to trust to maintain the privacy of his information. In many ways the user is already trusting the communication provider of his personal device (e.g. cell phone, and wireless internet) concerning the privacy of his location information.[17] More generally,

[17] As a side comment, many people enjoy far less privacy than perhaps presumed, due to existing location-tracking technology such as wireless 911 services (see §2). However, this may not bring much comfort.

while each user could be associated with a particular TTP for location tracking, a relatively large set of TTPs in the overall system could aid scalability and eliminate system-wide single points of failure.

The "Wireless Privacy Protection Act of 2003" [15] requires customer consent related to the collection and use of wireless call location information, and call transaction information. Further it requires that "the carrier has established and maintains reasonable procedures to protect the confidentiality, security, and integrity of the information the carrier collects and maintains in accordance with such customer consents." This or other legislation could mean that straight-forward approaches are practical if organizations can be trusted to adequately protect location data. However, it may be argued that many information-receiving organizations might not be able or trustworthy to guarantee protection of location information and personal transaction data.

As the idea of relying on regulation and the trustworthiness of information holders to protect location and other personal information may cause discomfort to those with strong privacy concerns, we encourage further research on using privacy-preserving techniques to achieve digital uniqueness with a trusted (or minimally trusted) third party. To this end, there exists extensive literature following on from Chaum's early work [4] on digital pseudonyms and mix networks, for protecting privacy including the identities involved in, and the source/destination of communications. Privacy-related applications of such techniques include e-elections (e.g. [21]), anonymous email delivery (e.g. [5]), and of particular relevance, location management in mobile communications [7]. (For further recent references, see e.g. [11].) While we do not foresee serious technical roadblocks to integrating largely existing privacy-enhancing technologies to significantly improve the privacy aspects of this proposal, further pursuit of this important topic is beyond the scope of this paper.

## 4 Concluding Remarks

We have proposed an architecture and system for authentication involving a physical location cross-check, and reliance on an entity uniqueness property and funneling within the verification process. While the system is relatively simple – essentially a selective combination of existing technology and techniques – we believe it may be successful at stopping many forms of identity theft. This appears to be among the first technical proposals to address identity theft in a research paper. In our view, part of the problem is that it is not clear which research community is a natural "owner" of the problem. Although in many ways more of a system-engineering than a traditional security problem, we believe that increasingly, technical solutions to identity theft will fall to the security research community. Indeed, phishing for passwords and installation of key-logging software/hardware, which both facilitate identity theft, are problems whose solutions one would naturally seek from the security research community.

It should be clear that we have not yet built the proposed system, even in a test environment, and doing so would not "prove" our proposal was secure in a practical sense. The best, and perhaps only true way to test such a system would be to observe any reduction in identity thefts in a real-world deployment. Nonetheless, we believe this paper lays out sufficient details for security-aware systems-level engineers within appropriate organizations (e.g. major credit card associations, banks, credit rating agencies, or national ID card system designers – cf. [20]) to implement such a system. Any such implementation must be designed keeping scalability in mind, particularly in light of the continuous nature of the tracking.

Effectively, our proposal is a mechanism for enforcing unique ownership of names (i.e. identities), and includes an extension addressing the minting of new (fraudulent) credentials from stolen credentials. We encourage the research community to explore alternate solutions to the latter problem, which is closely linked to that of identity theft.

**Acknowledgements.** We thank Carl Ellison for bringing references [8] and [10] to our attention. The first author acknowledges the support of the National Sciences and Engineering Research Council of Canada (NSERC) for support under both a Canada Research Chair and an NSERC Discovery Grant. The second author acknowledges that this material is based on work supported by the National Science Foundation under Grant No. 0208983.

## References

1. CNN.com, “Anti-identity theft freeze gaining momentum; Credit companies resist measure”, Aug.3 2004, http://www.cnn.com/2004/TECH/biztech/08/03/security.freeze.ap.
2. N. Chou, R. Ledesma, Y. Teraguchi, J.C. Mitchell, “Client-side defense against web-based identity-theft”, Proc. of *Network and Distributed System Security Symposium* (NDSS’04), Feb. 2004, San Diego.
3. Mark D. Corner, Brian D. Noble, “Zero-Interaction Authentication”, Proc. of *MOBICOM’02*, 23–28 Sept. 2002, Atlanta.
4. D. Chaum, “Untraceable electronic mail, return addresses, and digital pseudonyms,” *Comm. of the ACM*, 1981, pp.84–88.
5. G. Danezis, R. Dingledine, N. Mathewson, “Mixminion: Design of a Type III Anonymous Remailer Protocol”, pp.2–15, *2003 IEEE Symp. Security and Privacy.*
6. D.E. Denning, P.F. MacDoran, “Location-Based Authentication: Grounding Cyberspace for Better Security”, *Computer Fraud and Security*, Feb. 1996.
7. H. Federrath, A. Pfitzmann, A. Jerichow, “MIXes in Mobile Communication Systems: Location Management with Privacy”, *Workshop on Information Hiding*, Cambridge U.K., 1996.
8. A. Fox, E. Brewer, “Harvest, Yield and Scalable Tolerant Systems”, Proc. of *HotOS-VII*, 1999.
9. E. Gabber, A. Wool, “On Location-Restricted Services”, *IEEE Network*, November/December 1999.
10. Seth Gilbert, Nancy Lynch, “Brewer’s conjecture and the feasability of consistent, available, partition-tolerant web services”, *Sigact News* 33(2), June 2002.

11. P. Golle, A. Juels, “Parallel Mixing”, pp.220-226, *2004 ACM Conf. Computer and Comm. Security.*
12. Carl A. Gunter, Michael J. May, Stuart G. Stubblebine, “A formal privacy system and its application to location based services”, in: *Workshop on Privacy Enhancing Technologies 2004.*
13. Darcy Henton, “Identity-theft case costs taxpayers in Alberta $540,400”, 12 April 2004, *The Globe and Mail*, Toronto.
14. J. Hightower, G. Borriello, “Location Systems for Ubiquitous Computing”, *IEEE Computer*, Aug. 2001.
15. Wireless Privacy Protection Act of 2003, 108th Cong, H.R. 71 (United States).
16. Mark Hume, “Security breach lets criminals view Canadians’ credit reports”, 16 March 2004 (page A1/A7), *The Globe and Mail*, Toronto.
17. Mark Hume, “Identity theft cited as threat after Equifax security breach”, 17 March 2004 (page A7), *The Globe and Mail*, Toronto.
18. M. Just, P.C. van Oorschot, “Addressing the problem of undetected signature key compromise”, Proc. of *Network and Distributed System Security Symp.* (NDSS’99), Feb. 1999, San Diego.
19. G. Keizer, “Internet Scams Cost Consumers $2.4 Billion”, TechWeb News, InternetWeek, 16 June 2004.
20. S.T. Kent, L. Millette, eds., *IDs – Not That Easy: Questions About Nationwide Identity Systems*, National Academies Press (U.S.), 2002.
21. A. Kiayias, M. Yung, “The Vector-Ballot e-Voting Approach”, pp.72–89, *Financial Cryptography’04.*
22. Robert Lemos, “Pop-up program reads keystrokes, steals passwords”, CNET News.com, 29 June 2004, http://news.com.com/2100-7349-5251981.html.
23. P. Lincoln, P. Porras, V. Shmatikov, “Privacy-Preserving Sharing and Correlation of Security Alerts”, in: Proc. of *13th USENIX Security Symposium*, August 2004, San Diego.
24. Karen Lu, Asad Ali, “Prevent Online Identity Theft – Using Network Smart Cards for Secure Online Transactions”, *2004 Information Security Conference* (ISC’04), Sept. 2004, Palo Alto.
25. A. Malpani, R. Housely, T. Freeman, Simple Certificate Validation Protocol (SCVP), Internet Draft (work in progress), draft-ietf-pkix-scvp-15.txt, July 2004.
26. Declan McCullagh, “Season over for ‘phishing’?”, CNET News.com, 15 July 2004, http://news.com.com/2100-1028-5270077.html.
27. M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, X.509 Internet Public Key Infrastructure: Online Certificate Status Protocol – OCSP, Internet Request for Comments 2560, June 1999.
28. Panda Software, Bankhook.A (Virus Encyclopedia entry), http://www.pandasoftware.com.
29. Public Law No. 108-275, “Identity Theft Penalty Enhancement Act”, United States, July 2004.
30. Jonathan D. Salant, “Call centers lag in cell-phone tracking upgrade, group says”, 6 February 2004, (page A8), *The San Diego Union Tribune.*
31. N. Sastry, U. Shankar, D. Wagner, “Security verification of location claims”, in: *2003 ACM Workshop on Wireless Security (WiSe 2003).*
32. James Sherwood, “So you want to be a cybercrook...”, CNET News.com (ZDNET UK), Aug.29 2004, http://zdnet.com.com/2100-1105-5317087.html.
33. John A. Volpe National Transportation Systems Centre, Vulnerability Assessment of the Transportation Infrastructure Relying on the Global Positioning System, Final Report, 29 August 2001.

# Trust and Swindling on the Internet

Bezalel Gavish

Southern Methodist University,
Dallas, TX 75205
gavishb2000@yahoo.com

Fraud on the Internet is developing into a major issue of concern for consumers and businesses. Media outlets report that online fraud represents "an epidemic of huge and rapidly growing proportions”. One area that is particularly of interest is the area of swindling activities related to online auctions. Understanding fraud is especially important because of the "network externality” effect, in which a large number of satisfied users attracts other users to use the commercial services offered through the Internet, this effect is based on the knowledge that satisfied traders induce others to trade on the Internet increasing the trading system efficiency. Headlines that present swindling activities on the internet deter users from using the internet for commercial activities.

We will present and classify methods that swindlers use in order to defraud users, and suggest procedures to reduce the level of successful fraudulent activities on the web. We will also report on a preliminary empirical survey on the magnitude of fraudulent auctions on a large auction site. The empirical results obtained in this survey, invalidate claims by online auction site operators that fraudulent activity is negligible. We will also discuss methods to reduce fraud and the need for extensive research on Internet fraud.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, p. 44, 2005.
© Springer-Verlag Berlin Heidelberg 2005

# Identity-Based Partial Message Recovery Signatures (or How to Shorten ID-Based Signatures)*

Fangguo Zhang[1,**], Willy Susilo[2], and Yi Mu[2]

[1] Department of Electronics and Communication Engineering,
Sun Yat-sen University, Guangzhou 510275, P.R. China
isdzhfg@zsu.edu.cn

[2] School of Information Technology and Computer Science,
University of Wollongong, Australia
{wsusilo, ymu}@uow.edu.au

**Abstract.** We propose a new notion of short identity-based signature scheme. We argue that the identity-based environment is essential in some scenarios. The notion of short identity-based signature schemes can be viewed as identity-based (partial) message recovery signatures. Signature schemes with message recovery has been extensively studied in the literature. This problem is somewhat related to the problem of signing short messages using a scheme that minimizes the total length of the original message and the appended signature. In this paper, firstly, we revisit this notion and propose an identity-based message recovery signature scheme. Our scheme can be regarded as the identity based version of Abe-Okamoto's scheme [1]. Then, we extend our scheme to achieve an identity-based partial message recovery signature scheme. In this scheme, the signature is appended to a truncated message and the discarded bytes are recovered by the verification algorithm. This is to answer the limitation of signature schemes with message recovery that usually deal with messages of fixed length. This paper opens a new research area, namely how to shorten identity based signatures, in contrast to proposing a short signature scheme. We present this novel notion together with two concrete schemes based on bilinear pairings.

## 1 Introduction

Even in a small organization, it is desirable to authenticate all messages sent from one employee to the other. One way to authenticate an email is by incorporating a method such as PGP. However, the length of the signature itself is quite long. This drawback certainly has great influence in an organization where bandwidth is one of the main concerns. Therefore, the invention of a short signature scheme

[*] This work is supported by ARC Discovery Grant DP0557493.

[**] This work is supported by the National Natural Science Foundation of China (No. 60403007).

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 45–56, 2005.
© Springer-Verlag Berlin Heidelberg 2005

applicable to email is essential. This problem can be viewed as how to construct an identity based (or ID-based, for short) short signature scheme. The ID-based scenario is required to avoid the necessity to employ a certification system.

Several signature schemes have been proposed over the last decade by the research community. It is known that a signature scheme that produces signatures of length $\ell$ can have some security level of at most $2^\ell$, which means that given the public key, it is possible to forge a signature on any message in $\mathcal{O}(2^\ell)$ time. A natural question is how we can concretely construct a signature scheme that can produce shorter signature length whilst maintaining the same security level against existential forgery.

It was noted in [7] that in some situations, it is desirable to use very short signatures, for instance when one needs to sign a postcard. In this situation, it is desirable to minimize the total length of the original message and the appended signature. In the early days, research in this area has been mainly focusing on how to minimize the total length of the message and the appended signature [7, 1]. The idea was originated from the message recovery schemes, for example [8]. For example, the work proposed in [7] has shortened DSS signatures to provide security level $\mathcal{O}(2^\ell)$ with signature length of about $3.5\ell$ bits (in contrast to $4\ell$ bits in the original DSS scheme).

A totally new approach to this problem was made by Boneh, Lynn and Shacham by proposing a short digital signature scheme, where signatures are about half the size of DSA signatures with the same level of security [5]. The resulting signature scheme, referred to as the BLS signature scheme, is based on the Computational Diffie-Hellman (CDH) assumption on certain elliptic curves. The approach that was taken in this scheme is totally different from its predecessor, i.e. directly minimizing the signature without providing a partial message to the receiver, with the intention that on the receiver's side, the complete message can be recovered (eg. [7, 1]). In BLS signature scheme, with a signature length $\ell = 160$ bits (which is approximately half the size of DSS signatures with the same security level), it provides a security level of approximately $\mathcal{O}(2^{80})$ in the random oracle model. This signature scheme has attracted a lot of attention in the research community and has been used to construct several other new schemes (eg. [4, 12]). The main drawback of the BLS scheme is its dependency on a special hash function, i.e. an admissible encoding function, which is still probabilistic.

In [13], a more efficient approach to produce a signature of the same length of BLS was proposed by Zhang, Safavi-Naini and Susilo. However, its security is based on a stronger assumption. The same assumption has been used by Boneh and Boyen [2] to produce a short signature scheme *without* random oracles.

### 1.1 Our Contribution

In this paper, we revisit the notion of shortening messages and the appended signature [7] in the ID-based setting. We provide two formal models and schemes, namely an ID-based message recovery signature scheme and an ID-based partial message recovery signature scheme. Our ID-based message recovery signature scheme can be regarded as an ID-based version of the signature scheme in [1]. Although message recovery techniques seem to solve the signature size problem, they still suffer from several drawbacks. They usually deal with messages of fixed length and it is unclear how to extend them when the message exceeds some given size. For example, the Nyberg-Rueppel scheme applied to redundant messages of twenty bytes. This presumably means ten bytes for the message and ten for the redundancy, but the message could be of fourteen bytes long. In our ID-based partial message recovery signature scheme, we resolve this problem by providing a method to deal with the messages of arbitrary length.

The rest of this paper is organized as follows. In section 2, we review some preliminaries used throughout this paper. In section 3, we propose a notion of ID-based message recovery signature scheme and present a concrete scheme based on bilinear pairings. In section 4, we extend this notion to an ID-based partial message recovery signature scheme that can handle messages of arbitrary length. Finally, section 5 concludes the paper.

## 2 Preliminaries

### 2.1 Bilinear Pairings

Let $\mathbb{G}_1, \mathbb{G}_1'$ be cyclic additive groups generated by $P_1, P_1'$, respectively, whose order are a prime $q$. Let $\mathbb{G}_2$ be a cyclic multiplicative group with the same order $q$. We assume there is an isomorphism $\psi : \mathbb{G}_1' \to \mathbb{G}_1$ such that $\psi(P_1') = P_1$. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1' \to \mathbb{G}_2$ be a bilinear mapping with the following properties:

1. *Bilinearity*: $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P \in \mathbb{G}_1, Q \in \mathbb{G}_1', a, b, \in \mathbb{Z}_q$.
2. *Non-degeneracy*: There exist $P \in \mathbb{G}_1, Q \in \mathbb{G}_1'$ such that $\hat{e}(P, Q) \neq 1$.
3. *Computability*: There exists an efficient algorithm to compute $\hat{e}(P, Q)$ for all $P \in \mathbb{G}_1, Q \in \mathbb{G}_1'$.

For simplicity, hereafter, we set $\mathbb{G}_1 = \mathbb{G}_1'$ and $P_1 = P_1'$. We note that our scheme can be easily modified for a general case, when $\mathbb{G}_1 \neq \mathbb{G}_1'$.

Bilinear pairing instance generator is defined as a probabilistic polynomial time algorithm $\mathcal{IG}$ that takes as input a security parameter $\ell$ and returns a uniformly random tuple $param = (p, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, P)$ of bilinear parameters, including a prime number $p$ of size $\ell$, a cyclic additive group $\mathbb{G}_1$ of order $q$, a multiplicative group $\mathbb{G}_2$ of order $q$, a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ and a generator $P$ of $\mathbb{G}_1$. For a group $\mathbb{G}$ of prime order, we denote the set $\mathbb{G}^* = \mathbb{G} \setminus \{\mathcal{O}\}$ where $\mathcal{O}$ is the identity element of the group.

**Complexity Assumption**

**Definition 1. (Computational Diffie-Hellman (CDH) Problem)** *Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of order the same prime order $q$. Let $P$ be a generator of $\mathbb{G}_1$. Suppose there exists a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$. Let $\mathcal{A}$ be an attacker. $\mathcal{A}$ tries to solve the following problem:* Given $(P, aP, bP)$ for some unknown $a, b \in \mathbb{Z}_q^*$, compute $abP$.

The success probability of $\mathcal{A}$, which is polynomially bounded with a security parameter $\ell$, is defined as

$$\mathtt{Succ}^{CDH}_{\mathbb{G}_1,\mathcal{A}}(\ell) = Pr[\mathcal{A}(P, aP, bP, abP) = 1; a, b \in \mathbb{Z}_q^*].$$

The CDH problem is said to be intractable, if for every probabilistic polynomial time, 0/1-valued algorithm $\mathcal{A}$, $\mathtt{Succ}^{CDH}_{\mathbb{G}_1,\mathcal{A}}(\ell)$ is negligible.

### 2.2 Identity-Based Cryptography

The idea of ID-based system was originally proposed by Shamir [11]. In this system, the user's public key is simply an identifier information (identity) of the user. In other words, the user's public key can be calculated directly from his/her identity rather than being extracted from a certificate issued by a certificate authority (CA). ID-based public key setting can be a good alternative for certificate-based public key setting, especially when efficient key management and moderate security are required. The construction of identity based signature scheme was also proposed in [11], but the first efficient construction of ID-based encryption scheme was proposed in [3] that was developed using bilinear pairings.

### 2.3 Notations

Throughout this paper, we will use the following notations. Let $|q|$ denote the length of $q$ in bits. Let $[m]^{k_1}$ denote the most significant $k_1$ bits of $m$ and $[m]_{k_2}$ denote the least significant $k_2$ bits of $m$.

## 3 Identity-Based Message Recovery Signatures

### 3.1 Model

There exists a trusted Private Key Generator ($PKG$) in the system. An ID-based message recovery signature scheme consists of four algorithms.

- $\mathsf{Setup}$: A probabilistic algorithm that is on input a $PKG$'s secret key, $s_{PKG}$, outputs the $PKG$'s public key, $P_{pub}$, together with the system parameters, $\mathtt{param}$.
- $\mathsf{Extract}$: A deterministic algorithm that is on input an identity of a user, $\mathsf{ID}$, outputs a user's secret key, $\mathcal{S}_{\mathsf{ID}}$.
- $\mathsf{Sign}$: A probabilistic algorithm that accepts a message $m$, an identity $\mathsf{ID}$ and his/her secret key $\mathcal{S}_{\mathsf{ID}}$, outputs a signature $\sigma$ on $m$.
- $\mathsf{Verify}$: A deterministic algorithm that accepts an identity of the sender, $\mathsf{ID}$ and a signature $\sigma$, outputs either $\mathtt{true}$ or $\perp$ to indicate whether the verification is successful or not. When the output is $\mathtt{true}$, the original message $m$ can be reconstructed.

*Consistency*
For consistency of the scheme, we require

$$Pr\begin{pmatrix}(\mathtt{true}, m) \leftarrow \mathsf{Verify}(\sigma, \mathsf{ID});\\ \sigma \leftarrow \mathsf{Sign}(\mathsf{ID}, \mathcal{S}_{\mathsf{ID}}, m);\\ \mathcal{S}_{\mathsf{ID}} \leftarrow \mathsf{Extract}(\mathsf{ID})\end{pmatrix} = 1$$

holds with an overwhelming probability.

### 3.2 Formal Security Notion

We provide a formal definition of existential unforgeability of an ID-based message recovery signature scheme under a chosen message attack. To do this, we extend the definition of existential unforgeability against a chosen message attack of [6]. Our extension is strong enough to capture an adversary who can simulate and observe the scheme. It is defined using the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

- Setup: $\mathcal{C}$ runs Setup for a given security parameter $\ell$ to obtain a public parameter param. The public key of the $PKG$, $P_{pub}$, is also obtained. The associated $PKG$'s secret key is kept by $\mathcal{C}$.
- Extract Queries: $\mathcal{A}$ can request the private key corresponding to any identity, $\mathsf{ID}_i$ for $1 \leq i \leq q_{ex}$ where $q_{ex}$ denotes the number of extraction queries, which is polynomial in $l$. As a response to each query, $\mathcal{C}$ runs Extract taking $\mathsf{ID}_i$ as input and returns a resulting secret key $\mathcal{S}_{\mathsf{ID}_i}$.
- Sign Queries: $\mathcal{A}$ can request a signature on a message $m_j$ for $1 \leq j \leq q_m$ where $q_m$ denotes the number of extraction queries, which is polynomial in $l$, for any identity $\mathsf{ID}_i$. In response, $\mathcal{C}$ runs Extract to get a secret key $\mathcal{S}_{\mathsf{ID}_i}$ associated with $\mathsf{ID}_i$. It then runs Sign taking $\mathsf{ID}_i$, $\mathcal{S}_{\mathsf{ID}_i}$ and $m_j$ as inputs and returns a resulting a signature $\sigma_j$ for the message $m_j$.
- Verify Queries: Answers to these queries are not provided by $\mathcal{C}$ since $\mathcal{A}$ can compute them for himself using the Verify algorithm.
- Output: Finally, $\mathcal{A}$ outputs a tuple $(\mathsf{ID}, \sigma)$. $\mathcal{A}$ wins the game if $\mathsf{Verify}(\mathsf{ID}_i, \sigma) \stackrel{?}{=}$ true holds; no secret key for ID was issued in Extract Queries stage; and $\sigma$ was not obtained in Sign Queries stage.

The success probability of an adversary to win the game is defined by

$$\mathtt{Succ}_{\mathcal{A}}^{UF-IDMRSS-CMA}(\ell).$$

**Definition 2.** *We say that an ID-based message recovery signature scheme is existentially unforgeable under a chosen message attack if the probability of success of any polynomially bounded adversary in the above game is negligible. In other words,*

$$\mathtt{Succ}_{\mathcal{A}}^{UF-IDMRSS-CMA}(\ell) \leq \epsilon$$

### 3.3 A Concrete Scheme from Bilinear Pairing

In this section, we present a concrete ID-based message recovery signature scheme from bilinear pairing. Our scheme can be regarded as the ID-based version of Abe-Okamoto's scheme [1]. The scheme is illustrated as follows.

- Setup: $PKG$ chooses a random number $s \in \mathbb{Z}_q^*$ and sets $P_{pub} = sP$. $PKG$ also publishes system parameters $\{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, \lambda, P, H_0, H_1, F_1, F_2, k_1, k_2\}$, and keeps $s$ as the *master-key*, which is known only to itself. Here $|q| = k_1 + k_2$, $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$, $H_0 : \{0,1\}^* \rightarrow \mathbb{G}_1^*$, $F_1 : \{0,1\}^{k_2} \rightarrow \{0,1\}^{k_1}$ and $F_2 : \{0,1\}^{k_1} \rightarrow \{0,1\}^{k_2}$ are four cryptographic hash functions.
- Extract: A user submits his/her identity information $\mathsf{ID}$ to $PKG$. $PKG$ computes the user's public key as $\mathsf{Q}_{\mathsf{ID}} = H_0(\mathsf{ID})$, and returns $\mathcal{S}_{\mathsf{ID}} = s\mathsf{Q}_{\mathsf{ID}}$ to the user as his/her private key.
- Sign: Let the message be $m \in \{0,1\}^{k_2}$.
  S1 $v = \hat{e}(P,P)^k$, where $k \in_R \mathbb{Z}_q^*$
  S2 $f = F_1(m)||(F_2(F_1(m)) \oplus m)$
  S3 $r = H_1(v) + f \pmod q$
  S4 $U = kP - r\mathcal{S}_{\mathsf{ID}}$.
  The signature is $(r, U)$. We note the length of the signature is $|r + U| = |q| + |\mathbb{G}_1|$. This signature can be used to recover the message $m$, where $|m| = k_2$.
- Verification: Given $\mathsf{ID}$, a message $m$, and a signature $(r, U)$, compute
$$r - H_1(\hat{e}(U,P)\hat{e}(\mathsf{Q}_{\mathsf{ID}}, P_{pub})^r) = f,$$
  and
$$m = [f]_{k_2} \oplus F_2([f]^{k_1}).$$
  Check whether $[f]^{k_1} = F_1(m)$ holds. If it is correct, then accept this signature and output `true`. Otherwise, output $\perp$.

### 3.4 Security Analysis

**Theorem 1.** *Our ID-based message recovery signature scheme is correct and sound.*

*Proof.* The correctness of the scheme is justified as follows.

$$\begin{aligned}\hat{e}(U,P)\hat{e}(\mathsf{Q}_{\mathsf{ID}}, P_{pub})^r &= \hat{e}(kP - r\mathcal{S}_{\mathsf{ID}}, P)\hat{e}(\mathsf{Q}_{\mathsf{ID}}, P_{pub})^r \\ &= \hat{e}(kP - r\mathcal{S}_{\mathsf{ID}}, P)\hat{e}(s\mathsf{Q}_{\mathsf{ID}}, P)^r \\ &= \hat{e}(kP - r\mathcal{S}_{\mathsf{ID}}, P)\hat{e}(r\mathcal{S}_{\mathsf{ID}}, P) \\ &= \hat{e}(kP, P) \\ &= \hat{e}(P,P)^k\end{aligned}$$

Hence, we obtain

$$\begin{aligned}r - H_1(\hat{e}(U,P)\hat{e}(\mathsf{Q}_{\mathsf{ID}}, P_{pub})^r) &= r - H_1(\hat{e}(P,P)^k) \\ &= r - H_1(v) \\ &= f\end{aligned}$$

Since $f$ is computed from $f = F_1(m)||(F_2(F_1(m)) \oplus m)$, therefore testing whether

$$\begin{aligned}[r - H_1(\hat{e}(U, P)\hat{e}(\mathsf{Q}_{\mathsf{ID}}, P_{pub})^r)]^{k_1} &= [f]^{k_1} \\ &= F_1(m)\end{aligned}$$

should hold with equality. This way, we obtain $[f]^{k_1} = F_1(m)$. Finally, to recover the message from the signature, we can compute

$$\begin{aligned} m &= [f]_{k_2} \oplus F_2([f]^{k_1}) \\ &= [(F_1(m)||(F_2(F_1(m)) \oplus m))]_{k_2} \oplus F_2([f]^{k_1}) \\ &= F_2(F_1(m)) \oplus m \oplus F_2([f]^{k_1}) \\ &= F_2(F_1(m)) \oplus m \oplus F_2(F_1(m)) \\ &= m \end{aligned}$$

□

**Theorem 2.** *Our ID-based message recovery signature scheme is existentially unforgeable under a chosen message attack in the random oracle model, assuming the hardness of Computational Diffie-Hellman problem.*

*Proof.* We note that the complete security proof of this Theorem will be provided in the full version of this paper. □

### 3.5 Efficiency and Limitation

The length of the signature produced by our scheme is $|r + U| = |q| + |\mathbb{G}_1|$. This signature can be used to sign (and recover) the message $m$, where $|m| = k_2$. Using any of the families of curves described in [5], one can select $q$ to be a 170-bit prime and use a group $\mathbb{G}_1$ where each element is 171 bits. Hence, the total signature length is 341 bits or 43 bytes. With these parameters, security is approximately the same as a standard 1024-bit RSA signature, which is 128 bytes. This signature scheme can be used to recover a message $m$ where $|m| = k_2$ and $|q| = k_1 + k_2$. The overhead of this scheme is $|q| + |\mathbb{G}_1| - k_2 = |\mathbb{G}_1| + k_1$. To obtain a $2^{-80}$ probability of the verification condition holding for an attempted forgery generated by an adversary, we need to have $k_1 \leq 80$ bits. Hence, if $|\mathbb{G}_1|$ is chosen to be 171 bits, we obtain the signature overhead as 251 bits. We note that the previous pairing ID-based signature schemes normally requires two elements of $\mathbb{G}_1$, which is approximately 340 bits. The only limitation in this scheme is that the message size $|m|$ is limited to $k_2$. In the next section, we will eliminate this problem by proposing an ID-based partial message recovery signature scheme.

## 4 Identity-Based Partial Message Recovery Signatures

### 4.1 Model

There exists a trusted *PKG* in the system. An ID-based message recovery signature scheme consists of four algorithms.

- $\mathsf{Setup}$: A probabilistic algorithm that is on input a $PKG$'s secret key, $s_{PKG}$, outputs the $PKG$'s public key, $P_{pub}$, together with the system parameters, param.
- $\mathsf{Extract}$: A deterministic algorithm that is on input an identity of a user, $\mathsf{ID}$, outputs a user's secret key, $\mathcal{S}_{\mathsf{ID}}$.
- $\mathsf{Sign}$: A probabilistic algorithm that accepts a message $m$, an identity $\mathsf{ID}$ and his/her secret key $\mathcal{S}_{\mathsf{ID}}$, outputs a signature $\sigma$ on $m$ and a partial message $m_1$.
- $\mathsf{Verify}$: A deterministic algorithm that accepts an identity of the sender, $\mathsf{ID}$, a partial message $m_1$ and a signature $\sigma$, outputs either true or $\perp$ to indicate whether the verification is successful or not. If the output is true, outputs the complete message $m$.

*Consistency*
For consistency of the scheme, we require

$$Pr\begin{pmatrix}(\mathtt{true}, m) \leftarrow \mathsf{Verify}(m_1, \sigma, \mathsf{ID});\\ (\sigma, m_1) \leftarrow \mathsf{Sign}(\mathsf{ID}, \mathcal{S}_{\mathsf{ID}}, m);\\ \mathcal{S}_{\mathsf{ID}} \leftarrow \mathsf{Extract}(\mathsf{ID})\end{pmatrix} = 1$$

holds with an overwhelming probability.

### 4.2 Formal Security Notion

In this section, we provide a formal security notion for an ID-based partial message recovery scheme. We provide a formal definition of existential unforgeability of an ID-based partial message recovery signature scheme under a chosen message attack, which is similar to the notion of existential unforgeability of an ID-based message recovery signature. It is defined using the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

- $\mathsf{Setup}$: $\mathcal{C}$ runs $\mathsf{Setup}$ for a given security parameter $\ell$ to obtain a public parameter param. The public key of the $PKG$, $P_{pub}$, is also obtained. The associated $PKG$'s secret key is kept by $\mathcal{C}$.
- $\mathsf{Extract\ Queries}$: $\mathcal{A}$ can request the private key corresponding to any identity, $\mathsf{ID}_i$ for $1 \leq i \leq q_{ex}$ where $q_{ex}$ denotes the number of extraction queries, which is polynomial in $l$. As a response to each query, $\mathcal{C}$ runs $\mathsf{Extract}$ taking $\mathsf{ID}_i$ as input and returns a resulting secret key $\mathcal{S}_{\mathsf{ID}_i}$.
- $\mathsf{Sign\ Queries}$: $\mathcal{A}$ can request a signature on a message $m_j$ for $1 \leq j \leq q_m$ where $q_m$ denotes the number of extraction queries, which is polynomial in $l$, for any identity $\mathsf{ID}_i$. In response, $\mathcal{C}$ runs $\mathsf{Extract}$ to get a secret key $\mathcal{S}_{\mathsf{ID}_i}$ associated with $\mathsf{ID}_i$. It then runs $\mathsf{Sign}$ taking $\mathsf{ID}_i$, $\mathcal{S}_{\mathsf{ID}_i}$ and $m_j$ as inputs and returns a resulting a signature $\sigma_j$ for the message $m_j$ and a partial message $m'_j$ related to $m_j$.
- $\mathsf{Verify\ Queries}$: Answers to these queries are not provided by $\mathcal{C}$ since $\mathcal{A}$ can compute them for himself using the $\mathsf{Verify}$ algorithm.
- $\mathsf{Output}$: Finally, $\mathcal{A}$ outputs a tuple $(\mathsf{ID}, m', \sigma)$. $\mathcal{A}$ wins the game if $\mathsf{Verify}(\mathsf{ID}_i, \sigma) \overset{?}{=} \mathtt{true}$ holds; no secret key for $\mathsf{ID}$ was issued in $\mathsf{Extract\ Queries}$ stage; and $(\sigma, m')$ was not obtained in $\mathsf{Sign\ Queries}$ stage.

The success probability of an adversary to win the game is defined by

$$\mathsf{Succ}_{\mathcal{A}}^{UF-IDPMRSS-CMA}(\ell).$$

**Definition 3.** *We say that an ID-based partial message recovery signature scheme is existentially unforgeable under a chosen message attack if the probability of success of any polynomially bounded adversary in the above game is negligible. In other words,*

$$\mathsf{Succ}_{\mathcal{A}}^{UF-IDPMRSS-CMA}(\ell) \leq \epsilon$$

### 4.3 A Concrete Scheme from Bilinear Pairing

- Setup: $PKG$ chooses a random number $s \in Z_q^*$ and sets $P_{pub} = sP$. $PKG$ also publishes system parameters $\{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, \lambda, P, H_0, H_1, F_1, F_2, k_1, k_2\}$, and keeps $s$ as the *master-key*, which is known only by itself. Here $|q| = k_1 + k_2$, $H_1 : \{0,1\}^* \to Z_q^*$, $H_0 : \{0,1\}^* \to \mathbb{G}_1^*$, $F_1 : \{0,1\}^{k_2} \to \{0,1\}^{k_1}$ and $F_2 : \{0,1\}^{k_1} \to \{0,1\}^{k_2}$ are four cryptographic hash functions.
- Extract: A user submits his/her identity information $\mathsf{ID}$ to $PKG$. $PKG$ computes the user's public key as $\mathsf{Q}_{\mathsf{ID}} = H_0(\mathsf{ID})$, and returns $\mathcal{S}_{\mathsf{ID}} = s\mathsf{Q}_{\mathsf{ID}}$ to the user as his/her private key.
- Sign: Let the message be $m = m_2 \parallel m_1$, here $m_2 \in \{0,1\}^{k_2}$.
  S1 $v = \hat{e}(P, P)^k$, where $k \in_R Z_q^*$
  S2 $f = F_1(m_2)||(F_2(F_1(m_2)) \oplus m_2)$
  S3 $r = H_1(v) + f \pmod q$
  S4 $c = H_1(m_1 \parallel r)$
  S5 $U = kP - c\mathcal{S}_{\mathsf{ID}}$.

  The signature is $(m_1, r, U)$. We note that the size of the message-signature pair is $|m_1 + r + U|$, which is $|m_1| + |q| + |\mathbb{G}_1|$.
- Verify: Given $\mathsf{ID}$, a partial message $m_1$, and a signature $(r, U)$, compute

  $$r - H_1(\hat{e}(U, P)\hat{e}(\mathsf{Q}_{\mathsf{ID}}, P_{pub})^{H_1(m_1 \| r)}) = f.$$

  and

  $$m_2 = [f]_{k_2} \oplus F_2([f]^{k_1}).$$

  Check whether $[f]^{k_1} \overset{?}{=} F_1(m_2)$ holds. If it holds with an equality, then accept this signature and output `true` and the complete message $m = m_1 \parallel m_2$. Otherwise, output $\perp$.

### 4.4 Security Analysis

**Theorem 3.** *Our ID-based partial message recovery scheme is complete and sound.*

*Proof.* The correctness of the scheme is justified as follows.

$$\begin{aligned}\hat{e}(U,P)\hat{e}(\mathsf{Q}_{\mathsf{ID}},P_{pub})^{H_1(m_1\|r)} &= \hat{e}(kP-c\mathcal{S}_{\mathsf{ID}},P)\hat{e}(\mathsf{Q}_{\mathsf{ID}},sP)^{H_1(m_1\|r)}\\ &= \hat{e}(kP-c\mathcal{S}_{\mathsf{ID}},P)\hat{e}(\mathsf{Q}_{\mathsf{ID}},sP)^{c}\\ &= \hat{e}(kP-c\mathcal{S}_{\mathsf{ID}},P)\hat{e}(cs\mathsf{Q}_{\mathsf{ID}},P)\\ &= \hat{e}(kP-c\mathcal{S}_{\mathsf{ID}},P)\hat{e}(c\mathcal{S}_{\mathsf{ID}},P)\\ &= \hat{e}(kP,P)\\ &= \hat{e}(P,P)^k\end{aligned}$$

Obtaining this value, we can compute

$$\begin{aligned}r-H_1(\hat{e}(U,P)e(\mathsf{Q}_{\mathsf{ID}},P_{pub})^{H_1(m_1\|r)}) &= r-H_1(\hat{e}(P,P)^k)\\ &= r-H_1(v)\\ &= f\end{aligned}$$

Since $f=F_1(m_2)||(F_2(F_1(m_2))\oplus m_2)$, then testing $[f]^{k_1}\overset{?}{=}F_1(m_2)$ must hold with equality. Therefore, we obtain $F_2([f]^{k_1})=F_2(F_1(m_2))$. Hence, to recover the message, we can compute

$$\begin{aligned}m_2 &= [f]_{k_2}\oplus F_2([f]^{k_1})\\ &= [f]_{k_2}\oplus F_2(F_1(m_2))\\ &= [F_1(m_2)||(F_2(F_1(m_2))\oplus m_2)]_{k_2}\oplus F_2(F_1(m_2))\\ &= (F_2(F_1(m_2))\oplus m_2)\oplus F_2(F_1(m_2))\\ &= m_2\end{aligned}$$

The complete message is recovered as $m=m_1\parallel m_2$. □

**Theorem 4.** *Our ID-based message recovery signature scheme is existentially unforgeable under a chosen message attack in the random oracle model, assuming the hardness of Computational Diffie-Hellman problem.*

*Proof.* The proof is similar to the proof of Theorem 2 and therefore it is omitted. □

### 4.5 Efficiency

The length of the signature of the scheme presented in this section is $|m_1+r+U|$, which equal to $|m_1|+|q|+|\mathbb{G}_1|$. The scheme can be used to recover a message $m$ of arbitrary length, where $m$ is represented as $m=m_1||m_2$. Using any of the families of curves described in [5], one can select $q$ to be a 170-bit prime and use a group $\mathbb{G}_1$ where each element is 171 bits. Hence, the total signature length is $|m_1|+341$ bits or $\frac{|m_1|}{8}+43$ bytes. With these parameters, security is approximately the same as a standard 1024-bit RSA signature, which is 128 bytes. We note that the overhead of our second scheme is identical to our first scheme.

## 5 Conclusion

In this paper, we presented the first ID-based short signature schemes. Our schemes are essentially ID-based message recovery signature schemes and ID-based partial message recovery signature schemes. The construction has opened a new area of research, namely how to shorten ID-based signature schemes. Unlike the previous contributions in constructing short signature schemes, our schemes *are ID-based.* We presented concrete schemes for ID-based message recovery signature scheme and ID-based partial message recovery signature scheme. The efficiency of both algorithms are as follows.

| | *Scheme 1* | *Scheme 2* |
|---|---|---|
| Total Length | $\vert q\vert + \vert\mathbb{G}_1\vert$ | $\vert m_1\vert + \vert q\vert + \vert\mathbb{G}_1\vert$ |
| Signature Length in Practice | 341 bits | $\vert m_1\vert + 341$ bits |
| Maximum size of $m$ | $k_2$ | arbitrary length |

**Acknowledgement.** The authors would like to express their gratitude thanks to Dr. Joonsang Baek for his fruitful discussion and suggestion on the security model and proof for our paper. We would also like to thank the anonymous referees of Financial Cryptography & Data Security 2005 for the suggestions to improve this paper. Finally, we would also like to thank Dr. David Pointcheval for his motivation, encouragement and suggestion on the security proof.

## References

1. M. Abe and T. Okamoto. A Signature Scheme with Message Recovery as Secure as Discrete Logarithm. *Advances in Cryptology - Asiacrypt 1999, Lecture Notes in Computer Science 1716*, pages 378 – 389, Springer-Verlag, Berlin, 1999.
2. D. Boneh and X. Boyen. Short Signatures Without Random Oracles. *Advances in Cryptology - Eurocrypt 2004, Lecture Notes in Computer Science 3027*, pages 56–73, Springer-Verlag, Berlin, 2004.
3. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. *Lecture Notes in Computer Science 2139*, pages 213+, Springer-Verlag, Berlin, 2001.
4. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and Verifiable Encrypted Signatures from Bilinear Maps. *Proceedings of Eurocrypt 2003, Lecture Notes in Computer Science 2656*, pages 416 – 432, Springer-Verlag, Berlin, 2003.
5. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. *Asiacrypt 2001, Lecture Notes in Computer Science*, pages 514–532, Springer-Verlag, Berlin, 2001.
6. S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal of Computing*, 17/2:281–308, 1988.
7. D. Naccache and J. Stern. Signing on a Postcard. *Financial Cryptography (FC2000), Lecture Notes in Computer Science 1962*, pages 121 – 135, Springer-Verlag, Berlin, 2000.

8. K. Nyberg and R. Rueppel. A New Signature Scheme based on the DSA, Giving Message Recovery. *Proceedings of the 1st ACM conference on communications and computer security*, pages 58 – 61, 1993.
9. D. Pointcheval and J. Stern. Security Proofs for Signature Schemes. *Advanced in Cryptology - Eurocrypt 1996, Lecture Notes in Computer Science 1070*, pages 387 – 398, Springer-Verlag, Berlin, 1996.
10. D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000.
11. A. Shamir. Identity-based cryptosystems and signature schemes. *Advances in Cryptology - Crypto '84, Lecture Notes in Computer Science 196*, pages 47–53, Springer-Verlag, Berlin, 1985.
12. R. Steinfeld, L. Bull, H. Wang, and J. Pieprzyk. Universal designated-verifier signatures. *Proceedings of Asiacrypt 2003, Lecture Notes in Computer Science 2894*, pages 523 – 543, Springer-Verlag, Berlin, 2003.
13. F. Zhang, R. Safavi-Naini, and W. Susilo. An Efficient Signature Scheme from Bilinear Pairings and Its Applications. *Public Key Cryptography (PKC) 2004, Lecture Notes in Computer Science 2947*, pages 277 – 290, Springer-Verlag, Berlin, 2004.

# Time Capsule Signature

Yevgeniy Dodis[1] and Dae Hyun Yum[1,2]

[1] Dept. of Computer Science, New York University, NY, USA
[2] Dept. of Electronic and Electrical Eng., POSTECH, Pohang, Korea
{dodis, dhyum}@cs.nyu.edu

**Abstract.** We introduce a new cryptographic problem called *time capsule signature*. Time capsule signature is a 'future signature' that becomes valid from a specific future time $t$, when a trusted third party (called *Time Server*) publishes some trapdoor information associated with the time $t$. In addition, time capsule signature should satisfy the following properties:

(1) If the signer wants, she can make her time capsule signature effective before the pre-defined time $t$.
(2) The recipient of 'future signature' can verify right away that the signature will become valid no later than at time $t$.
(3) Time Server need not contact any user at any time, and in fact does not need to know anything about the PKI employed by the users.
(4) Signatures completed by the signer before time $t$ are indistinguishable from the ones completed using the Time Server at time $t$.

We provide the rigorous definition of time capsule signature and the generic construction based on another new primitive of independent interest, which we call *identity-based trapdoor hard-to-invert relation* (ID-THIR). We also show an efficient construction of ID-THIRs (and, hence, time capsule signatures) in the random oracle model, and a less efficient construction in the standard model.

If the time $t$ is replaced by a specific event, the concept of time capsule signature can be generalized to *event capsule signature*.

## 1 Introduction

### 1.1 Time Capsule Signature

In an ordinary signature scheme, the validity of a signature value is determined at the point of signature generation and never changes (unless the signer's public key is revoked). Users cannot generate the so-called 'future signature' which is not currently valid but becomes valid from a future time $t$. A naive way to achieve this is signing with a statement such as 'the signature of message $m$ becomes valid from time $t$.' This, however, has several drawbacks. First, and least serious, the verifier is required to be aware of the current time. When time is generalized to arbitrary events (i.e., 'the signature of $m$ becomes valid if the event $e$ happens'), this becomes even more problematic. More seriously,

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 57–71, 2005.
© Springer-Verlag Berlin Heidelberg 2005

however, in the naive solution the signer herself loses control over the validity of the future signature, i.e., even the real signer cannot make her signature valid before time $t$. This means that either the signer has to wait until time $t$ — which could be undesirable in certain situations (e.g., if the borrower wants to quickly repay her debt before the actual due date to improve her credit history) — or the signer can issue a new, independent signature of $m$ before time $t$. The latter solution, however, can also be undesirable in certain situations. First, in case the message $m$ carries some monetary value, the signer needs to make sure that no "double spending" occurs (i.e., to somehow revoke the original signature, so that it does not become valid at time $t$). Second, the verifier now knows whether the message $m$ was signed in the 'future' or 'regular' way, which seems to be unnecessary in most situations.

Therefore, we would like a solution where the signer can issue a future signature so that at least the following properties are satisfied:

(1) At the time of creation, the recipient is sure that the signature will become valid by time $t$, even if the signer refuses to cooperate after she produces the future signature.
(2) The legal signer can make the future signature valid at any time after the initial creation.
(3) Irrespective of whether the signer validated the signature earlier, or it became "automatically valid" at time $t$, the resulting signatures are indistinguishable. In other words, the verifier after time $t$ cannot tell the lower-level details of how the signature became valid.

Of course, it is also crucial to specify the mechanism under which the signature can be "automatically" completed at time $t$ (which we call "hatching" as opposed to "pre-hatching" which can be done by the signer at any time). As we remarked, we cannot just make it valid at time $t$, since this requires the verifier to "know" the current time, and, more importantly, will not make the hatching indistinguishable from pre-hatching. Another option would be to use some "timed-release" primitive, such as timed signature [10], where the verifier knows that by investing some intensive computation, he can complete a future signature within some pre-specified time, even if the signer refuses to cooperate. However, this option is only approximate (i.e., the verifier can open the signature roughly by time $t$ depending on its computational capabilities), and, more importantly, forces the verifier to invest a considerable computational effort the moment the future signature was generated.

Finally, we can follow the approach of optimistic fair exchange protocols [1, 2, 14], where an "off-line" arbitrator (a trusted third party) can complete the signer's partial signature into the full signature, shall the signer refuse to cooperate. In particular, so called verifiably committed signatures of [14] seem to be ideally suited for this task. The main drawback of this solution is that the arbitrator, although only involved if the signer refuses to cooperate (say, before time $t$), has to be involved in a message-by-message manner. Thus, in our scenario of future signatures, — where by default the signer will not pre-hatch her signature, — the arbitrator would have to literally complete almost every signature separately. The latter, of course, makes the arbitrator quite "on-line" and whole paradigm very unattractive for our application.

Instead, we introduce *time capsule signatures*, where the arbitrator (which we call the *Time Server*)

(1) Does not ever need to contact users, know about the particular format of their signature, or be involved in any signature resolution protocols.
(2) At the beginning of time period $t$, outputs a single message $Z_t$, which automatically allows anybody to complete any future signature set to hatch at time $t$.

More specifically, time capsule signature is a 'future signature' that becomes valid from a specific future time $t$, when a trusted third party (called *Time Server*) publishes some trapdoor information associated with the time $t$. When Alice gives Bob her time capsule signature $\sigma'_t$ for a future time $t$, Bob can verify that Alice's time capsule signature will become valid from the time $t$. In addition, if Alice wishes, she can make her time capsule signature effective before the predefined time $t$. The assumption on Time Server is minimal, in that Time Server only publishes some information at the beginning of each time period and need not contact any user at any time. Finally, the concept of time capsule signature can be generalized to *event capsule signature*, where *Event Server* issues the notification information of specific events. The event capsule signature becomes valid if a specific event happens or the signer makes valid before the event occurs.

### 1.2 Our Contribution

We provide the rigorous definition of time (or event) capsule signature and the generic construction based on another new primitive of independent interest, which we call *identity-based trapdoor hard-to-invert relation* (ID-THIR). Intuitively, ID-THIR is given by a family $\mathcal{R}$ of relations $R_{id}$, where (1) it is easy to sample a random pair $(c, d) \in R_{id}$ and verify if the pair $(c, d)$ belongs to $R_{id}$; (2) for each identity $id$, there exists a trapdoor $\mathsf{td}_{id}$, which allows one to compute a random $d$ corresponding (w.r.t. $R_{id}$) to any given $c$ (The trapdoor $\mathsf{td}_{id}$'s can be efficiently computed from a single "master key" $\mathsf{mtd}_{\mathcal{R}}$); (3) without the trapdoor $\mathsf{td}_{id}$, it is hard to find a matching $d$ corresponding (w.r.t. $R_{id}$) to a randomly sampled $c$, even if one knows many trapdoors corresponding to identities $id' \neq id$.

Our construction of time (or event) capsule signatures from ID-THIR is very natural: the future signature of $m$ is $(\mathbf{Sig}(m||c||t), c)$, while the full hatched signature is $(\mathbf{Sig}(m||c||t), c, d)$, where '$\mathbf{Sig}$' is any ordinary signature, '$m$' a message, '$||$' the concatenation and $(c, d)$ a random lock/proof pair corresponding to the "identity" equal to time $t$ (or event $e$). The legal signer would sample $(c, d)$ and remember $d$ for pre-hatching, while the Time (or Event) Server would periodically publish the trapdoors $\mathsf{td}_t$ (or $\mathsf{td}_e$) which would allow anyone to hatch the signature by computing the corresponding $d$ from $c$. Moreover, hatching and pre-hatching would look the same, by the security properties of the ID-THIR scheme.

Finally, we give a generic construction of ID-THIR (and, therefore, time/event capsule signature). In the standard model, the construction is mainly of theoretical interest, as it relied on non-interactive witness indistinguishable proofs of knowledge. Nevertheless, it shows that our primitives exist if trapdoor one-way permutations exist. On a practical front, we give several very efficient implementations of ID-THIR in the random oracle model. Generically, we show that in the random oracle model one can construct our primitives from mere one-way functions. Concretely, we show very efficient instantiations based on RSA or discrete log in the Gap Diffie-Hellman groups [24, 9].

### 1.3 Related Work

As we pointed out, there are two main lines of work related to time capsule signature, depending on whether or not the trusted third party is involved.

The first approach, which is that of timed-release cryptography, is to ensure that the encryption, commitment or signature can be opened in a brute force way by solving some time-consuming, but computationally feasible problem. For example, Dwork and Naor [15] used such moderately hard functions in order to deter abuse of resources, such as spamming. Bellare and Goldwasser [3, 4] suggested "(verifiable) time capsules"[1] for key escrowing in order to deter widespread wiretapping. There, the main issue is the verification at escrow time that the right key will be recovered. Rivest, Shamir and Wagner [28] suggested "time-lock puzzle," where the goal is to design "inherently sequential puzzles" which are resistant to parallel attacks. However, they did not address verifiability at escrow time. The latter was formally addressed by Boneh and Naor [10], who defined (verifiable) timed commitments. As one of their applications, they get an analog of our time capsule signature (termed "timed signature"), where the future signature can either be opened by the signer, or by the recipient — the latter if the recipient solves a moderately hard problem. More recent advances were made by [19, 20].

The second approach, based on the trusted third party, has two main flavors: optimistic fair exchange of digital signatures, and identity-based future encryption. In the former case, the server needs to resolve all individual signatures where the signer refused to validate the signature (say, by a given time $t$).[2] Representative examples include [1, 2, 8, 14]. In contrast, in our model we insist that users do not communicate ever with the trusted server.

In the case of future encryption [7, 6, 25], the main problem addressed was that the sender wants to ensure that the message would remain hidden before the Time Server would publish the corresponding trapdoor. However, this is orthogonal to our model, where we want to "encrypt" a *signature* on a *public* message. Thus, we do not need to hide the message (and can even leak partial information about the full signature, as long as the full signature is hidden). On

[1] Which should not be confused with our time capsule signatures, which are totally different.

[2] In fact, in some of the solutions the clients additionally need to either register their keys with the server, or have an interactive resolution protocol.

the other hand, we have to resolve two crucial complications not present in the above scenario: (1) the future signature has to be verifiable right away, to ensure the recipient it will be successfully completed at time $t$; (2) the sender can prehatch the signature in a manner indistinguishable from the regular hatching at time $t$. Not surprisingly, the solutions above all utilized some kind of identity-based encryption, and do not seem to be useful for our time-capsule signatures.

## 2 Primitives

### 2.1 $\Sigma$-Protocol

A $\Sigma$-protocol [13] is an efficient 3-move two-party protocol between the prover and the verifier on a common input $x \in \mathcal{L}_R$, where $\mathcal{L}_R$ is a language for an $\mathsf{NP}$ relation $R$. Besides $x$, a valid $\mathsf{NP}$-witness $w$ for $x$ is also given to the prover as a private input. The prover first sends a commitment message $a$ to the receiver. After receiving the commitment message $a$, the verifier sends a challenge message $b$ to the prover. Finally, the prover sends a response message $z$ to the verifier who decides to output 1 (accept) or 0 (reject) based on the input $x$ and the transcript $\pi = \{a, b, z\}$. The transcript $\pi$ is valid if the verifier outputs 1 (accept). A binary $\Sigma$-protocol is a special case of $\Sigma$-protocol where the challenge message takes only a binary value (0 or 1).

A $\Sigma$-protocol should satisfy three properties: correctness, special soundness, and special (honest-verifier) zero-knowledge. Correctness property states that for all $x \in \mathcal{L}_R$ and all valid witnesses $w$ for $x$, if the prover and the verifier follow the protocol honestly, the verifier must output 1 (accept). Special soundness property says that there is an efficient extraction algorithm (called a knowledge extractor) $\mathsf{Ext}$ that on input $x \in \mathcal{L}_R$ and two valid transcripts $\pi_1, \pi_2$ with the same commitment message outputs $w$ such that $(x, w) \in R$. Special zero-knowledge property says that there is an efficient simulation algorithm (called a simulator) $\mathsf{Sim}$ that on input $x \in \mathcal{L}_R$ and any challenge message $b$, outputs a valid transcript $\pi' = \{a', b, z'\}$. Moreover, the distribution of $(a', z')$ is computationally indistinguishable from the corresponding distribution on $(a, z)$ produced by the prover knowing a valid witness $w$ for $x$ and the verifier. This is true even if the distinguisher knows the witness $w$.

A function $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way function, if there exists a polynomial time algorithm which computes $f(x)$ correctly for all $x$ and the following probability is negligible for all PPT (Probabilistic Polynomial Time) algorithm $A$: $\Pr[f(x') = y \mid x \leftarrow \{0,1\}^k; y = f(x); x' \leftarrow A(y, 1^k)]$. It is known that any language in $\mathsf{NP}$ has a $\Sigma$-protocol if one-way functions exist [21, 16]. Of course, specific languages can have much more efficient $\Sigma$-protocols. A $\Sigma$-protocol can also be transformed into a signature scheme by using the Fiat-Shamir heuristic [18]. To sign a message $m$, the legal signer produces a valid transcript $\pi = \{a, b, z\}$ of the $\Sigma$-protocol, where $b = H(a, m)$ and $H(\cdot)$ is a cryptographic hash function modelled as a random function. The signature scheme obtained by applying the Fiat-Shamir heuristic to the $\Sigma$-protocol is secure in the random oracle model [5, 27]. It is also known that the Fiat-Shamir heuristic gives a non-interactive proof of knowledge in the random oracle model (i.e., the witness can be extracted by rewinding the adversary).

If there are two $\Sigma$-protocols, i.e., $\Sigma_1$ for $R_1$ and $\Sigma_2$ for $R_2$, we can construct another $\Sigma$-protocol $\Sigma_{OR}$ (called OR-proof) [13] which allows the prover to show that given two inputs $x_1, x_2$, he knows $w$ such that either $(x_1, w) \in R_1$ or $(x_2, w) \in R_2$ without revealing which is the case (called the witness indistinguishability property [17]). By applying the Fiat-Shamir heuristic to the OR-proof $\Sigma_{OR}$, we get a signature scheme (called OR-signature) secure in the random oracle model such that a valid signature can be generated by the signer who knows a valid witness $w$ corresponding to either of the two inputs $x_1, x_2$. It is known that the Fiat-Shamir heuristic does not affect the witness indistinguishability property of the $\Sigma$-protocol.

### 2.2 Identity-Based Trapdoor Hard-to-Invert Relation

A (binary) relation $R$ is a subset of $\{0,1\}^* \times \{0,1\}^*$ and the language $\mathcal{L}_R$ is the set of $\alpha$'s for which there exist $\beta$ such that $(\alpha, \beta) \in R$, i.e., $\mathcal{L}_R = \{\alpha \mid \exists \beta\ [(\alpha, \beta) \in R]\}$. We assume that (1) there is an efficient algorithm to decide whether $\alpha \in \mathcal{L}_R$ or not, (2) if $(\alpha, \beta) \in R$, then the length of $\beta$ is polynomially bounded in $|\alpha|$, and (3) there exists a short description $D_R$ which specifies the relation $R$.

We also assume that the membership in $f(X)$ can be efficiently determined for a (trapdoor) one-way function $f : X \to Y$.

**Definition 1.** *An* identity-based trapdoor hard-to-invert relation (ID-THIR) *is a set of relations $\mathcal{R} = \{R_{id} \mid id \in I_{\mathcal{R}}\}$, where each relation $R_{id}$ is trapdoor hard-to-invert relation (i.e., sampling a random lock/proof pair $(c, d) \in R_{id}$ is easy but finding a proof for a given lock is difficult without knowing the trapdoor* $\mathsf{td}_{id}$*) and there is a master trapdoor* $\mathsf{mtd}_{\mathcal{R}}$ *for extracting the trapdoor* $\mathsf{td}_{id}$ *of each relation $R_{id}$.* ID-THIR *can also be specified by 5-tuple of PPT algorithms* $(\mathsf{Gen}, \mathsf{Sample}, \mathsf{Check}, \mathsf{Extract}, \mathsf{Invert})$ *such that:*

- Gen. *This algorithm is used to generate $\mathcal{R} = \{R_{id} \mid id \in I_{\mathcal{R}}\}$, where $I_{\mathcal{R}}$ is a finite set of indices.* $\mathsf{Gen}(1^k)$ *returns $D_{\mathcal{R}}$ (the description of $\mathcal{R}$) and* $\mathsf{mtd}_{\mathcal{R}}$ *(the master trapdoor).*
- Sample. *This sampling algorithm takes $(D_{\mathcal{R}}, id)$ as input and* $\mathsf{Sample}_{D_{\mathcal{R}}}(id)$ *returns a random lock/proof pair $(c, d) \in R_{id}$.*
- Check. *This algorithm is used to check the validity of the proof. If $(c, d) \in R_{id}$, then* $\mathsf{Check}_{D_{\mathcal{R}}, id}(c, d)$ *returns* 1 *(accept). Otherwise, it returns* 0 *(reject).*
- Extract. *This algorithm is used to extract the trapdoor of each relation by using* $\mathsf{mtd}_{\mathcal{R}}$. $\mathsf{Extract}_{\mathsf{mtd}_{\mathcal{R}}}(id)$ *returns the trapdoor* $\mathsf{td}_{R_{id}}$ *of the relation $R_{id}$.*
- Invert. *This algorithm is used to find a proof $d$ for a given $c \in \mathcal{L}_{R_{id}}$ by using the trapdoor* $\mathsf{td}_{R_{id}}$. *If $c \in \mathcal{L}_{R_{id}}$, then* $\mathsf{Invert}_{\mathsf{td}_{R_{id}}}(c)$ *returns a proof $d$ such that $(c, d) \in R_{id}$.*

*Let $(c, d) \leftarrow$* $\mathsf{Sample}_{D_{\mathcal{R}}}(id)$ *and $\widetilde{d} \leftarrow$* $\mathsf{Invert}_{\mathsf{td}_{R_{id}}}(c)$. *Correctness property states that* $\mathsf{Check}_{D_{\mathcal{R}}, id}(c, d) = 1$ *and* $\mathsf{Check}_{D_{\mathcal{R}}, id}(c, \widetilde{d}) = 1$, *and ambiguity property states that $(c, d)$ and $(c, \widetilde{d})$ are computationally indistinguishable, even if the* *distinguisher knows the master key* $\mathsf{mtd}_{\mathcal{R}}$*. Let* $O_{\mathsf{Extract}}$ *be the oracle simulating the trapdoor extraction procedure* $\mathsf{Extract}$ *and* $Query(A, O_{\mathsf{Extract}})$ *the set of queries an algorithm* $A$ *asked to* $O_{\mathsf{Extract}}$*. One-wayness property states that the following probability is negligible for all PPT algorithm* $A = (A_1, A_2)$*:*

$$\Pr[\,\mathsf{Check}_{D_{\mathcal{R}},id}(c, \widehat{d}) = 1 \ \wedge\ id \not\in Query(A, O_{\mathsf{Extract}}) \mid (D_{\mathcal{R}}, \mathsf{mtd}_{\mathcal{R}}) \leftarrow \mathsf{Gen}(1^k);\ (id, h) \leftarrow A_1^{O_{\mathsf{Extract}}}(D_{\mathcal{R}});\ (c, d) \leftarrow \mathsf{Sample}_{D_{\mathcal{R}}}(id);\ \widehat{d} \leftarrow A_2^{O_{\mathsf{Extract}}}(D_{\mathcal{R}}, c, h)]$$

*Soundness property states that the following probability is negligible for all algorithm* $B$*:*

$$\Pr[\,R_{id} \in \mathcal{R} \ \wedge\ c \in \mathcal{L}_{R_{id}} \ \wedge\ \mathsf{Check}_{D_{\mathcal{R}},id}(c, \widetilde{d}) = 0 \mid (D_{\mathcal{R}}, \mathsf{mtd}_{\mathcal{R}}) \leftarrow \mathsf{Gen}(1^k);\ (c, id) \leftarrow B(D_{\mathcal{R}});\ \mathsf{td}_{R_{id}} \leftarrow \mathsf{Extract}_{\mathsf{mtd}_{\mathcal{R}}}(id);\ \widetilde{d} \leftarrow \mathsf{Invert}_{\mathsf{td}_{R_{id}}}(c)\,]$$

*If* ID-THIR *satisfies these four properties, we say that* ID-THIR *is secure.*

Construction. Each trapdoor hard-to-invert relation $R_{id}$ in ID-THIR $\mathcal{R} = \{R_{id} \mid id \in I_{\mathcal{R}}\}$ looks like a trapdoor one-way function. However, there is an important difference: we can sample a random lock/proof pair $(c, d) \in R_{id}$ but may not necessarily be able to compute a lock $c$ for a given proof $d$. Therefore, we can show that a trapdoor one-way function implies a trapdoor hard-to-invert relation but cannot prove the reverse direction. While the concept of ID-THIR also seems very general, the construction is not trivial. For example, it is not obvious whether identity-based encryption (IBE) [32, 7] implies ID-THIR or not, since IBE does not automatically guarantee the ambiguity property of ID-THIR.[3] Now, we provide our general construction of ID-THIR.

**Theorem 1.** *If there is a one-way function, there exists a secure* ID-THIR *in the random oracle model.*

**Proof:** Assume that there is a one way function $f : X \to Y$. We can build a secure signature scheme $(\mathbf{Set}, \mathbf{Sig}, \mathbf{Ver})$ from the one-way function $f$, since secure signatures exist if and only if one-way functions exist [29]. Let $\Sigma_{\mathbf{S\ g}}$ be the $\Sigma$-protocol for the knowledge of a signature value $\mathbf{Sig}(m)$ on a common input $m \in M$ and $\Sigma_f$ for the knowledge of a pre-image of a common input $f(x) \in f(X)$. If we denote by $\Sigma_{OR}$ the OR-proof for $\Sigma_{\mathbf{S\ g}}$ or $\Sigma_f$, we can obtain an OR-signature scheme $(\mathbf{Set}^{OR}, \mathbf{Sig}^{OR}, \mathbf{Ver}^{OR})$ by applying the Fiat-Shamir heuristic to $\Sigma_{OR}$. The OR-signature is secure in the random oracle model and $\mathbf{Set}^{OR}$ can be implicitly defined by $\Sigma_{\mathbf{S\ g}}$ and $\Sigma_f$.

Now, we define the identity-based trapdoor hard-to-invert relation $\mathcal{R}_{OR} = \{R_m \mid m \in M\}$ where $R_m = \{(y, \pi) \mid y = f(x) \text{ for } x \in X,\ \pi \text{ is an OR-signature on } m \| f(x) \text{ for the knowledge of a pre-image of } f(x) \text{ or } \mathbf{Sig}(m)\}$ and the algorithms (Gen, Sample, Check, Extract, Invert) as follows; Gen chooses $(pk, sk) \leftarrow \mathbf{Set}(1^k)$

[3] The decryption algorithm of IBE does not necessarily recover the temporary random number used in the encryption algorithm. For example, see [7].

and outputs $D_{\mathcal{R}_{OR}} = pk$, $\mathsf{mtd}_{\mathcal{R}_{OR}} = sk$ (technically, $D_{\mathcal{R}_{OR}}$ should also contain the one-way function $f$). We assume that message space $M$ is known implicitly. On input $id = m$, $\mathsf{Sample}$ randomly chooses $x \in X$ and generates an OR-signature $\pi$ for the knowledge of a pre-image of $f(x)$ or $\mathbf{Sig}(m)$. $\mathsf{Sample}$ outputs a lock/proof pair $(c, d) = (f(x), \pi)$. For a given $(id, c, d) = (m, f(x), \pi)$, $\mathsf{Check}$ verifies whether $\pi$ is a valid OR-signature for a pre-image of $f(x)$ or $\mathbf{Sig}(m)$. $\mathsf{Extract}$ takes as input $id = m$ and outputs $\mathsf{td}_{R_m} = \mathbf{Sig}(m)$. On input $(id, c) = (m, f(x))$, $\mathsf{Invert}$ knowing $\mathsf{td}_{R_m} = \mathbf{Sig}(m)$ generates an OR-signature $\pi$ for the knowledge of a pre-image of $f(x)$ or $\mathbf{Sig}(m)$.

Correctness property is obvious and ambiguity property results from the fact that the OR-proof $\Sigma_{OR}$ is witness indistinguishable. Now, consider the one-wayness property. The attacker $A$ against $\mathcal{R}_{OR}$ gets $D_{\mathcal{R}_{OR}} = pk$ as input and has access to the signing oracle $O_{\mathbf{S\ g}}$. $A$ wins if it comes up with $m$ which was not queried to $O_{\mathbf{S\ g}}$ such that for a given lock $f(x) \in \mathcal{L}_{R_m}$, $A$ can find an OR-signature $\pi$ for the knowledge of a pre-image of $f(x)$ or $\mathbf{Sig}(m)$. However, the Fiat-Shamir proof is actually proof of knowledge and the ability to come up with a valid proof implies that we can extract a valid witness which is either a new signature value or a pre-image of the one-way function. Therefore, if $A$ succeeds, we can either forge an ordinary signature or invert the one-way function, both of which easily lead to contradiction to the security of the underlying signature scheme and one-way function. Finally, the soundness property can be checked from the correctness property of the OR-proof $\Sigma_{OR}$. □

*Remark 1.* ($\Sigma$-PROTOCOLS) The $\Sigma$-protocol $\Sigma_f$ for the knowledge of a pre-image of a one-way function and $\Sigma_{\mathbf{S\ g}}$ for the knowledge of a signature value can be constructed in generic ways [16]. However, there exist very efficient $\Sigma$-protocols for specific cases. For example, $\Sigma$-protocol in [22] can be used for the RSA function or the FDH signature scheme [5], and $\Sigma$-protocol in [31] can be applied to the discrete logarithm function or the BLS signature scheme [9]. While $\Sigma$-protocols for the knowledge of a signature value in [22, 31] require the random oracle model, $\Sigma$-protocols for the knowledge of a signature value without the random oracle model can be founded in [11, 12].

*Remark 2.* (ALTERNATIVE TO THE FIAT-SHAMIR PROOF – I) Notice that the proof of Theorem 1 only requires the following properties from the Fiat-Shamir proof: (1) witness indistinguishability and (2) proof of knowledge. Therefore, we can use the straight-line extractable WI proof [26] instead of the Fiat-Shamir proof. Like the Fiat-Shamir proof, the construction of the straight-line extractable WI proof starts with $\Sigma$-protocol but the length of the resulting proof is much longer. However, non-programmable random oracle can be used and better exact security is obtained. Therefore, the choice depends on the tradeoff between efficiency and exact security.

*Remark 3.* (ALTERNATIVE TO THE FIAT-SHAMIR PROOF – II) Instead of the Fiat-Shamir proof, we can also use non-interactive witness indistinguishable proofs of knowledge (for 'I know the pre-image of $f(x)$' or 'I know the signature value **Sig**$(m)$'). In this case, we do not need the random oracle and can use instead a common reference string (which can be included in the public key $pk$). However, the best known way of constructing non-interactive witness indistinguishable proofs of knowledge requires the existence of trapdoor one-way permutations [30] and is extremely inefficient. Nevertheless, this observation leads to the following corollary.

**Corollary 1.** *If there is a trapdoor one-way permutation, there exists a secure* ID-THIR *in the standard model.*

## 3 Time Capsule Signature

### 3.1 Definition

**Definition 2.** *A* time capsule signature scheme *is specified by an 8-tuple of* PPT *algorithms* $(\mathsf{Setup}^{\mathsf{TS}}, \mathsf{Setup}^{\mathsf{User}}, \mathsf{TSig}, \mathsf{TVer}, \mathsf{TRelease}, \mathsf{Hatch}, \mathsf{PreHatch}, \mathsf{Ver})$ *such that:*

- $\mathsf{Setup}^{\mathsf{TS}}$*. This setup algorithm is run by Time Server. It takes a security parameter as input and returns a public/private time release key pair* $(\mathsf{TPK}, \mathsf{TSK})$.
- $\mathsf{Setup}^{\mathsf{User}}$*. This seup algorithm is run by each user. It takes a security parameter as input and returns the user's public/private key pair* $(\mathsf{PK}, \mathsf{SK})$*.*
- $\mathsf{TSig}$*. The time capsule signature generation algorithm* $\mathsf{TSig}$ *takes as input* $(m, \mathsf{SK}, \mathsf{TPK}, t)$*, where* $t$ *is the specific time from which the signature becomes valid. It outputs a time capsule signature* $\sigma'_t$*.*
- $\mathsf{TVer}$*. The time capsule signature verification algorithm* $\mathsf{TVer}$ *takes as input* $(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, t)$ *and outputs* $1$ *(accept) or* $0$ *(reject).*
- $\mathsf{TRelease}$*. The time release algorithm* $\mathsf{TRelease}$ *is run by Time Server and takes as input* $(t, \mathsf{TSK})$*. At the beginning of each time period* $t$*, Time Server publishes* $Z_t = \mathsf{TRelease}(t, \mathsf{TSK})$*. Note that Time Server dose not contact any user at any time and need not know anything about the users.*
- $\mathsf{Hatch}$*. This algorithm is run by any party and is used to open a valid time capsule signature which became mature. It takes as input* $(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, Z_t)$ *and returns the hatched signature* $\sigma_t$*.*
- $\mathsf{PreHatch}$*. This algorithm is run by the signer and used to open a valid time capsule signature which is not mature yet. It takes as input* $(m, \sigma'_t, \mathsf{SK}, \mathsf{TPK}, t)$ *and returns the pre-hatched signature* $\sigma_t$*.*
- $\mathsf{Ver}$*. This algorithm is used to verify a hatched (or pre-hatched) signature.* $\mathsf{Ver}$ *takes as input* $(m, \sigma_t, \mathsf{PK}, \mathsf{TPK}, t)$ *and returns* $1$ *(accept) or* $0$ *(reject).*

*Correctness property states that*

- $\mathsf{TVer}(m, \mathsf{TSig}(m, \mathsf{SK}, \mathsf{TPK}, t), \mathsf{PK}, \mathsf{TPK}, t) = 1$ *and*
- $\mathsf{Ver}(m, \sigma_t, \mathsf{PK}, \mathsf{TPK}, t) = 1$*, where* $\sigma_t = \mathsf{Hatch}(m, \mathsf{TSig}(m, \mathsf{SK}, \mathsf{TPK}, t), \mathsf{PK}, \mathsf{TPK}, Z_t)$ *or* $\sigma_t = \mathsf{PreHatch}(m, \mathsf{TSig}(m, \mathsf{SK}, \mathsf{TPK}, t), \mathsf{SK}, \mathsf{TPK}, t)$*.*

*Ambiguity property states that*

- *The "hatched signature"* $\sigma_t = \mathsf{Hatch}(m, \mathsf{TSig}(m, \mathsf{SK}, \mathsf{TPK}, t), \mathsf{PK}, \mathsf{TPK}, Z_t)$ *is computationally indistinguishable from the "pre-hatched signature"* $\sigma_t = \mathsf{PreHatch}(m, \mathsf{TSig}(m, \mathsf{SK}, \mathsf{TPK}, t),\ \mathsf{SK}, \mathsf{TPK}, t)$*, even if the distinguisher knows* $\mathsf{TSK}$*.*

The security of time capsule signatures consists of ensuring three aspects: security against the signer Alice, security against the verifier Bob, and security against Time Server. In the following, the oracle simulating the time capsule signature generation algorithm $\mathsf{TSig}$ is denoted by $O_{\mathsf{TSig}}$, the oracle simulating the time release algorithm $\mathsf{TRelease}$ by $O_{\mathsf{TR}}$, and the oracle simulating $\mathsf{PreHatch}$ by $O_{\mathsf{PreH}}$. The oracle $O_{\mathsf{TSig}}$ takes $(m, t)$ as input and returns Alice's time capsule signature $\sigma'_t$.[4] The oracle $O_{\mathsf{PreH}}$ takes $(m, t, \sigma'_t)$ as input and returns Alice's pre-hatched signature $\sigma_t$.

Security against Alice. We require that any PPT adversary $A$ succeeds with at most negligible probability in the following experiment.

$$\begin{aligned}
\mathsf{Setup}^{\mathsf{TS}}(1^k) &\rightarrow (\mathsf{TPK}, \mathsf{TSK})\\
(m, t, \sigma'_t, \mathsf{PK}) &\leftarrow A^{O_{\mathsf{TR}}}(\mathsf{TPK})\\
Z_t &\leftarrow \mathsf{TRelease}(t, \mathsf{TSK})\\
\sigma_t &\leftarrow \mathsf{Hatch}(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, Z_t)\\
\text{success of } A &= [\mathsf{TVer}(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, t) \overset{?}{=} 1 \ \wedge\ \mathsf{Ver}(m, \sigma_t, \mathsf{PK}, \mathsf{TPK}, t) \overset{?}{=} 0]
\end{aligned}$$

In other words, Alice should not be able to produce a time capsule signature $\sigma'_t$, where $\sigma'_t$ looks good to Bob but cannot be hatched into Alice's full signature by the honest Time Server.

Security against Bob. We require that any PPT adversary $B$ succeeds with at most negligible probability in the following experiment.

$$\begin{aligned}
\mathsf{Setup}^{\mathsf{TS}}(1^k) &\rightarrow (\mathsf{TPK}, \mathsf{TSK})\\
\mathsf{Setup}^{\mathsf{User}}(1^k) &\rightarrow (\mathsf{PK}, \mathsf{SK})\\
(m, t, \sigma_t) &\leftarrow B^{O_{\mathsf{TSig}}, O_{\mathsf{TR}}, O_{\mathsf{PreH}}}(\mathsf{PK}, \mathsf{TPK})\\
\text{success of } B &= [\mathsf{Ver}(m, \sigma_t, \mathsf{PK}, \mathsf{TPK}, t) \overset{?}{=} 1 \ \wedge\ t \notin \mathit{Query}(B, O_{\mathsf{TR}})\\
&\qquad \wedge\ (m, t, \cdot) \notin \mathit{Query}(B, O_{\mathsf{PreH}})]
\end{aligned}$$

where $\mathit{Query}(B, O_{\mathsf{TR}})$ is the set of queries $B$ asked to the time release oracle $O_{\mathsf{TR}}$, and $\mathit{Query}(B, O_{\mathsf{PreH}})$ is the set of *valid* queries $B$ asked to the oracle $O_{\mathsf{PreH}}$ (i.e., $(m, t, \sigma'_t)$ such that $\mathsf{TVer}(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, t) = 1$). In other words, Bob should not be able to open a pre-mature time capsule signature without help of the singer or Time Server. Notice that Bob can make any time release query to

[4] We assume that the adversary attacks an honest user Alice. The adversary can collude with all other (dishonest) users.

$O_{\mathsf{TR}}$ except the target time $t$. Therefore, the above experiment requires strong security guaranteeing both forward and backward secrecy.

Security against Time Server. We require that any PPT adversary $C$ succeeds with at most negligible probability in the following experiment.

$$\begin{aligned}
&\mathsf{Setup}^{\mathsf{TS}^*}(1^k) \to (\mathsf{TPK}, \mathsf{TSK}^*)\\
&\mathsf{Setup}^{\mathsf{User}}(1^k) \to (\mathsf{PK}, \mathsf{SK})\\
&\quad (m, t, \sigma_t) \leftarrow C^{O_{\mathsf{TSig}}, O_{\mathsf{PreH}}}(\mathsf{PK}, \mathsf{TPK}, \mathsf{TSK}^*)\\
&\text{success of } C = [\mathsf{Ver}(m, \sigma_t, \mathsf{PK}, \mathsf{TPK}, t) \overset{?}{=} 1 \ \wedge \ (m, \cdot) \notin Query(C, O_{\mathsf{TSig}})]
\end{aligned}$$

where $\mathsf{Setup}^{\mathsf{TS}^*}$ denotes the run of $\mathsf{Setup}^{\mathsf{TS}}$ with a dishonest Time Server (run by $C$), $\mathsf{TSK}^*$ is $C$'s state after this run, and $Query(C, O_{\mathsf{TSig}})$ is the set of queries $C$ asked to the time capsule signature generation oracle $O_{\mathsf{TSig}}$ (i.e., $(m, t') \notin Query(C, O_{\mathsf{TSig}})$ for all $t'$). In other words, Time Server should not be able to produce a valid hatched or pre-hatched signature on $m$ of Alice without explicitly asking Alice to produce a time capsule signature on $m$.

### 3.2 Generic Construction Based on ID-THIR

The scheme. Let $(\mathbf{Set}, \mathbf{Sig}, \mathbf{Ver})$ be an ordinary signature scheme and ($\mathsf{Gen}$, $\mathsf{Sample}$, $\mathsf{Check}$, $\mathsf{Extract}$, $\mathsf{Invert}$) be the procedures for ID-THIR.

- $\mathsf{Setup}^{\mathsf{TS}}$. Time Server chooses $(D_{\mathcal{R}}, \mathsf{mtd}_{\mathcal{R}})$ by running $\mathsf{Gen}(1^k)$ and sets $(\mathsf{TPK}, \mathsf{TSK}) = (D_{\mathcal{R}}, \mathsf{mtd}_{\mathcal{R}})$.
- $\mathsf{Setup}^{\mathsf{User}}$. Each user chooses $(pk, sk)$ by running $\mathbf{Set}(1^k)$ and sets $(\mathsf{PK}, \mathsf{SK}) = (pk, sk)$.
- $\mathsf{TSig}$. To generate a time capsule signature on a message $m$ for time $t$, the signer gets a random lock/proof pair $(c, d)$ from $\mathsf{Sample}_{D_{\mathcal{R}}}(t)$ and computes $s = \mathbf{Sig}_{sk}(m||c||t)$. The time capsule signature value $\sigma'_t$ is $(s, c)$ and the signer stores the proof $d$ for later use.
- $\mathsf{TVer}$. For a given time capsule signature $\sigma'_t = (s, c)$, the verifier checks that $c \in \mathcal{L}_{R_t}$ and $s$ is a valid signature on $m||c||t$ by running $\mathbf{Ver}_{pk}(m||c||t, s)$.
- $\mathsf{TRelease}$. For a given time value $t$, Time Server computes $\mathsf{td}_{R_t} = \mathsf{Extract}_{\mathsf{mtd}_{\mathcal{R}}}(t)$ and publishes $Z_t = \mathsf{td}_{R_t}$.
- $\mathsf{Hatch}$. To open a mature time capsule signature $\sigma'_t = (s, c)$, a party computes $\widetilde{d} = \mathsf{Invert}_{\mathsf{td}_{R_t}}(c)$ and returns the hatched signature $\sigma_t = (s, c, \widetilde{d})$.
- $\mathsf{PreHatch}$. To open a valid pre-mature time capsule signature $\sigma'_t = (s, c)$, the signer returns the pre-hatched signature $\sigma_t = (s, c, d)$ where the proof $d$ is a stored value in the stage of $\mathsf{TSig}$.
- $\mathsf{Ver}$. For a given hatched (or pre-hatched) signature $\sigma_t = (s, c, d)$, the verifier checks the lock/proof pair by running $\mathsf{Check}_{D_{\mathcal{R}}, t}(c, d)$. Then, he verifies that $s$ is a valid signature on $m||c||t$ by running $\mathbf{Ver}_{pk}(m||c||t, s)$.

The correctness property and the ambiguity property of the scheme are obvious from the properties of ID-THIR. We now analyze its security.

**Theorem 2.** *The time capsule signature scheme presented above is secure if the underlying ordinary signature scheme and the* $\mathsf{ID\text{-}THIR}$ *are secure.*

**Proof:** We prove the security against Alice, Bob, and Time Server.

Security against Alice. Security against Alice follows unconditionally. A valid time capsule signature $\sigma'_t = (s, c)$ satisfies that $c \in \mathcal{L}_{R_t}$ and $\mathbf{Ver}_{pk}(m||c||t, s) = 1$. If Time Server releases $\mathsf{td}_t = \mathsf{Extract}_{\mathsf{mtd}_{\mathcal{R}}}(t)$, any party can obtain a proof $\widetilde{d} = \mathsf{Invert}_{\mathsf{td}_t}(c)$ for the lock $c \in \mathcal{L}_{R_t}$. By the correctness property of $\mathsf{ID\text{-}THIR}$, $\mathsf{Check}_{D_{\mathcal{R}},t}(c, \widetilde{d}) = 1$ always holds. Therefore, the hatched signature $\sigma_t = (s, c, \widetilde{d})$ passes the verification algorithm $\mathsf{Ver}$.

Security against Bob. To show security against Bob, we convert any attacker $B$ that attacks our time capsule signature scheme into an inverter $Inv$ of $\mathsf{ID\text{-}THIR}$. Recall that $Inv$ gets $D_{\mathcal{R}}$ as input and has access to the trapdoor extraction oracle $O_{\mathsf{Extract}}$. $Inv$ wins if it comes up with $id$ which was not queried to $O_{\mathsf{Extract}}$ s.t. for a given lock $c \in L_{R_{id}}$, $Inv$ can find a proof $d$ for $c$. On the other hand, $B$ expects $(\mathsf{PK}, \mathsf{TPK})$ as input and has access to $O_{\mathsf{TSig}}, O_{\mathsf{TR}}, O_{\mathsf{PreH}}$. $B$ wins if it forges a hatched (or pre-hatched) signature $\sigma_t$ of some message $m$ without asking $t$ to $O_{\mathsf{TR}}$ or $(m, t, \sigma'_t)$ to $O_{\mathsf{PreH}}$. Let $(m_B, t_B, \sigma_{t_B})$ be the successful forgery of the attacker $B$. We can assume that $B$ obtained the corresponding time capsule signature $\sigma'_{t_B}$ from $O_{\mathsf{TSig}}$, since the underlying ordinary signature scheme $(\mathbf{Set}, \mathbf{Sig}, \mathbf{Ver})$ is existentially unforgeable against chosen message attacks.

When $Inv$ receives $D_{\mathcal{R}}$ from an $\mathsf{ID\text{-}THIR}$ challenger $\mathcal{C}$, it begins simulating the attack environment of $B$. $Inv$ picks a random public/private key pair $(pk, sk)$ by running $\mathbf{Set}(1^k)$, sets $\mathsf{PK} = pk$, $\mathsf{SK} = sk$, $\mathsf{TPK} = D_{\mathcal{R}}$, and gives $(\mathsf{PK}, \mathsf{TPK})$ to $B$. $Inv$ manages a list $L = \{(m_i, t_i, s_i, c_i, d_i)\}$ to answer $B$'s queries to $O_{\mathsf{PreH}}$. Let $q_{\mathsf{TSig}}$ be the total number of $O_{\mathsf{TSig}}$ queries made by $B$ and $r$ be a random number chosen by $Inv$ in the interval of $\{1, 2, \cdots, q_{\mathsf{TSig}}\}$. Now, $Inv$ knowing $\mathsf{SK} = sk$ responds to the $i$-th $O_{\mathsf{TSig}}$ query $(m_i, t_i)$ of $B$ as follows;

- If $i = r$, $Inv$ outputs $t_r$ to the challenger $\mathcal{C}$ and receives a random lock $c \in R_{t_r}$ from the challenger. $Inv$ sets $c_r = c$ and computes $s_r = \mathbf{Sig}_{sk}(m_r||c_r||t_r)$. $Inv$ returns $\sigma'_{t_r} = (s_r, c_r)$ to $B$ and stores the element $(m_r, t_r, s_r, c_r, \perp)$ in the list $L$.
- If $i \neq r$, $Inv$ picks a random lock/proof pair $(c_i, d_i)$ from $\mathsf{Sample}_{D_{\mathcal{R}}}(t_i)$ and computes $s_i = \mathbf{Sig}_{sk}(m_i||c_i||t_i)$. $Inv$ returns $\sigma'_{t_i} = (s_i, c_i)$ to $B$ and stores the element $(m_i, t_i, s_i, c_i, d_i)$ in the list $L$.

To simulate $O_{\mathsf{TR}}$ to the query $t_i$ of $B$, $Inv$ simply asks $t_i$ to its own trapdoor extraction oracle $O_{\mathsf{Extract}}$ and gets $\mathsf{td}_{R_{t_i}}$. If $t_i = t_r$, $Inv$ abort. Otherwise, $Inv$ returns $Z_{t_i} = \mathsf{td}_{R_{t_i}}$ to $B$.

To simulate $O_{\mathsf{PreH}}$ to the query $(m_i, t_i, s_i, c_i)$, $Inv$ checks whether the query is in the list $L$ or not (by considering only the first four components of an element in $L$). If $(m_i, t_i, s_i, c_i)$ is in the list $L$ and equal to $(m_r, t_r, s_r, c_r)$, $Inv$ aborts. If $(m_i, t_i, s_i, c_i)$ is in the list $L$ and not equal to $(m_r, t_r, s_r, c_r)$, $Inv$ obtains a proof $d_i$ from the list $L$ and give a pre-hatched signature $\sigma_{t_i} = (s_i, c_i, d_i)$ to

$B$. If $(m_i, t_i, s_i, c_i)$ is not in the list $L$ (i.e., the time capsule signature was not generated by $Inv$ and therefore the query is invalid with very high probability), $Inv$ answers randomly to $B$.

The probability that $Inv$ does not abort during the simulation is at least $1/q_{\mathsf{TSig}}$, since $r \in \{1, \cdots, q_{\mathsf{TSig}}\}$ is randomly chosen and a secure ID-THIR satisfies the ambiguity property. When $B$ outputs the forgery $(m_B, t_B, s_B, c_B, d_B)$, $Inv$ verifies that the forgery passes the verification algorithm $\mathsf{Ver}$ and $(m_B, t_B, s_B, c_B) = (m_r, t_r, s_r, c_r)$. If so, $Inv$ outputs the proof $d_B$. Otherwise, $Inv$ chooses a proof $d_{Inv}$ randomly and outputs $d_{Inv}$. Therefore, if $B$ forges with a probability $\epsilon$, $Inv$ succeeds in breaking the one-wayness of ID-THIR with a probability $\epsilon' \geq \epsilon/q_{\mathsf{TSig}}$.

Security against Time Server. To show security against Time Server, we convert any attacker $C$ that attacks our time capsule signature scheme into a forger $F$ for the underlying ordinary signature. Recall that $F$ gets $pk$ as an input, and has access to the signing oracle $O_{\mathbf{S\ g}}$. On the other hand, $C$ expects $(\mathsf{PK}, \mathsf{TPK}, \mathsf{TSK})$ as input and has access to $O_{\mathsf{TSig}}$ and $O_{\mathsf{PreH}}$. $C$ wins if it forges a hatched (or pre-hatched) signature $\sigma_t$ of some message $m$ without obtaining a time capsule signature on $m$ from $O_{\mathsf{TSig}}$.

So here is how $F$ simulates the run of $C$. To choose ID-THIR, $F$ runs $\mathsf{Gen}(1^k)$ and obtains $(D_{\mathcal{R}}, \mathsf{mtd}_{\mathcal{R}})$. Then, $F$ gives $(\mathsf{PK}, \mathsf{TPK}, \mathsf{TSK}) = (pk, D_{\mathcal{R}}, \mathsf{mtd}_{\mathcal{R}})$ to $C$. $F$ can respond to $O_{\mathsf{TSig}}$ queries $(m_i, t_i)$ of $C$ by choosing a random lock/proof pair $(c_i, d_i)$ from $\mathsf{Sample}_{D_{\mathcal{R}}}(t_i)$ and getting an ordinary signature $s_i$ on $m_i||c_i||t_i$ from its own signing oracle $O_{\mathbf{S\ g}}$. $F$ stores $(m_i, c_i, d_i, t_i)$ in the list $L = \{(m_i, c_i, d_i, t_i)\}$ to answer $C$'s queries to $O_{\mathsf{PreH}}$. To simulate $O_{\mathsf{PreH}}$ to the queries $(m_i, t_i, s_i, c_i)$, $F$ verifies that $s_i$ is a valid signature on $m_i||c_i||t_i$.

- If $s_i$ is a valid signature on $m_i||c_i||t_i$, $F$ checks whether $(m_i, c_i, t_i)$ is in the list $L$ or not. If it is in the list, $F$ can give the corresponding proof $d_i$ from the list $L$. Otherwise, $s_i$ is a new signature value and $F$ succeeds in producing a new forgery $s_i$ on $m_i||c_i||t_i$. $F$ stops the simulation.
- If $s_i$ is not a valid signature on $m_i||c_i||t_i$, $F$ answers randomly.

When $C$ outputs the forgery $(\widehat{m}, \widehat{t}, \widehat{\sigma_t})$ where $\widehat{\sigma_t} = (\widehat{s}, \widehat{c}, \widehat{d})$, $F$ outputs an ordinary signature $\widehat{s}$ on a message $\widehat{m}||\widehat{c}||\widehat{t}$. Therefore, if $C$ succeeds with a probability $\epsilon$, $F$ succeeds in producing a new forgery with a probability $\epsilon' \geq \epsilon$. □

**Theorem 3.** *If there is a one-way function, there exists a secure time capsule signature scheme in the random oracle model.*

**Proof:** Secure signatures exist if and only if one-way functions exist [29]. Together with Theorem 1 and Theorem 2, we obtain Theorem 3. □

**Theorem 4.** *If there is a trapdoor one-way permutation, there exists a secure time capsule signature scheme in the standard model.*

**Proof:** Secure signatures exist if and only if one-way functions exist [29]. Together with Corollary 1 and Theorem 2, we obtain Theorem 4. □

*Remark 4.* (Event Capsule Signature) In the definition and construction of time capsule signature, we did not use any characteristic of the real time. Actually, $t$ need not be a time value and any index works for $t$. Therefore, the definition and construction of time capsule signature can be efficiently converted to those of event capsule signature.

## 4 On Trapdoor Hard-to-Invert Relation

A trapdoor hard-to-invert relation (THIR) is a specific elementary relation $R_{id}$ in ID-THIR $\mathcal{R} = \{R_{id} \mid id \in I_R\}$. The definition of THIR can be derived from that of ID-THIR and the construction becomes even simpler as a signature on one identity is simply a one-way function. Notice that THIR is also very easily constructed without the random oracle model (unlike ID-THIR) if trapdoor one-way permutations exist (for details, refer to the full version).

However, it is interesting to ask whether THIR (primitive simpler than ID-THIR) can be built from one-way functions (or even one-way permutations) in the standard model. We leave this as an open problem. However, we comment that it is highly unlikely that a special case of THIR — so called deterministic THIR where only one proof $d$ exists for a given lock $c$ — can be constructed from one-way permutations. Indeed, one can easily see that the existence of a deterministic THIR implies that of a secure key agreement scheme. However, it is known that there exists no black-box reduction from one-way permutations to secure key agreement schemes [23].

## Acknowledgments

We would like to thank Leonid Reyzin for several very insightful discussions. The work of the second author was partially supported by Brain Korea 21 project.

## References

1. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. In *EUROCRYPT 1998*, LNCS 1403, pp. 591–606, Springer, 1998.
2. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. *IEEE J. Select. Areas Commun*, 18(4), pp. 593–610, 2000.
3. M. Bellare and S. Goldwasser. Encapsulated key escrow. MIT Laborator for Computer Science Technical Report 688, 1996.
4. M. Bellare and S. Goldwasser. Verifiable partial key escrow. In *the 4th ACM CCS*, pp. 78–91, 1997.
5. M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *the 1st ACM CCS*, pp. 62–73, 1993.
6. I. Blake and A. Chan. Scalable, server-passive, user-anonymous timed release public key encryption from bilinear pairing. `http://eprint.iacr.org/2004/211/`.
7. D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. In *CRYPTO 2001*, LNCS 2139, pp. 213–229, Springer, 2001.

8. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *EUROCRYPT 2003*, pp. 416–432.
9. D. Boneh, B. Lynn and H. Shacham. Short signatures from the Weil pairing. In *ASIACRYPT 2001*, LNCS 2248, pp. 514–532, Springer, 2001.
10. D. Boneh and M. Naor. Timed commitments. In *CRYPTO 2000*, pp. 236–254.
11. J. Camenisch and A. Lysyanskaya. Signature schemes with efficient protocols. In *SCN 2002*, LNCS 2576, pp. 268–289, Springer, 2002.
12. J. Camenisch and A. Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *CRYPTO 2004*, LNCS 3152, pp. 56–72, Springer, 2004.
13. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO 1994*, pp. 174–187.
14. Y. Dodis and L. Reyzin. Breaking and repairing optimistic fair exchange from PODC 2003. In *Digital Rights Management 2003*, pp. 47–54, 2003.
15. C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In *CRYPTO 1992*, LNCS 740, pp. 139–147, Springer, 2004.
16. U. Feige and A. Shamir. Zero knowledge proofs of knowledge in two rounds. In *CRYPTO 1989*, LNCS 435, pp. 526–544, Springer, 1989.
17. U. Feige and A. Shamir. Witness indistinguishable and witness hiding protocols. In *the 22nd Annual ACM Symposium on Theory of Computing*, pp. 416–426, 1990.
18. A. Fiat and A. Shamir. How to prove yourself: practical solutions to identification and signature problems. In *CRYPTO 1986*, LNCS 263, pp. 186–194, Springer, 1986.
19. J. Garay and M. Jakobsson. Timed release of standard digital signatures. In *Financial Cryptography 2002*, LNCS 2357, pp. 168–182, Springer, 2002.
20. J. Garay and C. Pomerance. Timed fair exchange of standard signatures. In *Financial Cryptography 2003*, LNCS 2742, pp. 190–207, Springer, 2003.
21. O. Goldreich, S. Micali and A.Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM,* 38(3), pp. 691–729, 1991.
22. L. Guillou and J.J. Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In *CRYPTO 1988*, pp. 216–231, Springer, 1988.
23. R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *the 21st STOC*, pp. 44–61, 1989.
24. A. Joux and K. Nguyen. Separating decision Diffie-Hellman from Diffie-Hellman in cryptographic groups. `http://eprint.iacr.org/2001/003/`.
25. I. Osipkov, Y. Kim and J. Cheon. New approaches to timed-release cryptography. `http://eprint.iacr.org/2004/231/`.
26. R. Pass. On deniability in the common reference string and random oracle model. In *CRYPTO 2003*, LNCS 2729, pp. 316–337, Springer, 2003.
27. D. Pointcheval and J. Stern. Security proofs for signature schemes. In *EUROCRYPT 1996*, LNCS 1070, pp. 387–398, Springer, 1996.
28. R. Rivest, A. Shamir, and D. Wagner. Time lock puzzles and timed release cryptography. Technical report, MIT/LCS/TR-684.
29. J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *the 22nd Annual ACM Symposium on Theory of Computing*, pp. 387–394, 1990.
30. A. De Santis and G. Persiano. Zero-knowledge proofs of knowledge without interaction. In *the 33rd FOCS*, pp. 427–436, 1992.
31. C. Schnorr. Efficient identification and signatures for smart cards. In *CRYPTO 1989*, LNCS 435, pp. 239–252, Springer, 1989.
32. A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO 1984*, LNCS 196, pp. 47–53, Springer, 1984.

# Policy-Based Cryptography and Applications[⋆]

Walid Bagga and Refik Molva

Institut Eurécom, Corporate Communications,
2229, route des Crêtes B.P. 193, 06904 Sophia Antipolis, France
{bagga, molva}@eurecom.fr

**Abstract.** In this paper, we introduce the concept of *policy-based cryptography* which makes it possible to perform policy enforcement in large-scale open environments like the Internet, while respecting the data minimization principle according to which only strictly necessary information should be collected for a given purpose. We propose concrete policy-based encryption and signature schemes, based on bilinear pairings, which allow performing relatively efficient encryption and signature operations with respect to credential-based policies formalized as boolean expressions written in generic conjunctive-disjunctive normal form. We illustrate the privacy properties of our policy-based cryptographic schemes through the description of three application scenarios.

**Keywords:** Policy, Authorization, Credentials, Privacy, ID-based Cryptography.

## 1 Introduction

In open computing environments like the Internet, many interactions may occur between entities from different security domains without pre-existing trust relationships. Such interactions may require the exchange of sensitive resources which need to be carefully protected through clear and concise policies. A policy specifies the constraints under which a specific action can be performed on a certain sensitive resource. An increasingly popular approach for authorization in distributed systems consists in defining conditions which are fulfilled by digital credentials. A digital credential is basically a digitally signed assertion by a trusted authority (credential issuer) about a specific user (credential owner). It describes one or multiple properties of the user that are validated by the trusted authority. It is generated using the trusted authority's private key and can be verified using its public key.

Consider the following scenario: a user named Bob controls a sensitive resource denoted 'res', and for a specific action denoted 'act' he defines a policy denoted 'pol' which specifies the conditions under which 'act' may be performed on 'res'. Policy 'pol' is fulfilled by a set of credentials generated by one or multiple trusted authorities. In order for a user named Alice to be authorized to perform 'act' on 'res', she has to prove her compliance to Bob's policy i.e. she has to prove that she possesses a minimal

[⋆] The work reported in this paper is supported by the IST PRIME project and by Institut Eurécom; however, it represents the view of the authors only.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 72–87, 2005.
© Springer-Verlag Berlin Heidelberg 2005

set of credentials that is required by 'pol' to permit action 'act' on 'res'. In standard credentials systems like $X.509$, Alice needs first to request the credentials from the appropriate trusted authorities. Then, Alice has to show her credentials to Bob who verifies their validity using the public keys of the issuing trusted authorities. Bob authorizes Alice to perform 'act' on 'res' if and only if he receives a set of valid credentials satisfying 'pol'. Such scenario does not meet the *data minimization* requirement (called the *data quality principle* in OECD guidelines [8]) according to which only strictly necessary information should be collected for a given purpose. In fact, the standard approach allows Bob, on one hand, to enforce his policy i.e. to get a proof that Alice is compliant to his policy before authorizing her to perform the requested action on the specified sensitive resource. On the other hand, it allows him to collect additional 'out-of-purpose' information on Alice's specific credentials.

In this paper, we formulate the concept of *policy-based cryptography* which allows to perform policy enforcement while respecting the data minimization principle. Such 'privacy-aware' policy enforcement is enabled by two cryptographic primitives: *policy-based encryption* and *policy-based signature*. Intuitively, policy-based encryption allows to encrypt data according to a policy so that only entities fulfilling the policy are able to successfully perform the decryption and retrieve the plaintext data, whereas policy-based signature allows to generate a digital signature on data with respect to a policy so that only entities satisfying the policy are able to generate a valid signature.

Our cryptography-based policy enforcement mechanisms manipulate policies that are formalized as monotonic logical expressions involving complex disjunctions and conjunctions of conditions. Each condition is fulfilled by a specific credential issued by a certain trusted authority. Such policy model allows multiple trusted authorities to participate to the authorization process which makes it, on one hand, more realistic because each authority should be responsible for a specific, autonomous and limited administrative domain, and on the other hand, more trustworthy compared with models relying on a centralized trusted authority (which could be seen as a single point of failure) to issue the required credentials. Furthermore, in contrast to the traditional approach where credentials are revealed during policy compliance proofs, our credentials have to be kept secret by their owners. They are used to perform policy-based decryption and policy-based signature operations. We note that the idea of using secret credentials as decryption keys has already been used or at least mentioned in the literature, especially in the contexts of access control and trust negotiation systems [3, 7, 15, 12, 9].

We use existing cryptographic primitives from bilinear pairings on elliptic curves to construct concrete policy-based cryptographic schemes. In fact, our credentials system is based on the short signature scheme defined in [4], our policy-based encryption scheme extends the ID-based encryption scheme described in [3] and our policy-based signature scheme extends the ID-based ring signatures given in [13, 18]. Our algorithms offer a more elegant and efficient way to handle complex authorization structures than the widely used naive approach based on onion-like encryptions to deal with conjunctions (AND) and multiple encryptions to deal with disjunctions (OR). Apart from performance considerations, our policy-based cryptographic primitives have many interesting applications in different critical contexts in today's Internet such as access control, sticky privacy policies, trust establishment, and automated trust negotiation.

The sequel of the paper is organized as follows: we provide in Section 2 a formal model for policy-based cryptography. Moreover, we give formal definitions for policy-based encryption and signature schemes. In Section 3, we describe our concrete policy-based encryption and signature schemes. We briefly discuss their efficiency in Section 4 and analyze their security properties in Section 5. In Section 6, we illustrate the privacy properties of our policy-based primitives. In Section 7, we discuss related work before concluding in Section 8.

## 2 Model

In this section, we formulate the concept of policy-based cryptography. We first describe the policy-based cryptosystem setup procedure. We then describe the policy model and define the related terminology. We finally provide formal definitions for policy-based encryption and policy-based signature.

### 2.1 System Setup

A policy-based cryptosystem setup procedure is specified by two randomized algorithms PBC-Setup and TA-Setup which we describe below.

**PBC-Setup.** On input of a security parameter $k$, this algorithm generates a set of public parameters, denoted $\mathcal{P}$, which specifies the different groups and public functions that will be used by the system procedures and participants. Furthermore, it includes a description of a message space denoted $\mathcal{M}$, a ciphertext space denoted $\mathcal{C}$, and a signature space denoted $\mathcal{S}$. We assume that the set of parameters $\mathcal{P}$ is publicly known so that we do not need to explicitly provide it as input to subsequent policy-based procedures.

**TA-Setup.** Each trusted authority $TA$ uses this algorithm to generate a secret master-key $s$ and a corresponding public key $R$. We assume that a set of trusted authorities denoted $\mathcal{T}$ is publicly known and thus can be referenced by all the system participants i.e. a trustworthy value of the public key of each trusted authority included in $\mathcal{T}$ is known by the system participants. At any time, a new trusted authority may be added to $\mathcal{T}$.

### 2.2 Policy Model

In the context of this paper, we define an assertion to be a declaration about a subject, where a subject is an entity (either human or computer) that has an identifier in some security domain. An assertion can convey information about the subject's attributes, properties, capabilities, etc. The representation of assertions being out of the scope of this paper, they will be simply encoded as binary strings. We define a credential to be an assertion which validity is certified by a trusted authority through a signature procedure. A trusted authority is basically 'trusted' for not issuing credentials corresponding to invalid assertions. Whenever a trusted authority $TA \in \mathcal{T}$ is asked to sign an assertion $A \in \{0,1\}^*$, it first checks the validity of $A$. If $A$ is valid, then $TA$ executes algorithm CredGen defined below and returns the output back to the credential requester. Otherwise, $TA$ returns an error message.

**CredGen.** On input of assertion $A$ and $TA$'s master-key $s$, this algorithm outputs a credential denoted $\varsigma(R,A)$ where $R$ denotes $TA$'s public key. For every pair $(TA,A)$, the credential $\varsigma(R,A)$ can be generated only by the trusted authority $TA$ using its secret master-key $s$, while its validity can be checked using its public key $R$.

We define a policy to be a monotonic logical expression involving conjunctions ($\wedge$) and disjunctions ($\vee$) of 'atomic' conditions. Each condition is defined through a pair $\langle TA,A\rangle$ which specifies an assertion $A$ and indicates the authority $TA$ that is trusted to check and certify $A$'s validity. Let the expression 'user $\leftharpoonup \varsigma(R,A)$' denote the fact that 'user' has been issued credential $\varsigma(R,A)$ and let the expression 'user $\rightleftharpoons \langle TA,A\rangle$' denote the fact that 'user' fulfills condition $\langle TA,A\rangle$. Then, we state the following property

$$\text{user} \rightleftharpoons \langle TA,A\rangle \;\Leftrightarrow\; \text{user} \leftharpoonup \varsigma(R,A)$$

As every statement in logic consisting of a combination of multiple $\wedge$ (AND) and $\vee$ (OR), a policy can be written in either conjunctive normal form (CNF) or in disjunctive normal form (DNF). In order to address these two normal forms, a policy denoted 'pol' will be written in conjunctive-disjunctive normal form (CDNF) (as defined in [15])

$$\text{pol} = \wedge_{i=1}^{m}[\vee_{j=1}^{m_i}[\wedge_{k=1}^{m_{i,j}}\langle TA_{i,j,k},A_{i,j,k}\rangle]]$$

Thus, policies expressed in CNF form are such that $m_{i,j}=1$ for all $i,j$, while policies expressed in DNF form are such that $m=1$.

Given $j_i \in \{1,\ldots,m_i\}$ for all $i \in \{1,\ldots,m\}$, we define $\varsigma_{j_1,\ldots,j_m}(\text{pol})$ to be the set of credentials $\{\{\varsigma(R_{i,j_i,k},A_{i,j_i,k})\}_{1\le k\le m_{i,j_i}}\}_{1\le i\le m}$. Let the expression 'user $\leftharpoonup \varsigma_{j_1,\ldots,j_m}(\text{pol})$' denote the fact that 'user' has been issued all the credentials included in $\varsigma_{j_1,\ldots,j_m}(\text{pol})$ i.e.

$$\forall\, i \in \{1,\ldots,m\}, \forall\, k \in \{1,\ldots,m_{i,j_i}\}, \text{user} \leftharpoonup \varsigma(R_{i,j_i,k},A_{i,j_i,k})$$

Let the expression 'user $\rightleftharpoons$ pol', for $\text{pol} = \wedge_{i=1}^{m}[\vee_{j=1}^{m_i}[\wedge_{k=1}^{m_{i,j}}\langle TA_{i,j,k},A_{i,j,k}\rangle]]$, denote the fact that 'user' fulfills (satisfies) policy 'pol'. Property (1) leads to the following

$$\text{user} \rightleftharpoons \text{pol} \;\Leftrightarrow\; \forall\, i \in \{1,\ldots,m\}, \exists\, j_i \in \{1,\ldots,m_i\} : \text{user} \leftharpoonup \varsigma_{j_1,\ldots,j_m}(\text{pol})$$

Informally, we may say that the set of credentials $\varsigma_{j_1,\ldots,j_m}(\text{pol})$ fulfills policy 'pol'.

### 2.3 Policy-Based Encryption

A policy-based encryption scheme (denoted PBE) consists of two randomized algorithms: PolEnc and PolDec which we describe below.

**PolEnc.** On input of message $m$ and policy $\text{pol}_{\text{A}}$, this algorithm returns a ciphertext $c$ which represents the message $m$ encrypted according to policy $\text{pol}_{\text{A}}$.

**PolDec.** On input of ciphertext $c$, policy $\text{pol}_{\text{A}}$ and a set of credentials $\varsigma_{j_1,\ldots,j_a}(\text{pol}_{\text{A}})$, this algorithm returns a message $m$.

Algorithms PolEnc and PolDec have to satisfy the standard consistency constraint i.e.

$$c = \mathsf{PolEnc}(m,\text{pol}_{\text{A}}) \;\Rightarrow\; \mathsf{PolDec}(c,\text{pol}_{\text{A}},\varsigma_{j_1,\ldots,j_a}(\text{pol}_{\text{A}})) = m$$

### 2.4 Policy-Based Signature

A policy-based signature scheme (denoted PBS) consists of two randomized algorithms: PolSig and PolVrf which we describe below.

**PolSig.** On input of message $m$, policy $\text{pol}_\text{B}$ and a set of credentials $\varsigma_{j_1,\dots,j_b}(\text{pol}_\text{B})$, this algorithm returns a signature $\sigma$ which represents the signature on message $m$ according to policy $\text{pol}_\text{B}$.

**PolVrf.** On input of message $m$, policy $\text{pol}_\text{B}$ and signature $\sigma$, this algorithm returns $\top$ (for 'true') if $\sigma$ is a valid signature on $m$ according to policy $\text{pol}_\text{B}$. Otherwise, it returns $\perp$ (for 'false').

Algorithms PolSig and PolVrf have to satisfy the standard consistency constraint i.e.

$$\sigma = \mathsf{PolSig}(m, \text{pol}_\text{B}, \varsigma_{j_1,\dots,j_b}(\text{pol}_\text{B})) \;\Rightarrow\; \mathsf{PolVrf}(m, \text{pol}_\text{B}, \sigma) = \top$$

## 3 Policy-Based Cryptography from Bilinear Pairings

In this section, we describe concrete policy-based encryption and signature schemes based on bilinear pairings over elliptic curves.

### 3.1 System Setup

We define algorithm BDH-Setup to be a bilinear Diffie-Hellman parameter generator satisfying the BDH assumption as this has been formally defined in [3]. Thus, on input of a security parameter $k$, algorithm BDH-Setup generates a tuple $(q, \mathbb{G}_1, \mathbb{G}_2, e)$ where the map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is a bilinear pairing, $(\mathbb{G}_1, +)$ and $(\mathbb{G}_2, *)$ are two groups of the same order $q$, where $q$ is determined by the security parameter $k$. We recall that a bilinear pairing satisfies the following three properties:

1. Bilinear: for $Q, Q' \in \mathbb{G}_1$ and for $a, b \in \mathbb{Z}_q^*$, $e(a \cdot Q, b \cdot Q') = e(Q, Q')^{ab}$
2. Non-degenerate: $e(P, P) \neq 1$ and therefore it is a generator of $\mathbb{G}_2$
3. Computable: there exists an efficient algorithm to compute $e(Q, Q')$ for all $Q$, $Q' \in \mathbb{G}_1$

The tuple $(q, \mathbb{G}_1, \mathbb{G}_2, e)$ is such that the mathematical problems defined below are such that there is no polynomial time algorithms to solve them with non-negligible probability.

- Discrete Logarithm Problem (DLP). Given $Q, Q' \in \mathbb{G}_1$ such that $Q' = x \cdot Q$ for some $x \in \mathbb{Z}_q^*$: find $x$
- Bilinear Pairing Inversion Problem (BPIP). Given $Q \in \mathbb{G}_1$ and $e(Q, Q')$ for some $Q' \in \mathbb{G}_1$: find $Q'$
- Bilinear Diffie-Hellman Problem (BDHP). Given $(P, a \cdot P, b \cdot P, c \cdot P)$ for $a, b, c \in \mathbb{Z}_q^*$: compute $e(P, P)^{abc}$

The hardness of the problems defined above can be ensured by choosing groups on supersingular elliptic curves or hyperelliptic curves over finite fields and deriving the bilinear pairings from Weil or Tate pairings [10]. As we merely apply these mathematical primitives in this paper, we refer to [17] for further details.

Our PBC-Setup, TA-Setup and CredGen algorithms are described below.

**PBC-Setup.** Given a security parameter $k$, do the following:

1. Run algorithm BDH-Setup on input $k$ to generate output $(q, \mathbb{G}_1, \mathbb{G}_2, e)$
2. Pick at random a generator $P \in \mathbb{G}_1$
3. For some chosen $n \in \mathbb{N}^*$, let $\mathcal{M} = \{0,1\}^n$
4. Let $\mathcal{C} = \mathbb{G}_1 \times (\{0,1\}^n)^* \times \mathcal{M}$ and $\mathcal{S} = (\mathbb{G}_2)^* \times \mathbb{G}_1$
5. Define five hash functions: $H_0 : \{0,1\}^* \to \mathbb{G}_1$, $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$, $H_2 : \{0,1\}^* \to \{0,1\}^n$, $H_3 : \{0,1\}^n \to \{0,1\}^n$ and $H_4 : \{0,1\}^* \to \mathbb{Z}_q^*$
6. Set the system public parameters to be $\mathcal{P} = (q, \mathbb{G}_1, \mathbb{G}_2, e, n, P, H_0, H_1, H_2, H_3, H_4)$

**TA-Setup.** Each trusted authority $TA$ picks at random a master-key $s \in \mathbb{Z}_q^*$ and keeps it secret while publishing the corresponding public key $R = s \cdot P$.

**CredGen.** Given a valid assertion $A$ and $TA$'s master-key $s$, this algorithm outputs the credential $\varsigma(R,A) = s \cdot H_0(A)$.

### 3.2 Policy-Based Encryption

Our policy-based encryption scheme can be seen as a kind of extension or generalization of the Boneh-Franklin ID-based encryption scheme given in [3]. Let $\text{pol}_\text{A}$ denote a policy of the form $\wedge_{i=1}^{a} [\vee_{j=1}^{a_i} [\wedge_{k=1}^{a_{i,j}} \langle TA_{i,j,k}, A_{i,j,k} \rangle]]$, we describe our PolEnc algorithm below.

**PolEnc.** Given message $m$ and policy $\text{pol}_\text{A}$, do the following:

1. Pick randomly $t_i \in \{0,1\}^n$ for $i = 1, \ldots, a$
2. Compute $t = \oplus_{i=1}^{a} t_i$, then compute $r = H_1(m \| t \| \text{pol}_\text{A})$ and $U = r \cdot P$
3. For $i = 1, \ldots, a$, for $j = 1, \ldots, a_i$,
   (a) Compute $g_{i,j} = \prod_{k=1}^{a_{i,j}} e(R_{i,j,k}, H_0(A_{i,j,k}))$
   (b) Compute $v_{i,j} = t_i \oplus H_2(g_{i,j}^r \| i \| j)$
4. Compute $w = m \oplus H_3(t)$
5. Set the ciphertext to be $c = (U, [v_{i,1}, v_{i,2}, \ldots, v_{i,a_i}]_{1 \le i \le a}, w)$

The intuition behind the encryption procedure described above is as follows: each conjunction of conditions $\wedge_{i,j} = \wedge_{k=1}^{a_{i,j}} \langle TA_{i,j,k}, A_{i,j,k} \rangle$ is associated to a kind of mask we denote $\mu_{i,j} = H_2(g_{i,j}^r \| i \| j)$. For each index $i$, a randomly chosen key $t_i$ is associated to the disjunction $\vee_i = \vee_{j=1}^{a_i} \wedge_{i,j}$. Each $t_i$ is encrypted $a_i$ times using each of the masks $\mu_{i,j}$. Thus, it is sufficient to compute any one of the masks $\mu_{i,j}$ in order to be able to retrieve the key $t_i$. In order to be able to perform the decryption procedure successfully, an entity needs to retrieve all the keys $t_i$. Our PolDec algorithm is described below.

**PolDec.** Given the ciphertext $c = (U, [v_{i,1}, v_{i,2}, \ldots, v_{i,a_i}]_{1 \le i \le a}, w)$, policy $\text{pol}_\text{A}$ and the set of credentials $\varsigma_{j_1,\ldots,j_a}(\text{pol}_\text{A})$, do the following:

1. For $i = 1, \ldots, a$,
   (a) Compute $\tilde{g}_{i,j_i} = e(U, \sum_{k=1}^{a_{i,j_i}} \varsigma(R_{i,j_i,k}, A_{i,j_i,k}))$
   (b) Compute $\tilde{t}_i = v_{i,j_i} \oplus H_2(\tilde{g}_{i,j_i} \| i \| j_i)$

2. Compute $\tilde{m} = w \oplus H_3(\oplus_{i=1}^{a} \tilde{t}_i)$
3. Compute $\tilde{U} = H_1(\tilde{m} \| \oplus_{i=1}^{a} \tilde{t}_i \| \mathrm{pol}_{\mathrm{A}}) \cdot P$
4. If $\tilde{U} = U$, then return message $\tilde{m}$, otherwise return $\perp$ (for ‘error’)

Our algorithms PolEnc and PolDec satisfy the standard consistency constraint. In fact, thanks to the properties of bilinear pairings, it is easy to check that for every index $i$, $\tilde{g}_{i,j_i} = g_{i,j_i}^{r}$.

### 3.3 Policy-Based Signature

Our policy-based signature scheme is a kind of extension of the ID-based ring signature schemes given in [18, 13]. In an ID-based ring signature, the signer sets up a finite set of identities including his identity. The set of identities represents the set of all possible signers i.e. ring members. A valid signature will convince the verifier that the signature is generated by one of the ring members, without revealing any information about which member has actually generated the signature. Let $\mathrm{pol}_{\mathrm{B}}$ denote a policy of the form $\wedge_{i=1}^{b}[\vee_{j=1}^{b_i}[\wedge_{k=1}^{b_{i,j}} \langle TA_{i,j,k}, A_{i,j,k} \rangle]]$, we describe our PolSig algorithm below.

**PolSig.** Given message $m$, policy $\mathrm{pol}_{\mathrm{B}}$ and the set of credentials $\varsigma_{j_1,\ldots,j_b}(\mathrm{pol}_{\mathrm{B}})$, do the following:

1. For $i = 1, \ldots, b$,
   (a) Pick randomly $Y_i \in \mathbb{G}_1$, then compute $x_{i,j_i+1} = e(P, Y_i)$
   (b) For $l = j_i + 1, \ldots, b_i, 1, \ldots, j_i - 1 \bmod (b_i + 1)$,
      i. Compute $\tau_{i,l} = \prod_{k=1}^{b_{i,l}} e(R_{i,l,k}, H_0(A_{i,l,k}))$
      ii. Pick randomly $Y_{i,l} \in \mathbb{G}_1$, then compute $x_{i,l+1} = e(P, Y_{i,l}) * \tau_{i,l}^{H_4(m \| x_{i,l} \| \mathrm{pol}_{\mathrm{B}})}$
   (c) Compute $Y_{i,j_i} = Y_i - H_4(m \| x_{i,j_i} \| \mathrm{pol}_{\mathrm{B}}) \cdot (\sum_{k=1}^{b_{i,j_i}} \varsigma(R_{i,j_i,k}, A_{i,j_i,k}))$
2. Compute $Y = \sum_{i=1}^{b} \sum_{j=1}^{b_i} Y_{i,j}$
3. Set the signature to be $\sigma = ([x_{i,1}, x_{i,2}, \ldots, x_{i,b_i}]_{1 \le i \le b}, Y)$

The intuition behind the signature procedure described above is as follows: each conjunction of conditions $\wedge_{i,j} = \wedge_{k=1}^{b_{i,j}} \langle TA_{i,j,k}, A_{i,j,k} \rangle$ is associated to a tag $\tau_{i,j}$. For each index $i$, the set of tags $\{\tau_{i,j}\}_j$ corresponds to a set of ring members. The signature key associated to the tag $\tau_{i,j}$ corresponds to the set of credentials $\{\varsigma(R_{i,j,k}, A_{i,j,k})\}_{1 \le k \le b_{i,j}}$. Our PolVrf algorithm is described below.

**PolVrf.** Given message $m$, policy $\mathrm{pol}_{\mathrm{B}}$ and the signature $\sigma = ([x_{i,1}, x_{i,2}, \ldots, x_{i,b_i}]_{1 \le i \le b}, Y)$, do the following:

1. Compute $z_1 = \prod_{i=1}^{b} [\prod_{j=1}^{b_i} x_{i,j}]$
2. For $i = 1, \ldots, b$ and for $j = 1, \ldots, b_i$, compute $\tau_{i,j} = \prod_{k=1}^{b_{i,j}} e(R_{i,j,k}, H_0(A_{i,j,k}))$
3. Compute $z_2 = e(P, Y) * \prod_{i=1}^{b} [\prod_{j=1}^{b_i} \tau_{i,j}^{H_4(m \| x_{i,j} \| \mathrm{pol}_{\mathrm{B}})}]$
4. If $z_1 = z_2$, then return $\top$, otherwise return $\perp$

Our algorithms PolSig and PolVrf satisfy the standard consistency constraint. In fact, it is easy to check that for $i = 1, \ldots, b$ and $j = 1, \ldots, b_i$, the following holds

$$\tau_{i,j}^{H_4(m\|x_{i,j}\|\mathrm{pol}_\mathrm{B})} = x_{i,j+1} * e(P,Y_{i,j})^{-1} \text{ (where } x_{i,b_i+1} = x_{i,1})$$

Let $\lambda = e(P,Y)$, then the following holds

$$z_2 = \lambda * \prod_{i=1}^{b}[\prod_{j=1}^{b_i} \tau_{i,j}^{H_4(m\|x_{i,j}\|\mathrm{pol}_\mathrm{B})}] = \lambda * \prod_{i=1}^{b}[\prod_{j=1}^{b_i-1} x_{i,j+1} * e(P,Y_{i,j})^{-1} * x_{i,1} * e(P,Y_{i,b_i})^{-1}]$$

$$= \lambda * \prod_{i=1}^{b}[\prod_{j=1}^{b_i} x_{i,j} * \prod_{j=1}^{b_i} e(P,Y_{i,j})^{-1}] = \lambda * [\prod_{i=1}^{b}\prod_{j=1}^{b_i} x_{i,j}] * [e(P,\sum_{i=1}^{n}\sum_{j=1}^{b_i} Y_{i,j})]^{-1} = \lambda * z_1 * \lambda^{-1}$$

## 4 Efficiency

The essential operation in pairings-based cryptography is pairing computation. Although such operation can be optimized as explained in [1], it still have to be minimized. Table 4 summarizes the computational costs of our policy-based encryption and signature schemes in terms of pairing computations.

**Table 1.** Computational costs in terms of pairing computations

| PolEnc | PolDec | PolSig | PolVrf |
|---|---|---|---|
| $\sum_{i=1}^{a}\sum_{j=1}^{a_i} a_{i,j}$ | $a$ | $\sum_{i=1}^{b} b_i + \sum_{i=1}^{b}\sum_{j\neq j_i} b_{i,j}$ | $1 + \sum_{i=1}^{b}\sum_{j=1}^{b_i} b_{i,j}$ |

Notice that for all $i, j, k$, the pairing $e(R_{i,j,k}, H_0(A_{i,j,k}))$ involved in algorithms PolSig, PolEnc and PolVrf does not depend on the message $m$. Thus, it can be pre-computed, cached and used in subsequent signatures, encryptions and verifications involving the condition $\langle TA_{i,j,k}, A_{i,j,k}\rangle$.

Let $l_i$ be the bit-length of the bilinear representation of an element of group $\mathbb{G}_i$ for $i = 1,2$. Then, the bit-length of a ciphertext produced by our encryption algorithm is equal to $l_1 + (1 + \sum_{i=1}^{a} a_i).n$, and the bit-length of a signature produced by our signature algorithm is equal to $(\sum_{i=1}^{b} b_i).l_2 + l_1$.

The sizes of the ciphertexts and the signatures generated by our policy-based encryption and signature algorithms respectively is highly dependent on the values $\sum_{i=1}^{a} a_i$ and $\sum_{i=1}^{b} b_i$, which then need to be minimized. For this reason, we require that the representation of a policy $\wedge_{i=1}^{m}[\vee_{j=1}^{m_i}[\wedge_{k=1}^{m_{i,j}}\langle TA_{i,j,k}, A_{i,j,k}\rangle]]$ minimizes the sum $\sum_{i=1}^{m} m_i$.

## 5 Security

In this section, we focus on the security properties of our policy-based cryptographic schemes. Informally, a policy-based encryption scheme must satisfy the semantic security property i.e. an adversary who does not fulfill the encryption policy learns nothing about the encrypted message from the corresponding ciphertext. While a policy-based signature scheme must satisfy, on one hand, the existential unforgeability property i.e.

an adversary cannot generate a valid signature without having access to a set of credentials fulfilling the signature policy, and, on the other hand, the credentials ambiguity property i.e. while the verifier is able to check the validity of the signature, there is no way for him to know which set of credentials has been used to generate it. A formal analysis of these security properties requires, in addition to the specification of attacks' goals, the establishment of adequate attack models i.e. chosen ciphertext attacks for policy-based encryption and chosen message attacks for policy-based signature. Because of the lack of space, we only point out, in this paper, the security properties of our schemes and provide intuitive and rather heuristic proofs of our claimed security properties. Our security analysis relies on the random oracle model [2].

### 5.1 Policy-Based Encryption

*Claim.* Our policy-based encryption scheme is semantically secure in the random oracle model under the assumption that BDHP is hard.

Given a policy $\mathrm{pol}_\mathrm{A} = \wedge_{i=1}^{a}[\vee_{j=1}^{a_i}[\wedge_{k=1}^{a_{i,j}}\langle TA_{i,j,k}, A_{i,j,k}\rangle]]$, we provide in the following a proof sketch of our claim through a step-by-step approach going from simple cases to more general ones.

Case 1. Assume that $a = 1$, $a_1 = 1$ and $a_{1,1} = 1$ i.e. $\mathrm{pol}_\mathrm{A} = \langle TA_{1,1,1}, A_{1,1,1}\rangle$. Here, our policy-based encryption algorithm is reduced to an ID-based encryption algorithm similar to algorithm FullIdent defined in [3]. Thus, we can define a game between a challenger and an adversary and run a corresponding simulation proving that our algorithm is secure as long as BDHP is hard. The game we may define is similar to the one described in Section 2 of [3]. The only difference is in the definition of extraction queries. In [3], an extraction query allows the adversary to get the credential corresponding to any specified identity $ID_i$, with the natural restriction that he does not get the credential corresponding to the identity $ID_i^*$ on which he is challenged. As we deal with multiple trusted authorities, an extraction query in our game should allow the adversary to get the credential corresponding to any pair $(TA_{i,j,k}, A_{i,j,k})$ he specifies, with the natural restriction that he does not get the credential corresponding to the pair $(TA_{i,j,k}^*, A_{i,j,k}^*)$ on which he is challenged. Notice that the adversary learns nothing about the challenge pair from queries on pairs $(TA_{i,j,k}^*, A_{i,j,k})$ and $(TA_{i,j,k}, A_{i,j,k}^*)$ because the trusted authorities generate their master-keys randomly and independently. Thus, we may conclude that our policy-based encryption algorithm is as secure as FullIdent. The latter is, in fact, proven to be semantically secure against chosen ciphertext attacks in the random oracle model.

Case 2. Assume that $a = 1$, $a_1 = 1$ and $a_{1,1} > 1$ i.e. $\mathrm{pol}_\mathrm{A} = \wedge_{k=1}^{a_{1,1}}\langle TA_{1,1,k}, A_{1,1,k}\rangle$. As for the previous case, we can define a game and run a corresponding simulation proving that our algorithm is secure as long as BDHP is hard. Here, each extraction query should allow the adversary to ask the challenger each time for the credentials corresponding to $a_{1,1}$ pairs of the form $(TA_{i,j,k}, A_{i,j,k})$, instead of a single pair as for the previous case. The only restriction is that the adversary does not get all the credentials corresponding to the set of pairs $\{(TA_{i,j,k}^*, A_{i,j,k}^*)_1, \ldots, (TA_{i,j,k}^*, A_{i,j,k}^*)_{a_{1,1}}\}$ on which he is challenged. The fact that the game defined for the previous simple case allows the adversary to perform an unlimited number of extraction queries, leads to the conclusion that our encryption algorithm remains semantically secure when $a = 1$, $a_1 = 1$ and $a_{1,1} > 1$.

Case 3. Assume that $a = 1$ and $a_1 > 1$ i.e. $\text{pol}_\text{A} = \vee_{j=1}^{a_1}[\wedge_{k=1}^{a_{1,j}}\langle TA_{1,j,k}, A_{1,j,k}\rangle]$. Here, the difference with the previous case is that the ciphertext contains $a_1$ encryptions of the randomly generated ephemeral key $t_1$, instead of a single one as for the previous case. The fact that $H_2$ is a random oracle allows to generate a different uniformly distributed pad for each of the input entries $(g_{1,j}^r, 1, j)$. The semantic security of the Vernam one-time pad leads to the conclusion that our encryption algorithm remains semantically secure when $a = 1$ and $a_1 > 1$.

Case 4. Assume that $a > 1$ (this corresponds to the general case). First of all, notice that for all $i$, encrypting $a_i$ times the ephemeral key $t_i$ does not weaken its security because the random oracle hash function $H_2$ outputs different uniformly-distributed pads for the different input entries $(g_{i,j}^r, i, j)$ so that no pad is used more than one time. Now, we give an intuitive recursive proof of the semantic security of our policy-based encryption scheme. Assume that the encryption is semantically secure if $a = A$ for some $A$, and consider the case where $a = A + 1$. For a given message $m$, let $c = (U, [v_{i,1}, v_{i,2}, \ldots, v_{i,a_i}]_{1 \le i \le p+1}, w = m \oplus H_3(\oplus_{i=1}^{A+1} t_i)$ be the ciphertext generated by our policy-based encryption algorithm. Let $c_\text{A} = (U, [v_{i,1}, v_{i,2}, \ldots, v_{i,a_i}]_{1 \le i \le A}, w_\text{A} = m \oplus H_3(\oplus_{i=1}^{A} t_i))$ and $c_{A+1} = (U, [v_{A+1,1}, v_{A+1,2}, \ldots, v_{A+1,a_{A+1}}], w_\text{A} \oplus H_3(t_{A+1}))$. We know that the adversary learns nothing about $m$ from $c_\text{A}$. Moreover, that the adversary learns nothing neither about $m$ nor about $w_\text{A}$ from $c_{A+1}$ thanks to the random oracle assumption. This leads to the fact that the adversary gets no useful information about $m$ from $c_\text{A}$ and $c_{A+1}$. As the different ephemeral keys $t_i$ are generated randomly, it is highly improbable that $\oplus_{i=1}^{A} t_i = t_{A+1}$. Because $m \oplus H_3(\oplus_{i=1}^{A+1} t_i)$ is at least as secure as $m \oplus H_3(\oplus_{i=1}^{A} t_i) \oplus H_3(t_{A+1})$, we may conclude that our policy-based encryption algorithm achieves the semantic security property.

### 5.2 Policy-Based Signature

*Claim.* Our policy-based signature scheme achieves signature unforgeability in the random oracle model under the assumption that DLP and BPIP are hard.

Given policy $\text{pol}_\text{B} = \wedge_{i=1}^{b}[\vee_{j=1}^{b_i}[\wedge_{k=1}^{b_{i,j}}\langle TA_{i,j,k}, A_{i,j,k}\rangle]]$, we give an intuitive proof of our claim similarly to the proof given in [13]: an adversary who does not possess a set of credentials fulfilling $\text{pol}_\text{B}$ may try to generate a signature $\sigma = ([x_{i,1}, x_{i,2}, \ldots, x_{i,b_i}]_{1 \le i \le b}, Y)$ on a message $m$ according to $\text{pol}_\text{B}$ through two possible attacks. On one hand, the adversary chooses the values $x_{i,j}$ for all $1 \le i \le b$ and all $1 \le j \le b_i$, then tries to compute $Y$ such that $\sigma$ is valid i.e. the adversary computes $Y$ from the equation

$$e(P, Y) = [\Pi_{i=1}^{b}[\Pi_{j=1}^{b_i} x_{i,j}]] * [\Pi_{i=1}^{b}[\Pi_{j=1}^{b_i} \tau_{i,j}^{H_4(m \| x_{i,j} \| \text{pol}_\text{B})}]]^{-1}$$

Such attack is equivalent to solving PBIP which is assumed to be hard. On the other hand, the adversary chooses $Y$ and all the values $x_{i,j}$ for $1 \le i \le b$ and $1 \le j \le b_i$ but the value $x_{i_0,j_0}$ for certain $1 \le i_0 \le b$ and $1 \le j_0 \le b_{i_0}$, then tries to compute $x_{i_0,j_0}$ such that $\sigma$ is valid i.e. the adversary solves the equation

$$x_{i_0,j_0} = \xi * \tau_{i_0,j_0}^{H_4(m\|x_{i_0,j_0}\|\mathrm{pol_B})}$$

where $\xi = [\prod_{i \neq i_0}[\prod_{j \neq j_0} x_{i,j}]]^{-1} * e(P,Y) * [\prod_{i \neq i_0}[\prod_{j \neq j_0} \tau_{i,j}^{H_4(m\|x_{i,j}\|\mathrm{pol_B})}]]$. Because $H_4$ is assumed to be a random oracle, there's no way for the adversary to solve such equation apart from a brute force approach which consists in trying all the elements of $\mathbb{G}_2$. Hence, the probability of forging a signature through this attack is less than $1/q$ which is considered to be negligible.

*Claim.* Our policy-based signature scheme achieves credentials ambiguity in the random oracle model.

We give an intuitive proof of our claim similarly to the proof given in [13]: for all indices $i$, $Y_i$ is chosen randomly in $\mathbb{G}_1$ which means that $x_{i,j_i}$ is uniformly distributed in $\mathbb{G}_2$. Similarly, for all indices $i$ and $l$, $Y_{i,l}$ is chosen randomly in $\mathbb{G}_1$ which leads to the fact that all $x_{i,l}$ are uniformly distributed in $\mathbb{G}_2$. Thus, given a message $m$ and the signature $\sigma = ([x_{i,1}, x_{i,2}, \ldots, x_{i,b_i}]_{1 \leq i \leq b}, Y)$ on $m$ according to $\mathrm{pol_B}$, $\sigma$ does not reveal which credentials have been used to generate it.

## 6 Application Scenarios

Assume that Bob (service provider) controls a sensitive resource 'res', and that for a specific action 'act' on 'res', he defines a policy 'pol' which specifies the conditions under which 'act' may be performed on 'res'. Assume that Alice (service requester) wants to perform action 'act' on 'res'. As a simple example, we assume that Bob's policy is

$$\mathrm{pol_B} = \langle \mathrm{IFCA}, \mathrm{alice{:}member} \rangle \wedge [\langle \mathrm{X}, \mathrm{alice{:}employee} \rangle \vee \langle \mathrm{Y}, \mathrm{alice{:}employee} \rangle]$$

Here 'IFCA' stands for the International Financial Cryptography Association, while 'X' and 'Y' are two partners of Bob. Bob's policy states that in order for Alice to be authorized to perform action 'act' on 'res', Alice must be a member of IFCA as well as an employee of either partner 'X' or partner 'Y'. We assume, for instance, that Alice is a member of 'IFCA' and works for 'X' i.e. Alice possesses the secret credentials $\varsigma_{\mathrm{IFCA}} = \varsigma(R_{\mathrm{IFCA}}, \mathrm{alice{:}member})$ and $\varsigma_{\mathrm{X}} = \varsigma(R_{\mathrm{X}}, \mathrm{alice{:}employee})$. In the following, we describe three different policy enforcement scenarios and show how our approach allows performing privacy-aware policy enforcement (with respect to the data minimization principle).

**Scenario 1.** Assume that 'res' is a PDF file containing a confidential report and assume that Alice wants to have a read access to the report. Here, the only concern of Bob is to ensure that Alice does not read the file if she is not compliant to $\mathrm{pol_B}$. He needs to know neither whether Alice fulfills his policy or not, nor whether she is an employee of X or Y. The standard approach allows Bob to get such 'out-of-purpose' information because Alice has to show her credentials in order to prove her compliance to $\mathrm{pol_B}$, whilst our policy-based cryptographic approach allows to avoid this privacy flaw as follows:

1. First, Bob encrypts the protected file according to policy $\mathrm{pol_B}$ i.e. Bob computes $c = \mathsf{PolEnc}(\mathrm{res}, \mathrm{pol_B})$. Then, he sends $c$ to Alice. Note that practically, Bob does not encrypt res but the session key which encrypts res.

2. Upon receiving $c$, Alice decrypts it using her secret credentials i.e. Alice computes $\text{res} = \mathsf{PolDec}(c, \text{pol}_{\text{B}}, \{\varsigma_{\text{IFCA}}, \varsigma_{\text{X}}\})$

Scenario 1 may be applied to solve the cyclic policy interdependency problem as described in [12, 9]. An additional interesting application of policy-based encryption is the sticky privacy policy paradigm, first defined in [11], according to which the policy that is specified and consented by data subjects at collection, and which governs data usage, holds true throughout the data's lifetime, even when the data is disclosed by one organization to another. Thus, a data subject may encrypt his private data according to a policy reflecting his privacy preferences. The exchange of encrypted privacy-sensitive data ensures that only principals fulfilling the privacy requirements are able to perform the decryption operation successfully and retrieve the privacy-sensitive data. As an illustrative example, a user Alice may require that a company is a member of either the Better Business Bureau (BBB) or the International Chamber of Commerce (ICC) in order to be able to have access to her professional e-mail address (*alice@X.net*). Thus, Alice may encrypt *alice@X.net* according to her policy

$$\text{pol}_{\text{A}} = \langle \text{BBB}, \text{member:current-year} \rangle \vee \langle \text{ICC}, \text{member:current-year} \rangle$$

**Scenario 2.** Assume that 'res' is a CD-ROM containing a confidential piece of software and that Alice asks Bob to ship it to her home address. The only useful information for Bob is to know whether Alice is compliant to $\text{pol}_{\text{B}}$ or not. He does not need to know for which company Alice works. While the standard approach obliges Alice to show her employee credential in order to prove her compliance to $\text{pol}_{\text{B}}$, our policy-based cryptographic approach allows to avoid this privacy flaw as follows:

1. First, Bob picks a random challenge nonce $n_{\text{ch}}$ and encrypts it according to $\text{pol}_{\text{B}}$ i.e. Bob computes $c = \mathsf{PolEnc}(n_{\text{ch}}, \text{pol}_{\text{B}})$. Then, he sends $c$ to Alice as a 'policy compliance' challenge
2. Upon receiving $c$, Alice decrypts it using her secret credentials i.e. Alice computes $n_{\text{resp}} = \mathsf{PolDec}(c, \text{pol}_{\text{B}}, \{\varsigma_{\text{IFCA}}, \varsigma_{\text{X}}\})$. Then Alice sends $n_{\text{resp}}$ as a response for Bob's challenge
3. Upon receiving $n_{\text{resp}}$, Bob checks whether $n_{\text{resp}} = n_{\text{ch}}$ in which case he authorizes the shipping of the requested CD-ROM to Alice's home address. If Alice does not send her response or if the response is not equal to the challenge nonce, Bob infers that she is not compliant to $\text{pol}_{\text{B}}$ and thus does not authorize the shipping of the requested CD-ROM

Scenario 2 applies either when the action 'act' on the sensitive resource 'res' is different from 'read' or when the communication partners wish to conduct mutli-round transactions during which a party needs to know whether the other is compliant to his policy or not.

**Scenario 3.** Consider the previous scenario while assuming now that Bob wishes to keep a non-forgeable and/or non-repudiable proof that Alice is compliant to $\text{pol}_{\text{B}}$. In the standard approach, Bob gets all the credentials of Alice allowing her to prove her compliance to $\text{pol}_{\text{B}}$. In this case, the set of received credentials may be seen as a policy compliance proof. In addition to the required proof, Bob knows for which company Alice works. The collection of such 'out-of-purpose' information represents a privacy flaw which could be avoided using our policy-based cryptographic approach as follows:

1. First, Bob picks a random challenge nonce $n_{\mathrm{ch}}$ and sends it to Alice
2. Upon receiving the challenge, Alice signs it according to $\mathrm{pol}_{\mathrm{B}}$ using her secret credentials i.e. Alice computes $\sigma = \mathsf{PolSig}(n_{\mathrm{ch}}, \mathrm{pol}_{\mathrm{B}}, \{\varsigma_{\mathrm{IFCA}}, \varsigma_{\mathrm{X}}\})$. Then Alice sends $\sigma$ to Bob as a response for his challenge
3. Upon receiving $\sigma$, Bob checks whether it is a valid signature with respect to $\mathrm{pol}_{\mathrm{B}}$ i.e. Bob checks whether $\mathsf{PolVrf}(n_{\mathrm{ch}}, \mathrm{pol}_{\mathrm{B}}, \sigma) = \top$, in which case Bob authorizes the requested action to be performed (CD-ROM shipping)

Scenario 3 allows a number of interesting value-added services such as accountability i.e. Alice cannot deny being compliant to Bob's policy at certain period in time, service customization i.e. Bob may make a special offers or discounts to customers respecting $\mathrm{pol}_{\mathrm{B}}$ at a certain period in time, policy-based single sign-on i.e. based on Alice's poof of compliance to policy $\mathrm{pol}_{\mathrm{B}}$, Alice may get multiple services from Bob's partners (within a federation) without re-proving her compliance to $\mathrm{pol}_{\mathrm{B}}$, etc. Note that the non-repudiation property is due to the fact that the credentials are attached to Alice's name (identifier).

## 7 Related Work

Many cryptography-based policy enforcement mechanisms have been presented over the years, especially in the context of access control. In [16], for instance, Wilkinson et al. show how to achieve trustworthy access control with untrustworthy web servers through standard symmetric and asymmetric cryptographic mechanisms. Their approach allows removing access control responsibilities from web server software which are subject to failure, while delegating access control functionalities to encryption and decryption proxies. Their access control 'expressions' (policies) are described through conjunctions and disjunctions of groups each containing a number of users. They describe how they perform encryption operations and generate decryption keys according to these policies. Their approach remains naive in the sense that they use onion-like encryptions to deal with conjunctions and multiple encryptions to deal with disjunctions. Moreover, they use standard public key cryptography whose main drawback consists in dealing with public key certificates. This weakness could be avoided by using identity-based cryptography [14, 3].

In [7], Chen et al. investigate a number of issues related to the use of multiple authorities in ID-based encryption from bilinear pairings. They present a number of interesting applications of the addition of keys, and show how to perform encryptions according to disjunctions and conjunctions of keys. However, their solution remains restricted to limited disjunctions of keys. In [15], Smart continues the ideas discussed in [7]. He presents an elegant and efficient mechanism to perform encryption according to arbitrary combinations of keys, yet generated by a single trusted authority. Our work could be seen as an extension of [15] in the sense that we use the same policy model while allowing multiple trusted authorities and defining the policy-based signature primitive.

Apart from access control systems, the exchange of digital credentials is an increasingly popular approach for trust establishment in open distributed systems where communications may occur between strangers. In such conditions, the possession of certain credentials may be considered as security or privacy sensitive information. Automated trust negotiation (ATN) allows regulating the flow of sensitive credentials during trust establishment through the definition of disclosure policies. One of the major problems in ATN is called the cyclic policy interdependency which occurs when a communication party is obliged to be the first to reveal a sensitive credential to the other. In [12], Li et al. model the cyclic policy interdependency problem as a 2-party secure function evaluation (SFE) and propose oblivious signature-based envelopes (OSBE) for efficiently solving the FSE problem. Among other schemes, they describe an OSBE scheme based on ID-based cryptography which is almost similar to our policy-based encryption scheme in the particular case where the considered policy is satisfied by a single credential. Thus, our encryption scheme could be seen as a generalization of the identity-based OSBE scheme.

In [9], Holt et al. introduce the notion of hidden credentials which are similar to our policy-based encryption scheme in that the ability to read a sensitive resource is contingent on having been issued the required credentials. In contrast with OSBE, hidden credentials deal with complex policies expressed as monotonic boolean expressions. They use onion-like encryptions and multiple encryptions to deal with conjunctions and disjunctions respectively. Their approach remains inefficient in terms of both computational costs and bandwidth consumption (ciphertext size) especially when authorization structures become very complex. While our policy-based encryption and signature schemes are based on publicly known policies, hidden credentials consider the policies as sensitive so that they should never be revealed. Thus, decryptions are performed in a blind way in the sense that the decrypting entity has not only to possess a set of credentials satisfying the encryption policy but also to find the correct combination of credentials corresponding to the policy structure. Very recently, Bradshaw et al. proposed a solution to improve decryption efficiency as well as policy concealment when implementing hidden credentials with sensitive policies [5].

In [6], Brands introduced practical techniques and protocols for designing, issuing and disclosing private credentials. He describes in chapter 3 of [6] a set of showing protocols enabling the credentials owner to selectively disclose properties about them. Brands' approach is data subject-centric, while our approach for privacy focuses on the quality of data exchange during privacy-sensitive transactions. Besides, Brands' credentials are based on standard public key cryptography, whilst our policy-based cryptographic schemes are based on identity-based cryptography from bilinear pairings.

## 8 Conclusion

In this paper, we introduced the concept of policy-based cryptography which allows performing privacy-aware policy enforcement in large-scale distributed systems like the Internet. We mainly focused on the compliance to the data minimization principle which has been advocated by several privacy protection guidelines and legislations. We defined the policy-based encryption and signature primitives, and we proposed concrete schemes from bilinear pairings. Our algorithms allow handling complex policies in an elegant and relatively efficient manner. Moreover, their properties allow using them in a wide range of applications, from the traditional access control systems to the more sophisticated privacy protection and trust establishment systems. Future research may focus on improving the efficiency of the proposed policy-based schemes and on developing additional policy-based cryptographic primitives. We are currently investigating the real deployment of our policy-based approach in the context of sticky privacy policies. Besides, we are developing formal security models and proofs for policy-based cryptographic schemes.

## References

1. P. Barreto, H. Kim, B. Lynn, and M. Scott. Efficient algorithms for pairing-based cryptosystems. In *Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, pages 354–368. Springer-Verlag, 2002.
2. M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on Computer and communications security*, pages 62–73. ACM Press, 1993.
3. D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, pages 213–229. Springer-Verlag, 2001.
4. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. In *Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security*, pages 514–532. Springer-Verlag, 2001.
5. R. Bradshaw, J. Holt, and K. Seamons. Concealing complex policies with hidden credentials. In *Proceedings of the 11th ACM Conference on Computer and Communications Security*, pages 146–157. ACM Press, 2004.
6. S. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. MIT Press, 2000.
7. L. Chen, K. Harrison, D. Soldera, and N. Smart. Applications of multiple trust authorities in pairing based cryptosystems. In *Proceedings of the International Conference on Infrastructure Security*, pages 260–275. Springer-Verlag, 2002.
8. Organization for Economic Cooperation and Development (OECD). Recommendation of the council concerning guidelines governing the protection of privacy and transborder flows of personal data, 1980. `http://www.oecd.org/home/`.
9. J. Holt, R. Bradshaw, K. E. Seamons, and H. Orman. Hidden credentials. In *Proc. of the 2003 ACM Workshop on Privacy in the Electronic Society*. ACM Press, 2003.
10. A. Joux. The weil and tate pairings as building blocks for public key cryptosystems. In *Proceedings of the 5th International Symposium on Algorithmic Number Theory*, pages 20–32. Springer-Verlag, 2002.
11. G. Karjoth, M. Schunter, , and M. Waidner. The platform for enterprise privacy practices–privacy-enabled management of customer data. In *2nd Workshop on Privacy Enhancing Technologies (PET 2002)*, volume 2482 of *LNCS*, pages 69–84. Springer-Verlag, April 2002.
12. N. Li, W. Du, and D. Boneh. Oblivious signature-based envelope. In *Proceedings of the 22nd annual symposium on Principles of distributed computing*, pages 182–189. ACM Press, 2003.
13. C. Lin and T. Wu. An identity-based ring signature scheme from bilinear pairings. In *Proceedings of the 18th International Conference on Advanced Information Networking and Applications*. IEEE Computer Society, 2004.

14. A. Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 47–53. Springer-Verlag New York, Inc., 1985.
15. N. Smart. Access control using pairing based cryptography. In *Proceedings CT-RSA 2003*, pages 111–121. Springer-Verlag LNCS 2612, April 2003.
16. T. Wilkinson, D. Hearn, and S. Wiseman. Trustworthy access control with untrustworthy web servers. In *Proceedings of the 15th Annual Computer Security Applications Conference*, page 12. IEEE Computer Society, 1999.
17. Y. Yacobi. A note on the bilinear diffie-hellman assumption. Cryptology ePrint Archive, Report 2002/113, 2002. `http://eprint.iacr.org/`.
18. F. Zhang and K. Kim. Id-based blind signature and ring signature from pairings. In *ASIACRYPT*, pages 533–547. Springer-Verlag LNCS 2501, 2002.

# A Chat at the Old Phishin' Hole

Richard Clayton, Drew Dean, Markus Jakobsson, Steven Myers,
Stuart Stubblebine, and Michael Szydlo

Phishing is an attack in which victims are lured by official looking email to a fraudulent web-site that appears to be that of a legitimate service provider. The email also provides victims with a convincing reason to log-on to the site. If users are fooled into logging-on, then the attacker is provided with the victims' authentication information for the legitimate service provider, often along with personal information, such as their credit-card data, checking account information or social security data. Successful phishing attacks can result not only in identity and asset theft, but also in more subtle attacks that need not be directly directly harmful to the victim but which have negative consequences for society (for example: money laundering).

Professional studies that have attempted to estimate the direct losses due to phishing in 2004 have come up with widely varying figures: from \$150-million to \$2.4-billion U.S. dollars. However, all the studies agree that the costs will continue to rise in the foreseeable future unless something is done to educate users and/or technologies are introduced to defeat or limit such attacks. Further, these estimates measure only the direct costs, and do attempt to measure the indirect costs that result from the loss of consumer confidence in the Internet infrastructure and all of the services it can be used to provide. Our panel will look at a broad number of issues relating to the past, present and future of phishing, in order to better understand this growing problem.

We will address topics that include the notion that phishing is a special case of "web-spoofing", an attack that was predicted and researched academically as early as 1996. We will look at the mutual progression of the research and practice of such attacks, and what we can learn from both. We will discuss the fact that phishing is currently a problem, and look at what information consumers are being given to mitigate their risk of exposure; we'll ask if the advice is practical and effective. We will see how the percentage of successful phishing attacks could dramatically increase if phishing attacks begin to make use of contextual information about their victims. It will be argued that such attacks are easily automated, begging the question of how long it will take for such context sensitive attacks to appear in the wild. We will see that phishing-graphs can be used not only to model phishing attacks, but also to quantify the feasibility and economic costs of attacks. We will discuss the issue of mutual authentication, and how it relates to phishing attacks. It will be argued that easy to use mutual authentication protocols could mitigate many of the risks of phishing, and we will discuss one such protocol. Finally, we will deliberate on the likelihood of the advent of a silver-bullet technology that will solve all of our phishing problems.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, p. 88, 2005.
© Springer-Verlag Berlin Heidelberg 2005

# Modeling and Preventing Phishing Attacks

Markus Jakobsson

School of Informatics,
Indiana University at Bloomington,
Bloomington, IN 47406
`www.markus-jakobsson.com`

A *first contribution* of this paper is a theoretical yet practically applicable model covering a large set of phishing attacks, aimed towards developing an understanding of threats relating to phishing. We model an attack by a *phishing graph* in which nodes correspond to knowledge or access rights, and (directed) edges correspond to means of obtaining information or access rights from already possessed information or access rights – whether this involves interaction with the victim or not. Edges may also be associated with probabilities, costs, or other measures of the hardness of traversing the graph. This allows us to quantify the effort of traversing a graph from some starting node (corresponding to publicly available information) to a target node that corresponds to access to a resource of the attacker's choice. We discuss how to perform economic analysis on the viability of attacks. A quantification of the economical viability of various attacks allows a pinpointing of weak links for which improved security mechanisms would improve overall system security.

A *second contribution* of this paper is the description of what we term a *context aware* phishing attack. This is a particularly threatening attack in that it is likely to be successful *not only* against the most gullible computer users (as is supported by experimental results we present.) A context aware attack is mounted using messages that somehow – from their context – are expected (or even welcomed) by the victim. To draw a parallel from the physical world, most current phishing attacks can be described as somebody who knocks on your door and says you have a problem with your phone, and that if you let him in, he will repair it. A context aware phishing attack, on the other hand, can be described by somebody who first cuts your phone lines as they enter your home, waits for you to contact the phone company to ask them to come and fix the problem – and *then* knocks on your door and says he is from the phone company. We can see that observing or manipulating the context allows an attacker to make his victim lower his guards. As a more technical example, we show how to obtain PayPal passwords from eBay users that do not take unusual measures *particularly intended* to avoid this attack.

Finally, a *third contribution* is a discussion of how to address the threats we describe – both in their specific and generic shapes.

A full version of this paper can be downloaded from
`www.markus-jakobsson.com`

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, p. 89, 2005.
© Springer-Verlag Berlin Heidelberg 2005

# Helping the Phish Detect the Lure

Steven Myers

School of Informatics, Indiana University at Bloomington,
Bloomington, IN 47406, USA
samyers@indiana.edu

When a client attempts to interact with an online service provider that performs any form of financial transaction, the service provider requires the client to authenticate itself. This is normally done by having the client provide a username and password that were previously agreed upon, through some procedure, the first time the client attempted to use the services provided by the provider. Asymmetrically, the client does not ask the provider for the same form of authentication. That is, the customer of the bank does not ask the web-page to somehow prove that it is really the bank's web-page. This asymmetry seems to come mostly from an attempt to port security models from the physical to the digital world: I would never expect a physical bank branch to authenticate itself to me through any form other than its branding. However, that is not to say customers don't implicitly authenticate their bank-branches, they do! However, it is a rather implicit authentication that is based on the use of branding and law-enforcement by the banks. Unfortunately, many of the security assumptions that hold in the physical world do not hold in the digital world: the costs of setting up an authentic looking but fraudulent web-page are low; the pay-off for successful phishing attacks is high; and digital law enforcement is weak to non-existent in the digital realm and so the risks are minimal. This makes phishing an attractive type of fraud, and has led to its growing popularity.

In order to reduce the ability of phishers to launch successful attacks, we suggest that users request authentication from their service providers. In other words, we suggest that the client and service provider engage in mutual authentication. While such authentication is easily achievable with public-key cryptography and certificates, this solution is not appealing due to the historical difficulty users have had in understanding these concepts: currently many users automatically accept most certificates that are brought to their attention by web-browsers, regardless of their validity or origin.

We will discuss a protocol for mutual authentication that relies solely on a client being able to remember a password to authenticate him or herself to the service provider, and the ability to recognize —and not recall, as in the case of a password— a unique series of images (or other forms of stimuli, such as sound and touch) corresponding to the appropriate service provider. The client only needs to be educated to realize that if his or her appropriate sequence of images does not appear, then the site is not legitimate and should not be used, nor should any personal information be provided to it. Further, the protocol has the property that it is secure against man-in-the-middle attacks in the random-oracle model.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, p. 90, 2005.
© Springer-Verlag Berlin Heidelberg 2005

# Who'd Phish from the Summit of Kilimanjaro?

Richard Clayton

University of Cambridge, Computer Laboratory
richard.clayton@cl.cam.ac.uk

Phishing emails are now so convincing that even experts cannot tell what is or is not genuine[1]; though one of my own quiz answering errors resulted from failing to believe that genuine marketeers could possibly be so clueless! Thus I believe that education of end users will be almost entirely ineffective and education of marketing departments – to remove "click on this" (and HTML generally) from the genuine material – is going to take some time.

Providing end users with one-time passwords (pads of single-use numbers, SecurID tokens, PINs sent by mobile phone) can ensure that phishing only works when there is a real-time, Man-in-the-Middle (MITM), attack. This will immediately deter the bad guys if their technical expertise runs solely to copying websites. However, formal analysis of online banking protocols shows that only a handful of the "bag of bits" being passed around can be considered to be authenticated – and so a MITM can, unhindered, steal whatever they wish.

Insisting on SSL (`https`) connections will prevent the use of random URLs for phishing websites and bring the focus back to control of the DNS. However, once the second level (`fakebankname.com`) is secured then the attackers will just move down a level (to `bankname.plausible-second-word.com`). I predict a lot of wasteful activity before the nature of DNS delegation is fully understood.

Insisting on client certificates prevents MITM attacks, but also stops me paying my gas bill from a holiday cybercafé – which is bad for business. But why do I need the same authority to pay the bill as to change the name of the gas company? A range of authentication systems is needed, chosen as the risk varies. The banks could learn from the activity monitoring systems of the credit card companies, and ensure that extra authentication is seldom necessary or onerous. For example, a check can be made on the IP address of incoming connections. If the session arrives from a cybercafé in Latvia or a web hosting rack in suburban Moscow then Mr. Jones in Acacia Avenue is not connecting directly... if he really does want to set up a new payee then perhaps he could ring his branch and confirm that he's taking an East European holiday?

To conclude; I can see no silver bullet (I can imagine success for phishing emails that ask for client certificates), and most of the proposed argento-ammunition is useless once the end-user machine is compromised. Nevertheless,

[1] MailFrontier Phishing IQ Test II `http://survey.mailfrontier.com/survey`

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 91–92, 2005.
© Springer-Verlag Berlin Heidelberg 2005

a blend of security improvements will freeze out all but the most competent criminals. Society may need a general solution to online security, but the banks only have to persuade the bad guys to move on to more attractive targets. However, the fixes must *not* be introduced one by one, allowing each to be overcome individually. What's needed is a 'Kilimanjaro effect', where the security suddenly dominates the landscape and it will always seem to be a long way to the summit.

# A Privacy-Protecting Coupon System

Liqun Chen[1], Matthias Enzmann[2,⋆], Ahmad-Reza Sadeghi[3],
Markus Schneider[2], and Michael Steiner[4]

[1] HP Labs, Filton Road, Stoke Gifford, Bristol BS34 8QZ, United Kingdom
liqun.chen@hp.com

[2] Fraunhofer Gesellschaft (FhG), Institute for Secure Information Technology (SIT),
Dolivostr. 15, D-64293 Darmstadt, Germany
{matthias.enzmann, markus.schneider}@sit.fraunhofer.de

[3] Ruhr-University Bochum,
Universitätsstr. 150, D-44780 Bochum, Germany
sadeghi@crypto.rub.de

[4] IBM T.J. Watson Research Center,
P.O. Box 704, Yorktown Heights, NY 10598, USA
msteiner@watson.ibm.com

**Abstract.** A coupon represents the right to claim some service which is typically offered by vendors. In practice, issuing bundled multi-coupons is more efficient than issuing single coupons separately. The diversity of interests of the parties involved in a coupon system demands additional security properties beyond the common requirements (e.g., unforgeability). Customers wish to preserve their privacy when using the multi-coupon bundle and to prevent vendors from profiling. Vendors are interested in establishing a long-term customer relationship and not to subsidise one-time customers, since coupons are cheaper than the regular price. We propose a secure multi-coupon system that allows users to redeem a predefined number of single coupons from the same multi-coupon. The system provides unlinkability and also hides the number of remaining coupons of a multi-coupon from the vendor. A method used in the coupon system might be of independent interest. It proves knowledge of a signature on a message tuple of which a single message can be revealed while the remaining elements of the tuple, the index of the revealed message, as well as the signature remain hidden.

## 1 Introduction

Today, coupons appear to be useful means for vendors to attract the attention of potential customers. Usually, coupons give the customer a financial incentive to purchase at a specific vendor. The purpose of coupons is many-fold. For instance, they can be used to draw the attention of customers to a newly opened shop or to prevent customers from buying at a competitor's shop [29]. Of course, coupons can also be purchased by

[⋆] The author was supported in part by the German Federal Ministry of Education and Research under grant 01AK706C, project *Premium*.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 93–108, 2005.

© Springer-Verlag Berlin Heidelberg 2005

customers, e.g., gift certificates. Even drug prescriptions from a doctor can be seen as a kind of a coupon.

In general, a coupon is a representation of the right to claim some good or service, usually from the party that issued the coupon. The types of coupons mentioned before can, in general, be redeemed only once, i.e., the coupon is invalidated after the service or good has been claimed. However, there are also coupons which can be redeemed more than once, such as a coupon book of a movie theater, where customers pay, e.g., for 9 movies and are entitled to see 10. We call such coupons *multi-coupons*. In this paper, we are particularly interested in this type of coupons.

Typically, a real-world multi-coupon of value $m$ is devalued by crossing out some field or by detaching a part of it. Offering such coupons can be beneficial for the issuing party, e.g., a movie theater. First, customers pay in advance for services or goods they have not claimed yet. Second, they are locked-in by the issuer/vendor, i.e., they are unlikely to switch to another vendor to purchase the same or similar service or good as long as they have not redeemed all their coupons. Hence, multi-coupons can also be seen as a kind of loyalty program since they are specific to some vendor and induce loyalty, at least, as long as the customer has coupons left to spend.

Clearly, vendors are interested in creating loyalty and hence, it is likely that we are going to see such coupon systems in the Internet, too. In fact, introducing such a coupon system might be even more valuable to Internet vendors than to their real-world counterparts. Since, from the customers' viewpoint, *a priori* all vendors, offering a certain good or service, look alike and can be reached as easily as their competitors.

This is in contrast to the real world where time and effort is required to go to physical stores to acquire information on that store's products [23]. In the real world, additional barriers may exist, such as physical distance or some kind of relationship to shop personnel, that incur indirect switching costs for customers [21]. In summary, it can be expected that in absence of notable switching costs customer fluctuation is higher in the Internet than in the real world because there is little that keeps customers from buying at a competitor's site whenever this competitor offers a better price. Thus, it is in the vendor's interest to introduce switching costs in order to retain an installed base of customers [30].

## 1.1 Desirable Properties for Coupon Systems

At first, introducing a coupon system looks like a win-win situation, since both parties seem to benefit from such a coupon system. Vendors have a means to create a loyal customer base and customers value the financial benefit provided by coupons. However, since a customer normally redeems her coupons in different transactions, a multi-coupon can be used as a means to link transactions, and thus, to allow a vendor to create a record of the customer's past purchases. Such customer information might be exploited for data mining, to infer new customer data, customer profiling, promotion of new products, price discrimination, etc. [24]. Thus, if through usage of the coupon system customers expect a misuse of their personal data, e.g., by using it to create profiles for price discrimination [16], they are more likely to decline the coupon system. According to [19, 20] privacy is a concern to Internet users, especially when it comes to electronic commerce scenarios. Hence, a prudent vendor should take these concerns into account when planning to offer a coupon system.

In order to rule out privacy concerns of customers from the start, vendors might want to introduce a coupon system that does not infringe their customers' privacy. Thus, a coupon should disclose as little information as possible. For instance, a multi-coupon should only give vendors an indication that it is still valid, i.e., that at least one coupon is not spent, instead of disclosing the number of unspent coupons. Such a property could be useful in sensitive areas, e.g., in health care scenarios, where a multi-coupon can be used as a prescription for a certain number of doses of some medicine. In this case, the pharmacist would deduct a single coupon from the multi-coupon and may only detect if the prescription has been used up. Also in welfare, paper-based checks or food stamps could be replaced by electronic coupons. In fact, recently, the U.S. announced to replace their paper-based food stamp program with electronic benefits and debit cards [26]. However, this electronic program does not protect the privacy of recipients, since the cards are processed similar to ordinary debit cards.

For vendors, in addition to common security requirements such as unforgeability, there are other requirements which are specific to a coupon system. As mentioned before, a vendor's driving reason for offering a coupon system is to establish a long term relationship with customers. However, customers may be interested in sharing a multi-coupon, i.e., each customer obtains and redeems a fraction of the coupons in the multi-coupon. Moreover, this behaviour allows them, e.g., to sell coupons on an individual basis for a cheaper price[1], e.g., to one-time customers who otherwise would have purchased full-price services or goods. Thus, ideally, vendors wish to prevent customers from *splitting* their coupons.

To illustrate splitting, we consider the following variants as examples of real-world multi-coupons. The first variant, being a coupon book with detachable coupons and the second one being a multi-coupon where spent coupons are crossed out, i.e., coupons cannot be detached. The coupon book can be easily shared by a group of customers, since each customer can detach its share of coupons from the coupon book and each coupon may be independently redeemed by a different customer. In the second variant, the multi-coupon must be given to the vendor *as a whole* to allow him to devalue the multi-coupon by crossing out one of the coupons. Hence, in this variant, individual coupons cannot be split and redeemed separately and independently as in the first variant.

Nevertheless, even in the multi-coupon scenario with non-detachable coupons some kind of sharing is possible if we transfer it to the digital world. Since digital coupons can be easily copied, colluding customers may jointly purchase a multi-coupon, distribute copies of it among each other, and agree to redeem only the share of the coupons for which each of them paid for. In this scenario, however, customers have to fully trust each other that none of them redeems more than its share of coupons. Since each of the colluders owns a copy of the multi-coupon this means that every colluder has full control of all single coupons. Hence, each of them could redeem single coupons of other colluders without their knowledge. A colluder deceived in such a way would only learn

[1] Recall that a multi-coupon for $m$ goods is sold for the price of $m - k$ goods, $k \geq 1$.

about it when he or she tries to redeem a single coupon and the vendor rejects it because it was already spent. Thus, it seems less likely that multi-coupons are traded between customers.

In this context, another scenario with multi-coupons is possible where trust is only one-way. If customer A buys a multi-coupon, uses up, say, half of the coupons and sells the remaining half of the coupons to customer B then A does not have to trust B. Only B has to trust A that he indeed received the purported half of the coupons. There is nothing that really can stop A from doing so, neither in a real-world scenario with paper coupons nor in the digital scenario, unless (a) the multi-coupon contains information that ties it to A's identity and which the vendor may verify (b) customer A has a strong incentive to keep the multi-coupon, e.g., because some private and/or "valuable" information is encoded into the multi-coupon.

We do not pursue any of these two approaches since, first, we do not want to identify customers because this may violate their privacy and, second, encoding valuable information seems to be unsatisfactory as well because encoding a "valuable" secret implies that such a secret exists and that a customer is willing to encode it into a, potentially, considerably less valuable multi-coupon. Instead, we employ *all-or-nothing sharing* which has been used in other works before [3, 8] to *discourage* users from sharing / disclosing certain data, such as private credential information.

In case of a multi-coupon, all-or-nothing means that a customer cannot give away or sell any single coupon from its multi-coupon without giving away all other single coupons — this includes used and unused single coupons alike. Therefore, our scheme is comparable with the real-world multi-coupon example from above where used coupons are crossed out. The difference is that in the digital world one may effortlessly create identical copies of a multi-coupon while in the real world creating exact hardcopies of such a coupon may require some effort.

## 1.2 Overview of Our Coupon System

The coupon system proposed here can be viewed as a digital counterpart to the real-world multi-coupon with non-detachable coupons, as mentioned before. In our coupon system, a multi-coupon $M$ is a signature on a tuple $X$ where $X = (x_1, \ldots, x_m)$. In the system specification, we denote a set of coupons by $M$ and a single coupon by $x \in \{x_1, \ldots, x_m\}$.

In the coupon issue phase, a user first convinces a vendor that she knows $X$ without revealing the values of $X$. Then, the verifier issues the coupon $M$ by "blindly" signing $X$, i.e., $M := Sign(X)$, and sending $M$ to the user. Here we make use of the Camenisch and Lysyanskaya (CL) signature scheme [9].

When redeeming a single coupon $x$, the user reveals $x$ to the vendor and proves that she is in possession of a valid multi-coupon $M = Sign(X)$ and $x \in \{x_1, \ldots, x_m\}$. The vendor then checks if $x$ is in a list of used coupons. If it is not, the vendor accepts $x$ and puts it in the list. Beside satisfying common security requirements, the scheme has the following properties: The vendor is not able to trace $x$ back to $M$ or to link two redemptions since $M$ is never given back to the vendor and the single coupons $x$ are independent of each other. Furthermore, the vendor does not learn anything about the status of the multi-coupon, i.e., how many single coupons are left in the multi-coupon.

Concerning the vendor's requirement, the scheme does not allow users to split a multi-coupon without sharing all values $(x_1, \ldots, x_m)$.

A method used in the coupon system might be of independent interest. It proves knowledge of the CL signature $M$ on a message tuple $X := (x_1, \ldots, x_m)$, of which an arbitrary single message $x_j$ can be revealed while the remaining elements of the tuple, the revealed message's index, $j$, and the signature $M$ remain hidden.

## 2 Related Work

At first it may seem that the coupon system can be easily realised using an existing payment system or credential system which supports $m$-showable credentials or at least one-showable credentials of which $m$ can be obtained. However, none of these systems satisfied all the requirements of the coupon system we had in mind, or could only satisfy them in an inefficient manner. In addition, some of the systems discussed below require the existence of a trusted third party to issue certificates of some sort. We do not have such a requirement.

The payment system of Chaum [11] as well as the one of Brands [2] use digital coins which can be anonymously spent. Withdrawal and spending of coins is roughly the same as issuance and redemption of single coupons. However, using $m$ single-spendable digital coins as a multi-coupon easily allows splitting of the multi-coupon. Even if we would use multi-valued coins such that one unit of an $m$-coin can be spent and an $m-1$ coin is returned, we would still not have the coupon system that we have in mind, since the number of remaining coins is disclosed to the vendor. In the coin system of Ferguson [17] a multi-coin is introduced that can be spent $m$ times. However, when paying with the same multi-coin the vendor learns the remaining value of the coin and, in addition, transactions are linkable.

Okamoto and Ohta [25] proposed a scheme which resembles our coupon system in the sense that they use a multiple blind signature to issue a "large" coin which is comprised of "smaller" coins, or rather, can be subdivided into smaller ones. However, subdividability in their system is considered a feature while in a coupon system this translates to splitting and, hence, is less desirable. In [12] and [31], Chen and Verheul, respectively, proposed credential systems where the credentials are multi-showable, i.e., can be shown for an unlimited number of times. The credentials obtained through both systems are intended for pseudonymous usage, thus, our requirements for unlinkable redemptions and $m$-redeemability are not satisfied.

In the work of Brands [3], attribute certificates were proposed that allow selective showing of individual attributes. These attributes are normally multi-showable but can be made $m$-showable, however, then different transactions become linkable. Persiano and Visconti [27] used some of the ideas of [3] to build a credential system which is multi-showable and does not allow to link different showings of a credential. Still, showings of credentials cannot be limited.

An anonymous credential system where credentials can be made one-showable was proposed by Camenisch and Lysyanskaya [8]. Through this system, a user may obtain $m$ one-show credentials which can be regarded as single coupons. However, this approach is not very efficient when used in a coupon system, since a credential generation protocol must be run for each credential and the credentials can be shared by different users and independently shown[2]. This means, when applied as a coupon system, coupons can be independently spent and splitting is easily possible.

The aspect of technically supporting loyalty in commercial applications has also been explored before. Maher [22] proposed a framework to introduce loyalty points, however, the privacy aspect was not an issue there. Enzmann et al. [15] proposed a counter for a privacy-friendly, point-based loyalty system, where users anonymously obtained points for their purchases. Finally, Wibowo et al. [32] proposed a loyalty system, however, based on pseudonyms, thus, providing a weaker form of privacy.

## 3 Model

The coupon system considered here involves mainly two parties, a customer $\mathcal{U}$ (user) and a vendor $\mathcal{V}$. The system itself is comprised of an *issue* protocol and a *redeem* protocol which both are carried out between $\mathcal{U}$ and $\mathcal{V}$. The output of the issue protocol is a multi-coupon $M$ for $\mathcal{U}$ and the result of the redeem protocol is a spent single coupon for $\mathcal{V}$ and a multi-coupon devalued by one single coupon for $\mathcal{U}$. Next, we state the main security requirements for the involved parties.

### 3.1 Requirements

In the following, we will use the notation $M \rightsquigarrow N$ to indicate that multi-coupon $N$ is a successor of multi-coupon $M$. That is, $N$ has strictly less coupons left to spent than $M$ and both originate from the same initial multi-coupon. Given two multi-coupons $M$ and $N$, if either $M \rightsquigarrow N$ or $N \rightsquigarrow M$ we say that $M$ and $N$ are *related.*

***Unforgeability.*** It must be infeasible to create new multi-coupons, to increase the number of unspent coupons, or to reset the number of spent coupons.

***Double-spending detection.*** A vendor must be able to detect attempts of redeeming 'old' coupons that have already been redeemed. This means, given two runs of the redeem protocol, where a single coupon $x$ is deducted from multi-coupon $M$ and $y$ is deducted from $N$, the vendor must be able to decide if $x = y$.

***Redemption limitation.*** An $m$-redeemable coupon $M$ may not be accepted by the vendor more than $m$ times.

***Protection against splitting.*** A coalition of customers $\mathcal{U}_i$ should not be able to split an $m$-redeemable multi-coupon $M$ into (disjoint) $s_i$-redeemable shares $M_i$ with $\sum_i s_i \leq m$ such that $M_i$ can only be redeemed by customer $\mathcal{U}_i$ and none of the other customers $\mathcal{U}_j, j \neq i$, or a subset of them is able to redeem the share $M_i$ or a part of it. We call this property *strong protection against splitting*.

A weaker form of this property is *all-or-nothing-sharing*[3]. This means that splitting is possible, however, only if customers trust each other not to spent (part of) the other's

[2] In [8] a solution was proposed to deal with this kind of lending. However, this solution hurts performance because it adds complexity and additional protocol runs to the basic scheme.

[3] This is similar to all-or-nothing-disclosure used in [3, 8] to prevent lending of credentials.

share $M_i$. Another way of putting this is to say that sharing $M$ means sharing all $m$ single coupons. We call this *weak protection against splitting*.

***Unlinkability.*** It must be infeasible for vendors to link protocol runs of honest users. For this, we have to consider linking a run of an issue protocol to runs of corresponding redeem protocols and linking of any two redeem protocol runs.

*(1) issue vs. redeem*: Given a run of the issue protocol with output a multi-coupon $M$ and given a redeem protocol run with output a devalued multi-coupon $N$, the vendor must not be able to decide if $M \rightsquigarrow N$.

*(2) redeem vs. redeem*: Given two runs of the redeem protocol with output two multi-coupons $M, N$, the vendor must not be able to decide if $M \rightsquigarrow N$ or $N \rightsquigarrow M$, i.e., he cannot tell if $M$ and $N$ are related or unrelated.

***Minimum Disclosure.*** As a result of a redeem protocol run, the vendor may only learn of the single coupon being redeemed but not the number of remaining coupons. This already follows from the unlinkability requirement but we make it explicit here, nevertheless.

# 4 Building Blocks

In order to illustrate our coupon system, we first introduce the employed technical building blocks.

Throughout the paper, we will use the following notational conventions. Let $n = pq$ where $p = 2p' + 1$, $q = 2q' + 1$ and $p', q', p, q$ are all primes. Denote the binary length of an integer $I$ by $\ell_I$. We require $\ell_p = \ell_n/2$. We denote the set of residues modulo $n$ that are relatively prime to $n$ by $\mathbb{Z}_n^*$ and the set of quadratic residues modulo $n$ by $QR_n$, i.e., for all $a \in QR_n$ there exists $b \in \mathbb{Z}_n^*$ such that $a \equiv b^2 \bmod n$. By $a \in_R S$ we mean that $a$ is chosen uniformly and at random from the set of integers $S$. For saving space, we omit the operation $\bmod n$ in the following specifications.

## 4.1 Commitment Scheme

A commitment scheme is a two-party protocol between a committer $\mathcal{C}$ and a receiver $\mathcal{R}$. In general, the scheme includes a *Commit* process and an *Open* process. In the first process, $\mathcal{C}$ computes a commitment $C_x$ with a message $x$, such that $x$ cannot be changed without changing $C_x$ [4]. $\mathcal{C}$ then gives $C_x$ to $\mathcal{R}$ and keeps $x$ secret. In the second process, $\mathcal{C}$ opens $C_x$ by revealing $x$.

The commitment scheme we employ is due to Damgård and Fujisaki (DF) [14] which is a generalization of the Fujisaki-Okamoto scheme [18]. We skip the basic DF scheme for committing to a single value $x$ and proceed to the scheme where the commitment is to a message tuple $(x_1, x_2, \ldots, x_m)$.

Let $\langle h \rangle$ denote the group generated by $h \in_R QR_n$, and let $g_1, g_2, \ldots, g_m \in \langle h \rangle$. On secret input $X := (x_1, x_2, \ldots, x_m)$, where $x_i \in [0, 2^{\ell_x})$, and public input $PK := (g_1, \ldots, g_m, h, n)$, the commitment is $C_X := \prod_{i=1}^m g_i^{x_i} h^{r_X}$, where $r_X \in_R \mathbb{Z}_n$ are chosen at random.

## 4.2 Signature Scheme

The signature scheme stated in the following is a variant of the Camenisch and Lysyanskaya (CL) signature scheme [9] for signing a block of messages which was used before in [5]. The signed message is denoted by a tuple $X := (x_1, x_2, \dots, x_m)$ where $x_i \in [0, 2^{\ell_x})$, $i = 1, \dots, m$ and $\ell_x$ is a parameter for the message length.

***Key Generation.*** Set the modulus $n$ as described before. Choose $a_1, a_2, \dots, a_m$, $b$, $c$ $\in_R QR_n$ and output a public key $PK := (A, b, c, n)$ where $A := (a_1, a_2, \dots, a_m)$ and a secret key $SK := (p, q, n)$.

***Signing.*** On input $X := (x_1, x_2, \dots, x_m)$, choose a random prime number $e \in_R [2^{\ell_e - 1}, 2^{\ell_e - 1} + 2^{\ell'_e - 1}]$ and a random number $s$ of length $\ell_s$, where $\ell'_e$ is the length of the interval that the $e$ values are chosen from, $\ell_e$ is the length of the $e$ values, $\ell_s$ is the length of the $s$ value. Both the values $\ell_e$ and $\ell_s$ are dependent on a security parameter $\ell_\phi$, for the details see [5]. The resulting signature is the tuple $(v, e, s)$ such that $c \equiv v^e a_1^{x_1} \cdots a_m^{x_m} b^s$. We denote this algorithm by: $(v, e, s) \leftarrow \mathit{Sign}_{(A,b,c,n,p)}(X)$.

***Verification.*** In order to verify that $(v, e, s)$ is a signature on $X := (x_1, x_2, \dots, x_m)$, check that $c \equiv v^e a_1^{x_1} \cdots a_m^{x_m} b^s$ and also that $x_i \in [0, 2^{\ell_x})$, $i = 1, \dots, m$, and $2^{\ell_e - 1} \geq e \geq 2^{\ell_e - 1} + 2^{\ell'_e - 1}$. We denote this algorithm by: $\mathit{ind} \leftarrow \mathit{Verify}_{(A,b,c,n)}(X, v, e, s)$, where $\mathit{ind} \in \{\mathit{accept}, \mathit{reject}\}$.

***Remark 1.*** The CL signature scheme is separable, i.e., the signature $(v, e, s)$ on $X$ is also the signature on a sub-tuple of $X$ if we change the public key accordingly. In the following, we use the notation $X \backslash (x_j)$ to denote the sub-tuple of $X$ which is comprised of all of $X$'s components but its $j$-th one, i.e., $X \backslash (x_j) = (x_1, \dots, x_{j-1}, x_{j+1}, \dots, x_m)$. Now, the signature on $X$ under the public key $(A, b, c, n)$ is the same as the signature on $X \backslash (x_j)$ under the public key $(A \backslash (a_j), b, c/a_j^{x_j}, n)$, i.e., $\mathit{Sign}_{(A,b,c,n,p)}(X) = (v, e, s) = \mathit{Sign}_{(A \backslash (a_j), b, c/a_j^{x_j}, n, p)}(X \setminus (x_j))$. This holds for any sub-tuple $Y$ of $X$. We will use this property in our coupon system to redeem a single coupon from a multiple set of coupons.

***Remark 2.*** As discovered in [7], the CL signature scheme has the property of randomisation, i.e., the signature $(v, e, s)$ can be randomised to $(T = vb^w, e, s^* = s - ew)$ with an arbitrary $w$. From a verifier's point of view, $(T, e, s^*)$ and $(v, e, s)$ are equivalent since they both are signatures on $X$. This property benefits our scheme because a proof of knowledge of $(T, e, s^*)$ can be done more efficiently than proving knowledge of $(v, e, s)$ in an anonymous manner.

## 4.3 Proofs of Relations Between Committed Numbers

For the construction of our coupon system, we need several sub-protocols in order to prove certain relations between committed numbers [18, 10, 1, 13, 28]. These are proofs of knowledge ($\mathit{PoK}$) where the commitments are formed using the DF scheme. We will now briefly state the various protocols that we employ. The output of each of the protocols for the verifier is $\mathit{ind}_{\mathcal{V}} \in \{\mathit{accept}, \mathit{reject}\}$.

***PoKRep.*** A prover $\mathcal{P}$ proves knowledge of a discrete logarithm representation (DL-Rep) modulo a composite to a verifier $\mathcal{V}$. Common inputs are a description of group $\mathcal{G}$, $PK := (g_1, \ldots, g_m, h)$ with $h, g_i \in \mathcal{G}$, and a commitment $C$. By this protocol, $\mathcal{P}$ convinces $\mathcal{V}$ of knowledge of $X := (x_1, \ldots, x_m)$, such that $C = \prod_{i=1}^{m} g_i^{x_i} h^r$.

***PoKEqRep.*** A prover $\mathcal{P}$ proves to a verifier $\mathcal{V}$ knowledge of equality of representations of elements from possibly different groups $\mathcal{G}_1, \mathcal{G}_2$. Common inputs are $PK_1 := (g_1, \ldots, g_m, h)$, $g_i, h \in \mathcal{G}_1$, $PK_2 := (g_1', \ldots, g_m', h')$, $g_i', h' \in \mathcal{G}_2$, commitments $C_1 \in \mathcal{G}_1$ and $C_2 \in \mathcal{G}_2$. By running the protocol, $\mathcal{P}$ convinces $\mathcal{V}$ of knowledge of $X := (x_1, \ldots, x_m)$ such that $C_1 = \prod_{i=1}^{m} g_i^{x_i} h^{r_1}$ and $C_2 = \prod_{i=1}^{m} g_i'^{x_i} h'^{r_2}$, i.e., $log_{g_i}(g_i^{x_i}) = log_{g_i'}(g_i'^{x_i})$ $(i = 1, \ldots, m)$.

***PoKInt.*** A prover $\mathcal{P}$ proves to a verifier $\mathcal{V}$ knowledge of $x$ and $r$ such that $C = g^x h^r$ and $a \leq x \leq b$. Common inputs are parameters $(g, h, n)$, the commitment $C$, and the integers $a, b$. We use a straightforward extension to the basic protocol, such that the proved knowledge is two tuples, instead of two values, and the interval membership of each component from a tuple, instead of one value. Within this extension, we denote $G := (g_1, g_2, ..., g_m)$, $H := (h_1, h_2, ..., h_l)$, $X := (x_1, x_2, ..., x_m)$, $R := (r_1, r_2, ..., r_l)$, and $C := \prod_{i=1}^{m} g_i^{x_i} \prod_{j=1}^{l} h_j^{r_j}$. By running the protocol, $\mathcal{P}$ proves to $\mathcal{V}$ knowledge of $X$ and $R$, and the interval membership, $a \leq x_i \leq b$.

***PoKOr.*** A prover $\mathcal{P}$ proves to a verifier $\mathcal{V}$ an OR statement of a commitment $C$, such that $C := (C_1, \ldots, C_m)$, where $C_i := \prod_{j \in \alpha_i} g_j^{x_{ij}} h^{r_i}$ and $\alpha_i \subseteq \{1, ..., m\}$, and $\mathcal{P}$ knows at least one tuple $\{x_{ij} \mid j \in \alpha_i\}$ for some undisclosed $i$. We denote the OR statement as $\bigvee_{i=1}^{m} C_i = \prod_{j \in \alpha_i} g_j^{x_{ij}} h^{r_i}$. Common inputs are $C$ and parameters $(G, n)$ where $G := (g_1, \ldots, g_m)$. By running the protocol, $\mathcal{P}$ proves to $\mathcal{V}$ knowledge of $\{x_{ij} \mid j \in \alpha_i\}$ without revealing the values $x_{ij}$ and $i$. A number of mechanisms for proving the "OR" statement have been given in [6, 13, 28].

***PoK.*** Sometimes, we need to carry out two or more of the above protocols simultaneously, e.g., when responses to challenges have to be used in more than one validity check of the verifier to prove intermingled relations among commitments. Instead of giving concrete constructions of these protocols each time, we just describe their aim, i.e., what the verifier wants to prove. For this we apply the notation used, e.g., in [10]. For instance, the following expression

$$ind_{\mathcal{V}} \leftarrow PoK\{(\alpha, \beta) : C = g^{\alpha} h^{\beta} \ \wedge \ D = \hat{g}^{\alpha} \hat{h}^{\beta} \ \wedge \ 0 \leq \alpha < 2^k\}$$

means that knowledge of $\alpha$ and $\beta$ is proven such that $C = g^{\alpha} h^{\rho}$ and $D = \hat{g}^{\alpha} \hat{h}^{\beta}$ holds, and $\alpha$ lies in the integer interval $[0, 2^k)$.

## 4.4 Blind Signatures and Signature Proof

***BlindSign.*** Next we state a secure protocol for signing a blinded tuple, shown in Figure 1, which is based on [9, 5]. In this protocol, a user $\mathcal{U}$ obtains a signature from the signer $\mathcal{S}$ on a tuple $X := (x_1, x_2, \ldots, x_m)$ without revealing $X$ to $\mathcal{S}$. We assume that $\mathcal{S}$ has the public key $PK := (A, b, c, n)$, the secret key $SK := p$, and public length

| **User** $\mathcal{U}$ | | **Signer** $\mathcal{S}$ |
|---|---|---|
| **Common Input:** | Verification key $PK := (A, b, c, n), A := (a_1, \dots, a_m)$<br>Length parameters $\ell_x, \ell_e, \ell'_e, \ell_s, \ell_n, \ell_\phi$ | |
| **User's Input:** | Message $X := (x_1, \dots, x_m)$ | |
| **Signer's Input:** | Factorisation of $n$ : $(p, q, n)$ | |
| choose $s' \in_R \{0,1\}^{\ell_n+\ell_\phi}$ | | |
| compute $D := \prod_{i=1}^{m} a_i^{x_i} b^{s'}$ | $\xrightarrow{D}$ | |
| | Run $PoK\ \{\ (\xi_1, \dots, \xi_m, \sigma) : D = \pm a_1^{\xi_1} \cdots a_m^{\xi_m} b^{\sigma}\ \wedge$<br>for $i = 1, \dots, m : \xi_i \in \{0,1\}^{\ell_x+\ell_\phi+2}\ \wedge$<br>$\sigma \in \{0,1\}^{\ell_n+\ell_\phi+2}\} \rightarrow ind_{\mathcal{S}}$ | |
| | | check $ind_{\mathcal{S}} \overset{?}{=} accept$<br>choose $\hat{s} \in_R \{0,1\}^{\ell_s-1}$<br>compute $s'' := \hat{s} + 2^{\ell_s-1}$<br>choose $e \in_R (2^{\ell_e-1}, 2^{\ell_e-1} + 2^{\ell'_e-1})$ |
| compute $s := s' + s''$; | $\xleftarrow{(v,e,s'')}$ | compute $v := (c/(Db^{s''}))^{1/e}$ |
| check $Verify_{(A,b,n)}(X, v, e, s) \overset{?}{=} accept$<br>[i.e., $c = v^e b^s \prod_{i=1}^{m} a_i^{x_i}$ ]<br>output $(v, e, s)$ | | |

**Fig. 1.** Protocol for blindly signing a tuple: *BlindSign*

parameters $\ell_n, \ell_x, \ell_e, \ell'_e, \ell_s$, and $\ell_\phi$ which are parameters controlling the statistical zero-knowledge property of the employed $PoK$. $\mathcal{U}$'s input to the protocol is the message $X := (x_1, \dots, x_m)$ for which $\mathcal{U}$ wants to obtain a signature.

Among the first steps, $\mathcal{U}$ computes the value $D := \prod_{i=1}^{m} a_i^{x_i} b^{s'}$ and sends it to $\mathcal{S}$. The next steps assure to $\mathcal{S}$ that $\mathcal{U}$ indeed knows the discrete logarithms of $D$ with respect to the basis $(a_1, \dots, a_m, b)$ respectively, and the interval of the committed values in $D$ are selected correctly.

If all proofs are accepted, $\mathcal{S}$ chooses a prime $e$ and computes $v := (c/(Db^{s''}))^{1/e} = (c/(\prod_{i=1}^{m} a_i^{x_i} b^{(s'+s'')}))^{1/e}$. At the end, $\mathcal{V}$ sends the resulting tuple $(v, e, s'')$ to $\mathcal{U}$. Finally, $\mathcal{U}$ sets $s := (s' + s'')$ and obtains $(v, e, s)$ as the desired signature on $X$. We denote this protocol for blindly signing a tuple by $(v, e, s) \leftarrow BlindSign_{(PK)}(X)$.

***PoKSign.*** The next protocol, shown in Figure 2, is a zero-knowledge proof of a signature created in the *BlindSign* protocol. The idea of this protocol is to convince a verifier $\mathcal{V}$ that a prover $\mathcal{P}$ holds a valid signature $(v, e, s)$ on $X$ satisfying $c \equiv v^e a_1^{x_1} \cdots a_m^{x_m} b^s$ without $\mathcal{V}$ learning anything of the signature but its validity. The common inputs are the same as in the *BlindSign* protocol. $\mathcal{P}$'s secret input is the message $X$ and the corresponding signature $(v, e, s)$.

The protocol works as follows: $\mathcal{P}$ first randomises the signature components, $v$ and $s$, by choosing $w$ at random and computing $T := vb^w$ and $s^* = s - ew$. $\mathcal{P}$ sends only $T$ to $\mathcal{V}$. Then, $\mathcal{P}$ proves to $\mathcal{V}$ his knowledge specified in $PoK$.

As discussed in Remark 2 of Section 4.2, in $\mathcal{V}$'s view, $(T, e, s^*)$ is a valid signature on $X$, as is $(v, e, s)$. The difference between them is that we are allowed to reveal the

**Prover $\mathcal{P}$** **Verifier $\mathcal{V}$**

**Common Input:** Verification key $PK := (A, b, c, n), A = (a_1, a_2, \ldots, a_m)$

Length parameters $\ell_x, \ell_e, \ell'_e, \ell_\phi$

**Prover's Input:** Message $X := (x_1, \ldots, x_m)$, Signature $(v, e, s)$

choose $w \in_R \{0,1\}^{\ell_n + \ell_\phi}$

compute $T := vb^w$; $\xrightarrow{T}$

$$\text{Run } PoK \ \{ \ (\xi_1, \ldots, \xi_m, \sigma, \epsilon) : c = \pm T^\epsilon a_1^{\xi_1} \cdots a_m^{\xi_m} b^\sigma \ \wedge$$

$$\text{for } i = 1, \ldots, m : \xi_i \in \{0,1\}^{\ell_x + \ell_\phi + 2} \ \wedge$$

$$(\epsilon - 2^{\ell_e}) \in \{0,1\}^{\ell'_e + \ell_\phi + 1}\} \rightarrow ind_\mathcal{V}$$

check $ind_\mathcal{V} \overset{?}{=} accept$

**Fig. 2.** Protocol for proving knowledge of a signature: *PoKSign*

value $T$ to $\mathcal{V}$, but not the value $v$, because $T$ is different in every proof. Therefore, to prove the signature with $c \equiv v^e \prod_{i=1}^m a_i^{x_i} b^s$ becomes one with $c \equiv T^e \prod_{i=1}^m a_i^{x_i} b^{s^*}$. Clearly, to prove the second equation is much simpler then the first one. $PoK$ here performs the following three simple proofs in one go: (1) $PoKRep$: to prove knowledge of discrete logarithms of $c$ ($\equiv T^e \prod_{i=1}^m a_i^{x_i} b^{s^*}$) with respect to the basis $(T, a_1, \ldots, a_m, b)$ respectively; (2) $PoKInt$: to prove the values $x_1, \ldots, x_m$ are within a right bound, i.e., for $i = 1, \ldots, m : x_i \in \{0,1\}^{\ell_x + \ell_\phi + 2}$; (3) $PoKInt$: to prove the value $e$ is also within a right bound, i.e., $(e - 2^{\ell_e}) \in \{0,1\}^{\ell'_e + \ell_\phi + 1}$.

## 5 Construction of the Coupon System

In this section we propose a concrete scheme for a coupon system that allows issuance and redemption of multi-coupons. The scheme is comprised of two protocols, *Issue* and *Redeem*, which are carried out between a user $\mathcal{U}$ and a vendor $\mathcal{V}$, and an Initialisation algorithm.

***Initialisation.*** $\mathcal{V}$ initialises the system by generating a key pair $PK = (A, b, c, n)$ where $A = (a_1, a_2, \ldots, a_m)$ and $SK = (p, q, n)$. The vendor keeps $SK$ secret and publishes $PK$ with length parameters $\ell_x$, $\ell_e$, $\ell'_e$, $\ell_n$, $\ell_s$, and the security parameter $\ell_\phi$.

***Issue.*** In the issue protocol, $\mathcal{U}$ chooses serial numbers $x_i \in_R \{1, \ldots, 2^{\ell_x} - 1\}$ $(i = 1, \ldots, m)$ and sets $X := (x_1, \ldots, x_m)$. Then $\mathcal{U}$ runs $(v, e, s) \leftarrow \mathit{BlindSign}_{(PK)}(X)$ with $\mathcal{V}$ to obtain a blind CL signature $(v, e, s)$ on $X$. The tuple $M := (X, v, e, s)$ will act as the user's multi-coupon.

***Redeem.*** In the redeem protocol, $\mathcal{U}$ (randomly) chooses an unspent coupon $x_j$ from the tuple X, sets $x := x_j$ and $X' := X \setminus (x)$. The value $x$ then becomes a common input to the redeem protocol. Next $\mathcal{U}$ proves to $\mathcal{V}$ that she is in possession of a valid multi-coupon ($\mathcal{V}$'s signature on $X$) containing $x$ without revealing the signature itself.

In addition, we have the restriction that the index information of $x$, i.e. $j$, must not be disclosed to $\mathcal{V}$ when proving that $x$ is part of the signed message in the CL signature.

If $\mathcal{V}$ would learn the index, he would be able to break the unlinkability property. To see this, simply suppose that two different coupons $x$ and $y$ are revealed both with respect to the base $a_j$ from the CL signature. In this case, $\mathcal{V}$ immediately learns that the corresponding multi-coupons are different, since clearly $(z_1, z_2, \ldots, z_{j-1}, x, z_{j+1}, \ldots, z_m) \neq (z'_1, z'_2, \ldots, z'_{j-1}, y, z'_{j+1}, \ldots, z'_m)$, where the $z_i$ and $z'_i$ are the other single coupons from the multi-coupon of $x$ and $y$, respectively. So in fact, by revealing the index $j$, it is proven that $x$ is the $j$-th component of an *ordered* set of messages $(x_1, x_2, \ldots, x_m)$, where the $x_i$, with $i \neq j$, are unknown to $\mathcal{V}$. However, in order to retain unlinkability the index must not be disclosed and we need to prove that $x$ is contained in an *unordered* set of messages, i.e., $x \in \{x_1, x_2, \ldots, x_m\}$ where the $x_i$, $i \neq j$, are unknown to $\mathcal{V}$.

Note that proving that $x$ is the $j$-th message of the signed message tuple $X$ could be easily done by using Remark 1 from section 4.2: $\mathcal{V}$ computes a modified public key $PK_j := (A\backslash(a_j), b, c/a_j^x)$ and $\mathcal{U}$ runs the protocol *PoKSign* with respect to $PK_j$, $X'$, and $(v, e, s)$. This way the signature $(v, e, s)$ and $X'$ would still be kept from $\mathcal{V}$, though the index $j$ would be disclosed by the public key $PK_j$.

In order to overcome this index problem, $\mathcal{U}$ runs an extended version of the *PoKSign* protocol from Figure 2 which proves that $x$ is part of the signature but does not disclose the index of the spent coupon. The idea for this extension is as follows. Instead of disclosing to $\mathcal{V}$ which concrete public key $PK_i$ $(i = 1, \ldots, m)$ is to be used for the verification, $\mathcal{U}$ proves that one of the public keys $PK_i$ is the verification key with respect to the signature $(v, e, s)$ on the message $X'$. For this, the proof *PoK* is extended with the *PoKOr* protocol which adds the proof for the term $\bigvee_{i=1}^{m} C_i = T^{\epsilon} \prod_{l \in \{1,\ldots,m\}, l \neq i} a_l^{\xi_l} b^{\sigma}$ to *PoK* (see also Section 4.3). —Note that the terms $C_i = c/a_i^x$ are computed by both $\mathcal{U}$ and $\mathcal{V}$.— Using *PoKOr*, $\mathcal{U}$ proves that she knows the DLRep of one of the $C_i$ with respect to its corresponding basis $(T, A\backslash(a_i), b)$ without revealing which one — since $x$ is equal to $x_j$, the commitment $C_j = c/a_j^x$ must have a representation to the basis $(T, A\backslash(a_j), b)$ which is known to $\mathcal{U}$. Also note that this proves knowledge of the signature $(T = vb^w, e, s^* = s - ew)$ with respect to the public key $PK_j := (A\backslash(a_j), b, c/a_j^x)$. This is according to Remark 1 the same as proving it with respect to the public key $(A, b, c)$ and, by Remark 2, the randomised signature $(T, e, s^*)$ is, from $\mathcal{V}$'s point of view, equivalent to the signature $(v, e, s)$. Hence, $x$ must be part of a valid multi-coupon, i.e., a component from a message tuple that was signed by the vendor.

Eventually, if the proof protocol yields $accept$ then $\mathcal{V}$ is convinced that $\mathcal{U}$ owns a signature on an $m$-tuple $X$ which contains $x$. At last $\mathcal{V}$ checks if $x$ is already stored in his database of redeemed coupons. If it is not, $\mathcal{V}$ accepts $x$ as a valid coupon and will grant the service.

### 5.1 Properties

In the following, we sketch how the coupon system proposed in the previous subsection satisfies the requirements from section 3.1. We will analyse the security of the system assuming that the strong RSA assumption holds.

***Unforgeability.*** The property of unforgeability of our coupon system follows from the unforgeability of the CL signature scheme. As described in the previous section, a set of multi-coupons is a single CL signature on a block of messages. As has been proven in [9], forging CL signatures would break the strong RSA assumption.

Resetting the number of spent coupons requires to change some component in the tuple $X$, e.g., replacing a redeemed coupon $x_i$ with $x_i^* \neq x_i$, since the vendor stores each spent single coupon $x_i$. However, replacing $x_i$ by $x_i^*$, yielding tuple $X^*$, must be done such that $Sign_{(\cdot)}(X) = Sign_{(\cdot)}(X^*)$. Suppose the latter can be done. Then, we get $v^e \prod_{i=1}^{m} a_i^{x_i} b^s \equiv v^e \prod_{j=1}^{i-1} a_j^{x_j} a_i^{x_i^*} \prod_{j=i+1}^{m} a_j^{x_j} b^s$. Dividing by the right hand side yields $a_i^{x_i - x_i^*} \equiv 1 \bmod n$. Since $x_i \neq x_i^*$ it must be the case that $x_i - x_i^* = \alpha \cdot ord(\mathbb{Z}_n)$.

Now, choose any $e$ such that $1 < e < (x_i - x_i^*)$ and $gcd(e, x_i - x_i^*) = 1$. By the extended Euclidean algorithm we can find $d$ such that $ed + (x_i - x_i^*)t = 1$. Using this, we can compute $e$-th roots in $\mathbb{Z}_n$. For this, let $u$ be any value from $\mathbb{Z}_n^*$ and compute $w := u^d$. Since $u \equiv u^{ed+(x_i - x_i^*)t} \equiv u^{ed} u^{\alpha \cdot ord(\mathbb{Z}_n) \cdot t} \equiv (u^d)^e \equiv w^e \bmod n$, the value $w$ is an $e$-th root of $u$. This means we would have found a way to break the strong RSA assumption. Since this is assumed to be infeasible $x_i$ cannot be replaced by $x_i^* \neq x_i$ without changing the signature $(v, e, s)$.

***Double-spending detection.*** If a cheating user tries to redeem an already spent single coupon $x_i$, she will be caught at the end of the redeem protocol, since the coupon to be redeemed must be disclosed and, thus, can easily be looked up in the vendor's database.

***Redemption limitation.*** An $m$-redeemable coupon $M$ cannot be redeemed more than $m$ times (without the vendor's consent). Each multi-coupon $M$ contains a signature on an $m$-tuple $(x_1, \ldots, x_m)$ of single coupons and in each run of the issue protocol a single coupon $x_i$ is disclosed. Thus, after $m$ honest runs using the same $M$, all $x_i$ will be disclosed to the vendor. As argued under *unforgeability* and *double-spending detection*, already redeemed $x_i$ cannot be replaced by fresh $x_i^*$ and any attempt to 'reuse' an already disclosed $x_i$ will be caught by the double-spending check.

***Weak protection against splitting.*** Suppose that two users $\mathcal{U}_1$ and $\mathcal{U}_2$ want to share multi-coupon $M := (X, v, e, s)$ such that $\mathcal{U}_1$ receives single coupons $X_1 := \{x_1, \ldots, x_i\}$ and $\mathcal{U}_2$ receives the remaining coupons $X_2 := \{x_j, \ldots, x_m\}$, $i < j$. To achieve splitting, they have to find a way to make sure that $\mathcal{U}_1$ is able to redeem all $x_j \in X_1$ while not being able to redeem any coupon $x_j' \in X_2$, and analogously for $\mathcal{U}_2$. However, in the redeem protocol it is necessary to prove knowledge of the DLRep of $C_X$, which is $X$. Since proving knowledge of $X$ while knowing only $X_1$ or $X_2$ would violate the soundness of the employed proof of knowledge $PoKRep$, and hence violate the strong RSA assumption, this is believed to be infeasible. Again, the missing part of $X$, either $X_1$ or $X_2$, cannot be replaced by 'fake' coupons $X_{1|2}'$ since this violates the unforgeability property of t he coupon system. Hence, $X$ cannot be split and can only be shared if both $\mathcal{U}_1$ and $\mathcal{U}_2$ have full knowledge of $X$ which comes down to *all-or-nothing sharing*.

***Unlinkability.*** For unlinkability, we have to consider two cases, unlinkability between issue and redeem protocol runs and between executions of the issue protocol.

(1) *issue vs. redeem*: The issue protocol is identical to the protocol *BlindSign* and, hence, the vendor $\mathcal{V}$, acting as signer, does not learn anything about the message $X$

being signed. However, $\mathcal{V}$ has partial knowledge of the signature $(v, e, s)$, i.e., at the end of the issue protocol since he knows $(v, e)$ but not $s$. To exploit this knowledge, he would have to recognize $v$ or $e$ in a run of the redeem protocol. In the redeem protocol, however, the user only sends commitments $T$ and $C_i$ $(i = 1, \ldots, m)$ to $\mathcal{V}$. Since the commitment scheme is statistically hiding the vendor cannot learn anything about $v$ or $e$ from the commitments.

Furthermore, all sub-protocols in $Redeem$ operate only on the commitments $T$ and $C_i$ and since the opening information of either $T$ or any $C_i$ is never given to $\mathcal{V}$ during the execution of any of these proof protocols the vendor is unable to recognise $v$ or $e$.

(2) *redeem vs. redeem*: The redeem protocol mainly consists of the $PoKSign$ protocol which only employs zero-knowledge proofs and statistically hiding commitments and, hence, is unlinkable. However, in the redeem protocol the coupon value $x$ is given to the vendor $\mathcal{V}$, i.e., the verifier. In the following we sketch that $\mathcal{V}$ cannot use this information to infer other information that helps him to link redemptions of single coupons?

To see this, let $\tau$ be the transcript of the redeem protocol where $x$ is released. Since all proofs of knowledge applied in the redeem protocol perform challenge-response protocols, there will be some values $u$, containing $x$, which were formed by the user in response to challenges $t$ chosen by $\mathcal{V}$. The general form of a response is $u = ty + r$, where $t$ is $\mathcal{V}$'s challenge, $y$ is some committed value of which knowledge is proven, and $r$ is a witness randomly chosen by the user. However, since $x$'s index is not revealed (due to $PoKOr$) every response $u_i$ $(i = 1, \ldots, m)$ is equally likely to contain $x$.

Now, if $\mathcal{V}$ guesses $x$'s index, say $j$, he would only learn the value $r_j$ from the response $u_j$. However, this reveals no information of any other response $u_i$, $i \neq j$, from $\tau$, since for any value $x_i$, contained in the response $u_i$, the witness $r_i$ is randomly and uniformly chosen anew for each $u_i$. Hence, from $\mathcal{V}$'s point of view $u_i$ is a random value and may contain any value $x'_i$ and, thus, $x_i$ is (still) statistically hidden in $u_i$.

The consequence of the arguments mentioned above is that given any two redeem protocol transcripts $\tau^{(x)}$ and $\tau^{(y)}$ where $x$ and $y$ are deducted from (hidden) multi-coupons $M^{(x)}$ and $N^{(y)}$, respectively, the verifier cannot determine whether $x$ is hidden in any response $u_i^{(y)}$ from $\tau^{(y)}$ and analogously if $y$ is hidden in any $u_l^{(x)}$ from $\tau^{(x)}$. This means the vendor cannot decide with a non-negligible probability better than pure guessing if the multi-coupons $M^{(x)}$ and $N^{(y)}$ are related or unrelated.

***Minimum Disclosure.*** A further consequence of the unlinkability of transactions in the coupon system, and due to the fact that no counter value is sent in any protocol, the number of unspent coupons cannot be inferred from any redeem protocol run.

# 6 Conclusion

The coupon system presented in this work allows vendors to issue multi-coupons to their customers, where each single coupon of such a multi-coupon can be redeemed at the vendor's in exchange for some good, e.g., an MP3 file, or some service, e.g., access to commercial online articles of a newspaper. Issuing coupons is advantageous to vendors since coupons effectively retain customers as long as they have coupons left to spent. However, multi-coupons might be misused by vendors to link transactions of customers in order to collect and compile information from their transactions in a profile. To protect the privacy of customers in this respect, the coupon system that we proposed allows customers to unlinkably redeem single coupons while preserving security requirements of vendors.

## Acknowledgement

The authors are grateful to the anonymous reviewers who provided valuable comments which greatly helped to improve the quality of this paper.

## References

1. F. Boudot. Efficient proofs that a commited number lies in an interval. *Adv. in Cryptology - EUROCRYPT 2000*, LNCS 1807. Springer, 2000.
2. S. Brands. An efficient off-line electronic cash system based on the representation problem. CWI Report, CS-R9323, Centrum voor Wiskunde en Informatica (CWI), 1993.
3. S. Brands. *Rethinking Public Key Infrastructure and Digital Certificates – Building in Privacy*. PhD thesis, Eindhoven Institute of Technology, The Netherlands, 1999.
4. G. Brassard, D. Chaum, C. Crepéau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37, 1988.
5. E. Brickell, J. Camenisch, L. Chen. Direct anonymous attestation. *Proc. 11th ACM Conference on Computer and Communications Security*, pages 132-145, ACM press, 2004.
6. J. Camenisch. *Group Signature Schemes and Payment Systems Based on the Discrete Logarithm Problem*. PhD thesis, ETH Zürich, Switzerland, 1998.
7. J. Camenisch, J. Groth. Group signatures: better efficiency and new theoretical aspects. *Forth Int. Conf. on Security in Communication Networks – SCN 2004*, LNCS 3352, Springer, 2005.
8. J. Camenisch, A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. *EUROCRYPT '01*, LNCS 2045. Springer, 2001.
9. J. Camenisch, A. Lysyanskaya. A signature scheme with efficient protocols. *Third Conference on Security in Communication Networks – SCN'02*, LNCS 2576. Springer, 2002.
10. J. Camenisch, M. Michels. Separability and efficiency for generic group signature schemes. *Adv. in Cryptology - CRYPTO '99*, LNCS 1666. Springer Verlag, 1999.
11. D. Chaum. Privacy protected payments: Unconditional payer and/or payee untraceability. *Smart Card 2000, Proceedings*. North Holland, 1989.
12. L. Chen. Access with pseudonyms. *Cryptography: Policy and Algorithms, International Conference, Brisbane, Australia, July, 1995, Proceedings*, LNCS 1029. Springer, 1996.
13. R. Cramer, I. Damgård, B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. *Adv. in Cryptology - CRYPTO '94*, LNCS 839. Springer, 1994.
14. I. Damgård, E. Fujisaki. A statistically hiding integer commitment scheme based on groups with hidden order. *Adv. in Cryptology - ASIACRYPT '02*, LNCS 2501. Springer, 2002.
15. M. Enzmann, M. Fischlin, M. Schneider. A privacy-friendly loyalty system based on discrete logarithms over elliptic curves. *Financial Cryptography*, LNCS 3110, Feb. 2004.
16. F. Feinberg, A. Krishna, Z. Zhang. Do we care what others get? A behaviorist approach to targeted promotions. *Journal of Marketing Research*, 39(3), Aug. 2002.
17. N. Ferguson. Extensions of single term coins. *Adv. in Cryptology - CRYPTO '93*, LNCS 773. Springer, 1993.
18. E. Fujisaki, T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. *Adv. in Cryptology - CRYPTO '97*, LNCS 1294. Springer, 1997.

19. D. Hoffman, T. Novak, M. Peralta. Building consumer trust online. *Communications of the ACM*, 42(4), Apr. 1999.
20. A. Kobsa. Tailoring privacy to users's needs. *User Modeling 2001 (UM 2001)*, LNAI 2109. Springer, 2001.
21. G. Macintosh, L. Lockshin. Retail relationships and store loyalty: A multi-level perspective. *International Journal of Research in Marketing*, 14(5), Dec. 1997.
22. D. Maher. A platform for privately defined currencies, loyalty credits, and play money. *Financial Cryptography*, LNCS 1465. Springer, 1998.
23. G. O'Connor, R. O'Keefe. The Internet as a new marketplace: Implications for consumer behaviour and marketing management. *Handbook on Electronic Commerce*. Springer, 2000.
24. A. Odlyzko. Privacy, Economics, and Price Discrimination on the Internet. *5th International Conference on Electronic Commerce (ICEC 2003)*. ACM Press, 2003.
25. T. Okamoto, K. Ohta. Disposable zero-knowledge authentications and their applications to untraceable electronic cash. *Adv. in Cryptology - CRYPTO '89*, LNCS 435. Springer, 1990.
26. R. Pear. Electronic cards replace coupons for food stamps. New York Times, June 23, 2004.
27. P. Persiano, I. Visconti. An efficient and usable multi-show non-transferable anonymous credential system. *Financial Cryptography*, LNCS 3110. Springer, Feb. 2004.
28. A. de Santis, G. di Crescenzo, G. Persiano, M. Yung. On monotone formula closure of SZK. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, November 1994.
29. G. Shaffer, Z. Zhang. Competitivec coupon marketing. *Marketing Science*, 14(4), 1995.
30. C. Shapiro, H. Varian. *Information Rules*. Harvard Business School Press, 1999.
31. E. Verheul. Self-blindable credential certificates from the weil pairing. *Adv. in Cryptology - ASIACRYPT '01*, LNCS 2248. Springer-Verlag, 2001.
32. A. Wibowo, K. Lam, G. Tan. Loyalty program scheme for anonymous payment systems. *Electronic Commerce and Web Technologies*, LNCS 1875. Springer, 2000.

# Testing Disjointness of Private Datasets

Aggelos Kiayias[1,⋆] and Antonina Mitrofanova[2]

[1] Computer Science and Engineering,
University of Connecticut Storrs, CT, USA
aggelos@cse.uconn.edu
[2] Computer Science, Rutgers University,
New Brunswick, NJ, USA
amitrofa@cs.rutgers.edu

**Abstract.** Two parties, say Alice and Bob, possess two sets of elements that belong to a universe of possible values and wish to test whether these sets are disjoint or not. In this paper we consider the above problem in the setting where Alice and Bob wish to disclose no information to each other about their sets beyond the single bit: "whether the intersection is empty or not." This problem has many applications in commercial settings where two mutually distrustful parties wish to decide with minimum possible disclosure whether there is any overlap between their private datasets. We present three protocols that solve the above problem that meet different efficiency and security objectives and data representation scenarios. Our protocols are based on Homomorphic encryption and in our security analysis, we consider the semi-honest setting as well as the malicious setting. Our most efficient construction for a large universe in terms of overall communication complexity uses a new encryption primitive that we introduce called "superposed encryption." We formalize this notion and provide a construction that may be of independent interest. For dealing with the malicious adversarial setting we take advantage of recent efficient constructions of Universally-Composable commitments based on verifiable encryption as well as zero-knowledge proofs of language membership.

## 1 Introduction

Suppose that Alice and Bob, wish to test whether their stock-portfolios share any common stocks. Nevertheless they are mutually distrustful and they are reluctant to reveal the contents of their portfolios to each other or to a third party. More generally, Alice and Bob possess two datasets drawn from a universe of publicly known values and wish to find out whether the intersection of their sets is empty or not with the minimum possible disclosure of information: only a single bit should become known, "whether the two datasets are disjoint or not." We call this a *private disjointness test*.

[⋆] Research partly supported by NSF CAREER Award CNS-0447808.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 109–124, 2005.
© Springer-Verlag Berlin Heidelberg 2005

A private disjointness test is a useful primitive in various online collaboration procedures. Indeed, depending on the disjointness of their two datasets, Alice and Bob may then proceed to different courses of action, e.g., in the case of a positive answer, Alice and Bob may seek further authorization and proceed to actually compute the intersection of their two datasets; on the other hand, in case of a negative outcome Alice and Bob may terminate their negotiation. The minimum disclosure property of a single bit output that we require is optimal from the point of view of decision-making based on private collections of data.

**Problem Formulation.** Let us formulate the problem that we put forth in more concrete terms: the two participants of a private disjointness test, Alice and Bob (and henceforth called $A$ and $B$) possess two subsets $S_A$ and $S_B$ of the universe $\Omega$; they wish to extract the bit "$S_A \bigcap S_B \overset{?}{=} \emptyset$" without disclosing any information about $S_A, S_B$ to each other (except the sizes of $S_A$ and $S_B$ that are publicly known). We will make the abstraction that $\Omega = \{1, \ldots, N\}$ as for a publicly known universe set $\Omega$ it is sufficient for the two parties to use the indices of the actual elements that are contained in their datasets instead of the elements themselves (using a predetermined ordering). Each dataset may be represented in two possible ways: either using its characteristic bitstring (and thus the two players in this case possess inputs of length $N$ bits) or using a list of indices listed explicitly i.e., the length of each player's input $N_X \log N$ where $N_X = \#S_X$ (the number of elements of the $X$ player's dataset) where $X \in \{A, B\}$.

A protocol solution for a private disjointness test will be called a Private Intersection Predicate Evaluation (PIPE) protocol. Given a correct PIPE protocol, we will be interested in various objectives such as (i) the level of security of each player against the other, (ii) minimization of the total communication and time complexity based on the input sizes of the two players, (iii) minimization of the number of rounds of interaction between the two players. Note that we will consider only PIPE protocols where one of the two players (selected to be $A$) has output. This is a standard assumption and consistent with a client-server type of interaction between the two parties. Naturally players may execute a PIPE protocol changing roles if output is desired in both sides.

**Our Results.** We present three PIPE protocols that satisfy different aspects of the objectives above; PIPE protocol #1 is suitable for settings where each player represents its input as a characteristic bitstring (this setting is suitable when the universe is not substantially larger than the size of the datasets); our protocol is very efficient and achieves communication complexity linear in $N$ in only a single round of interaction between the two players (we note that this communication is optimal for the general case of the disjointness problem, cf. the overview of related work in the following paragraph). The setting where datasets are substantially smaller compared to the universe size and one wishes to construct protocols that are sublinear in $N$ is more challenging. Our PIPE protocol #2 applies to this setting and operates efficiently with $N_A \times N_B$ total communication using $N_B$ rounds of interaction. Finally, our PIPE protocol #3, operating in the same setting as the second protocol, requires more communication, proportional to $\binom{N_A+N_B}{N_B}$ but reduces the number of rounds to a single round of interaction between the two players.

It should be stressed that protocols such as PIPE #2 and PIPE #3, reveal the size of the datasets of the players while PIPE #1 does not. This seems unavoidable if one wants to obtain sublinear communication. Nevertheless, it is straightforward that players need only reveal an *upper bound* on the number of elements of their dataset (this can be done by adjoining to the universe a sufficient amount of "dummy" elements different for each player that can be used for dataset padding). We defer further details for the full version.

We first describe our protocols in the so called semi-honest setting (parties are assumed to follow the protocol specifications) and then we consider their extension to the malicious setting (players may deviate arbitrarily from protocol specifications). We provide a concrete formulation of the malicious adversarial setting and efficient transformations of our PIPE protocols (of asymptotically the same complexity) to this setting, taking advantage of Universally-Composable commitments based on a recent verifiable encryption scheme. From a technical viewpoint we remark that, in our attempt to provide a communication efficient protocol for the setting where each party encodes his dataset as a list of values, we came up with a notion of public-key encryption (which we formalize and realize as part of the description of PIPE protocol #2) that is called **superposed encryption** and may have further applications in concrete secure function evaluation protocols.

**Related Work.** A private disjointness test falls into a general category of secure protocols that allow two parties to compare information they possess without leaking it. In turn, such protocols can be formulated and solved in terms of two-party Secure Function Evaluation [25] (see also [18]).

The special case of a private disjointness test where each party has a single element in the dataset has been studied extensively and is called a "private equality test." Protocols for private equality tests were considered in [13, 22, 20]. The problem of securely computing the *intersection* of two private datasets was considered in [22] and more recently in [15]. A related problem to a private disjointness test is scalar multiplication. A concurrent to the present work investigation of the notion of scalar products in the context of private data mining procedures appeared in [17].

The disjointness test problem itself was considered from the communication complexity perspective (without taking into account privacy) and a linear lower bound was shown [19, 23] even in the probabilistic setting (that allows an amount of error in the test). This suggests that there is no chance for a sublinear communication solution in the size of the universe in the worst case.

**PIPE Protocols Based on Existing Techniques.** As mentioned above, a private disjointness test can be formulated in the setting of secure function evaluation over a circuit. We note that protocol constructions based on secure circuit evaluation in the sense of [25] are not particularly efficient; nevertheless, they may be well within reach of current computational capabilities depending on the circuit as some recent results suggest [21]. Regarding building a private disjointness test over a circuit, we consider the two scenarios of input encoding: (i) input to each player is by the characteristic bitstring of the subset, (ii) input is by the subset as a list of elements. Using the formulation of secure two-party evaluation of Goldreich [18] and a security parameter $\ell$, we have the following: regarding (i), we can design a circuit for testing disjointness that uses $2N$ `AND` gates and $2N - 1$ `XOR` gates something that will result in a protocol with communication complexity $16N\ell + 2N$ bits (using $2N\ell$ oblivious transfers). Regarding (ii), (assuming subset sizes of $N_A$ and $N_B$ for the two players respectively) we can design a circuit that contains $N_A N_B(3\log_2(N) - 1))$ `XOR`-gates and $N_A N_B(\log_2(N) + 1)$ `AND`-gates that will produce a protocol of total communication $8N_A N_B \ell(\log_2(N) + 1) + N_A \log_2(N) + N_B \log_2(N)$ bits (using $N_A N_B(\log_2(N) + 1)$ oblivious transfers). The number of rounds required by the two protocols is $O(N)$ in the first case and $O(N_A N_B + \log N)$ in the second case. Evidently our PIPE constructions compare very favorably to generic secure function evaluation techniques – e.g. for case (i) our PIPE protocol #1 has total communication of $2(N+1)\ell$ bits and a single round of interaction, where for the case (ii) our PIPE protocol #2 has total communication of $3(N_B(N_A + 2)\ell$ bits with $N_B$ rounds of interaction; finally, our PIPE protocol #3 (applying to case (ii) as well) has a single round of interaction (at the cost of substantially larger communication though).

Beyond secure function evaluation techniques, the most related to the present work previous protocol constructions are those of [15]. This latter paper deals with the problem of computation of the actual intersection set of two private datasets; moreover the protocol construction of [15] (as noted in that paper) can also be easily modified to a protocol that reveals the size of the intersection only (not its elements); a private disjointness test nevertheless has a much more stringent security requirement (only one bit of information must be revealed — and perhaps an upper bound on the size of *both* datasets); for this reason it appears to be much more challenging to achieve. Another related problem to the intersection computation that is mentioned by [15] is "private threshold matching" that requires the computation of whether the intersection is larger than a specified threshold. Naturally a private disjointness test is a special case of this problem; nevertheless, no efficient protocol construction of a private disjointness test that is entirely independent from generic secure function evaluation techniques is known for this problem (cf. [15]).

Regarding our notion of superposed encryption we remark that it can be paralleled w.r.t. functionality to a $(2, 2)$ threshold homomorphic encryption with the main difference being in that in a superposed scheme key-generation is executed locally without communication and independently assuming fixed public-parameters. Concurrently and independently to the present work applications of $(2, 2)$-threshold-homomorphic encryption in two-party secure computations were considered in [24].

**Organization.** In section 2 we present the cryptographic primitives that are used in our constructions. In section 3 we present our three private intersection evaluation protocols as well as the notion of superposed encryption. Finally, in section 4 we consider the malicious adversary setting.

Due to lack of space we have omitted the proofs of our theorems from this extended abstract. They will appear in the full version of this work.

## 2 Preliminaries

**Homomorphic Encryption.** An encryption scheme is a triple $\langle K, E, D\rangle$ of algorithms defined as follows: the key generation algorithm $K$ on input $1^\ell$ (where $\ell$ is the key length) outputs a public key $pk$ and a secret key $sk$. The encryption function $E_{pk}$ uses the public key $pk$ for its operation $E_{pk} : R \times P \to C$. In this case, $P$ is the plaintext space, $C$ is the ciphertext space and $R$ is the randomness space (all parameterized by $\ell$). At the same time, the decryption function $D_{sk} : C \to P$ uses the secret key $sk$ so that for any plaintext $m \in P$, if $E_{pk}(r,p) = c$, then $D_{sk}(c) = p$ for any $r \in R$. Homomorphic encryption adds to the above the following requirements: there exist binary operations $+$, $\oplus$, $\odot$ defined over the spaces $P$, $R$, $C$ so that $\langle P, +\rangle$, $\langle R, \oplus\rangle$ are the groups written additively and $\langle C, \odot\rangle$ – multiplicatively. We say that an encryption scheme is Homomorphic if for all $r_1, r_2 \in R$ and all $x_1, x_2 \in P$ it holds that

$$E_{pk}(r_1, x_1) \odot E_{pk}(r_2, x_2) = E_{pk}(r_1 \oplus r_2, x_1 + x_2)$$

Informally, this means that if we want to "add" plaintexts that are encrypted, we may "multiply" their corresponding ciphertexts. As a result, we can "add" any two plaintexts (by multiplying corresponding ciphertexts); also we can multiply an encrypted plaintext by an integer constant, by raising its corresponding ciphertext to the power that is equal to the integer constant — which is essentially multiplying a ciphertext by itself a number of times; note that this can be done efficiently by using standard repeated squaring.

**ElGamal Homomorphic Encryption.** We will employ a standard variant of ElGamal encryption [12]. This variant of ElGamal has been employed numerous times in the past (e.g., in the context of e-voting [8]). This public-key encryption scheme is a triple $\langle K, E, D\rangle$ defined as follows:

- Key-generation $K$. Given a security parameter $\ell$, the probabilistic algorithm $K(1^\ell)$ outputs a public-key $pk := \langle p, g, h, f\rangle$ and the corresponding secret-key $x$ so that the following are satisfied: (i) $p$ is a $\ell$-bit prime number so that $(p-1)/2 = q$ is also a prime number. (ii) $g$ is an element of order $q$ in $\mathbf{Z}_p^*$. (iii) $h, f \in \langle g\rangle$ are randomly selected. (iv) $x = \log_g h$.
- Encryption $E$. Given public-key $pk = \langle p, g, h, f\rangle$ and a plaintext $m \in \mathbf{Z}_q$, $E$ samples $r \leftarrow_R \mathbf{Z}_q$ and returns $\langle g^r, h^r f^m\rangle$.
- Decryption $D$. Given secret-key $x$ and a ciphertext $\langle G, H\rangle$ the decryption algorithm returns $G^{-x}H(\bmod p)$. Note that this will only return $f^m$, never-

theless this would be sufficient for our setting as, given a ciphertext $\langle g^r, h^r f^m \rangle$ we will only be interested in testing whether $m \overset{?}{=} 0$ (something that can easily be done by testing $G^{-x}H \overset{?}{\equiv}_p 1$).

Observe that the above encryption scheme is homomorphic: indeed, the randomness space $R$, the plaintext space $P$ and the ciphertext space $C$ satisfy the following: (i) $R = P = \mathbf{Z}_q$ and $(R, \oplus)$, $(P, +)$ are additive groups by setting the operations $\oplus, +$ to be addition modulo $q$. (ii) $C \subseteq \mathbf{Z}_p^* \times \mathbf{Z}_p^*$ and it holds that $(C, \odot)$ is a multiplicative group when $\odot$ is defined as pointwise multiplication modulo $p$. (iii) it holds that for any $r_1, r_2 \in R$, $x_1, x_2$, and $pk = \langle p, g, h, f \rangle$,

$$E_{pk}(r_1, x_1) \odot E_{pk}(r_2, x_2) = \langle g^{r_1}, h^{r_1} f^{x_1} \rangle \odot \langle g^{r_2}, h^{r_2} f^{x_2} \rangle = \langle g^{r_1+r_2}, h^{r_1+r_2} f^{x_1+x_2} \rangle$$

**Interactive Protocols.** A two-party interactive protocol $\mathcal{P}$ is specified by a pair of probabilistic Interactive Turing machines $\langle \mathcal{A}, \mathcal{B} \rangle$. Each TM has input tape, private work tapes, output tape, as well as both have access to a communication tape; one of them is designated to make the first move. An execution of a protocol $\mathcal{P}$ is denoted by $\mathsf{exec}\langle \mathcal{A}(a), \mathcal{B}(b) \rangle$ where $a$ is the private input for $\mathcal{A}$, $b$ is the private input for $\mathcal{B}$. We will denote as $\mathsf{out}_{\mathcal{A}}\langle \mathcal{A}(a), \mathcal{B}(b) \rangle$ and $\mathsf{out}_{\mathcal{B}}\langle \mathcal{A}(a), \mathcal{B}(b) \rangle$ the private outputs of the two ITM's. We write $\mathsf{out}\langle \mathcal{A}(a), \mathcal{B}(b) \rangle$ for the concatenation of the outputs of the two parties.

Here we will consider protocols where only player $\mathcal{A}$ has output. We say that a protocol $\mathcal{P}$ computes a certain functionality $f$ if it holds that for all $a, b$ $\mathsf{out}_{\mathcal{A}}\langle \mathcal{A}(a), \mathcal{B}(b) \rangle = f(a, b)$ (note that $\mathsf{out}_{\mathcal{B}}\langle \mathcal{A}(a), \mathcal{B}(b) \rangle$ is not relevant in this case and it may be designated to just $\top$, a dummy "accept" symbol).

For a given protocol $\mathcal{P} = \langle \mathcal{A}, \mathcal{B} \rangle$ we define as $\mathsf{view}_{\mathcal{A}}\langle \mathcal{A}(a), \mathcal{B}(b) \rangle$ the random variable tuple $\langle a, \rho, m_1, \ldots, m_k \rangle$ where $\rho$ is the internal coin tosses of $\mathcal{A}$ and $m_1, \ldots, m_k$ are the messages received from $\mathcal{B}$. In a similar fashion we define $\mathsf{view}_{\mathsf{B}}\langle \mathcal{A}(a), \mathcal{B}(b) \rangle$.

**Definition 1.** *A protocol $\mathcal{P}$ computing a functionality $f$ is said to be* private w.r.t. semi-honest behavior *if the following hold true for all $a, b$: there exists a simulator $\mathcal{S}$ (respectively $\mathcal{S}'$) so that $\mathcal{S}(a, f(a, b))$ (respectively $\mathcal{S}'(b)$) is computationally indistinguishable from* $\mathsf{view}_{\mathsf{A}}\langle \mathcal{A}(a), \mathcal{B}(b) \rangle\rangle$*. (respectively* $\mathsf{view}_{\mathsf{B}}\langle \mathcal{A}(a), \mathcal{B}(b) \rangle$*).*

Privacy w.r.t. malicious behavior is a bit more complicated as in this case we cannot assume that either party follows the protocol $\mathcal{P}$ as it is specified (i.e., either party may abort or transmit elements that do not follow the specifications).

The way that security is dealt in this case is by comparing the player's views with respect to an "ideal" protocol implementation. In particular, an ideal two-party protocol for the functionality $f$ operates with the help of a trusted-third party $\mathcal{T}$ as follows: player $\mathcal{A}$ transmits to $\mathcal{T}$ the value $a$; player $\mathcal{B}$ transmits to $\mathcal{T}$ the value $b$. $\mathcal{T}$ computes the value $f(a, b)$ and transmits it to player $\mathcal{A}$ while it sends the value $\top$ to player $\mathcal{B}$. If either player fails to transmit its input to $\mathcal{T}$ then $\mathcal{T}$ returns $\perp$ to both parties. Normally $\mathcal{A}, \mathcal{B}$ output whatever output is given by $\mathcal{T}$ in their private output tape.

Privacy w.r.t. malicious behavior then is defined (informally) in the following fashion: for any implementation of player $\mathcal{B}$ denoted by $\mathcal{B}^*$ in the real world there exists a $\mathcal{B}^*_{ideal}$ that operates in the ideal world so that the random variables $\mathsf{out}\langle\mathcal{A}(a),\mathcal{B}^*(b)\rangle$ and $\mathsf{out}\langle\mathcal{A}_{ideal}(a),\mathcal{B}^*_{ideal}(b)\rangle$, are computationally indistinguishable. Likewise, for any implementation of player $\mathcal{A}$ denoted by $\mathcal{A}^*$ in the real world there exists a $\mathcal{A}^*_{ideal}$ that operates in the ideal world so that the random variables $\mathsf{out}\langle\mathcal{A}^*(a),\mathcal{B}(b)\rangle$ and $\mathsf{out}\langle\mathcal{A}^*_{ideal}(a),\mathcal{B}_{ideal}(b)\rangle$, are computationally indistinguishable. A formal definition of the above will be given in the full version — it is omitted here due to lack of space; we refer to [18] for more details w.r.t. secure two-party function evaluation.

**Universally Composable Commitments.** A commitment is a scheme that allows to a party called the committer holding a value $x$ to submit a value $C(x)$ to a party called the receiver so that the two following properties are satisfied (i) hiding: the value $C(x)$ does not reveal any information about $x$, (ii) binding: the committer can "open" $C(x)$ to reveal that it is a commitment to $x$ in a unique way (this is called the decommitment phase). Universally-Composable (UC) commitments, introduced by Canetti and Fischlin [3] is a very useful tool for proving security in the malicious setting for secure function evaluation protocols. Informally, a UC-commitment simulates an ideal commitment scheme where the committer submits $x$ to a trusted third party $\mathcal{T}$ in the commitment phase, and in the decommitment phase, $\mathcal{T}$ simply transfers $x$ to the receiver. More efficient UC-commitments were presented by Damgard and Nielsen [10].

Verifiable encryption of discrete-logarithms was suggested by Camenisch and Shoup in [2] and is a useful tool for constructing UC-commitments. In the construction suggested in [2] (using the common reference string model, cf. [9]) a UC-commitment scheme can be constructed as a pair $\langle\psi, C\rangle$ where $C = \gamma_1^x\gamma_2^r$ and $\psi$ is a verifiable encryption of the pair $x, r$. In a real execution the relative discrete-logarithms of $\gamma_1, \gamma_2$ and the secret-key of the verifiable encryption are unknown; this allows one to prove that the commitment scheme is computationally binding and hiding. In the simulation, on the other hand, the simulator controls the common reference string so that the secret-key of the verifiable encryption as well as the relative discrete-logarithm of $\gamma_1$ base $\gamma_2$ are known; this allows the simulator to extract the committed value as well as equivocate the committed value (open the commitment in an arbitrary way).

**Zero-knowledge Proofs of Language Membership.** Proofs of language membership were introduced in [16]. A proof of knowledge of language membership is a protocol between parties, the prover and the verifier, that allows a prover to show knowledge of a witness $w$ so that $(w, x) \in R$ holds, where $R$ is a polynomial-time relation and $x$ is a publicly known value; the existence of such witness suggests that $x$ belongs to the NP-language $L$ defined as $x \in L \leftrightarrow \exists w : (w, x) \in R$. Such a protocol is called a proof of knowledge if it satisfies the property that the verifier cannot extract any knowledge about $w$ except for the fact that such $w$ exists and it is known to the prover (this requires the existence of a knowledge extractor for the protocol, see [1]).

In our context, we will consider *efficient* and *non-interactive* zero-knowledge proofs of language membership that deal with languages of committed and encrypted values. Regarding non-interactiveness, we will resort to the Fiat-Shamir heuristics [14] that employ a random oracle for providing the challenge for the prover. Security will be argued in the random oracle model. Regarding efficiency, we stress that we will not resort to generic zero-knowledge arguments for NP-languages. Instead, we will rely on three-move zero-knowledge proofs of knowledge, cf. [7], that are specifically designed for the encryption and commitment schemes at hand. Below we describe the proofs that we will employ:

*Proofs of knowledge of a discrete-log representation.* $\mathsf{PK}(x_1, x_2 : y = g_1^{x_1} g_2^{x_2})$. This is a standard protocol used extensively in various constructions; it originates from [4].

*Proof of knowledge of Equality of Discrete-logarithms.* $\mathsf{PK}(x_1, x_2 : y_1 = g_1^{x_1} \wedge y_2 = g_2^{x_2})$, also a standard protocol used extensively; it originates from [5].
*Proof of knowledge of a Product.* $\mathsf{PK}(x_1, x_2 : y = g^{x_1 \cdot x_2})$; this is slightly more tricky than the above, as it requires at least one separate discrete-log based commitment to one of the two values; see e.g., [6].

We will also consider OR/AND compositions of the above protocols [11, 7].

## 3 Private Intersection Predicate Evaluation

In this section we give a formal definition of Private Intersection Predicate Evaluation (PIPE) protocols and we present three protocol constructions for different settings and objectives that allow the computation of the intersection predicate of the two datasets possessed by the two players.

**Definition 2.** *A two-party Private Intersection Predicate Evaluation (*PIPE*) protocol is a pair $\langle \mathcal{A}, \mathcal{B} \rangle$ of ITM's that have the following properties:*

- *(Correctness) For any $S_A, S_B \subseteq \{1, \ldots, N\}$ , let $f(S_A, S_B) \in \{0, 1\}$ so that $f(S_A, S_B) = 1$ if and only if $S_A \cap S_B \neq \emptyset$. $\mathcal{A}, \mathcal{B}$ is a correct* PIPE *protocol if it is a two-party protocol that computes the functionality $f$.*
- *(Security) It will be argued in the semi-honest and malicious behavior setting according to the definitions of section 2.*

*The time and communication complexity of* PIPE *protocols will be measured over the parameters $N, N_A = \#S_A, N_B = \#S_B$ (note that $S_A = a_1, .., b_{N_A}$ and $S_B = b_1, .., b_{N_B}$) as well as a security parameter $\ell$; we will also measure the number of rounds that a protocol requires (a round = a full interaction between A and B).*

Note that we will somewhat relax the definition of security above for protocols PIPE #2 and PIPE #3 in order to achieve sublinear communication in the size of the universe. In particular we will allow to the ideal functionality to release upper bounds on the list sizes in addition to the single bit output (in fact , without loss of generality, we will assume that the ideal functionality releases the sizes of the datasets).

### 3.1 Two-Party PIPE Protocol #1

Consider two players $A$ and $B$ that have sets of values $S_A = \{a_1, \ldots, a_{N_A}\}$ and $S_B = \{b_1, \ldots, b_{N_B}\}$ where $S_A, S_B \subseteq [N]$ and $[N] = \{1, \ldots, N\}$. The two datasets are stored by the players in the form of their characteristic bitstring of length $N$: $S_X$ is represented by a bitstring $Bit_X = \langle bit_1^X, \ldots, bit_N^X \rangle$ so that $bit_j^X = 1$ iff $j \in S_X$, where $X \in \{A, B\}$.

**Step 1.** Player $A$ executes the key generation algorithm for a public-key encryption $\langle K, E, D \rangle$ to obtain $pk, sk$. After this, player $A$ sends to the player $B$ $pk$ and the encryption of $Bit_A$, as follows: $\langle c_1, c_2, \ldots, c_N \rangle$, where $c_j = E_{pk}(bit_j^A)$.

**Step 2.** Upon receiving $\langle c_1, \ldots, c_N \rangle$ and $pk$ from player $A$, player $B$ computes a ciphertext $c$ as follows: Let $random \neq 0$ be some random number drawn uniformly from $R$ (the randomness space of the encryption), then:

$$c = (c_1^{bit_1^B} \odot \cdots \odot c_N^{bit_N^B})^{random} \odot E_{pk}(0)$$

Note that $E_{pk}(0)$ is used to refresh the randomness of the ciphertext $c$. The above expression is equivalent to :

$$c = E_{pk}((bit_1^A \cdot bit_1^B + \cdots + bit_N^A \cdot bit_N^B) \times random + 0)$$

Observe that if $bit_j^A = 1$ and $bit_j^B = 1$, then $bit_j^A \cdot bit_j^B$ will produce 1. In all other cases, the result of multiplication will be 0. It follows that we will obtain $1's$ in all cases of $j \in S_A \bigcap S_B$. The sum of all multiplications together can result in 0 only if $S_A \bigcap S_B = \emptyset$. The sum is multiplied by some random number so that it becomes impossible to determine how many elements belong to the intersection. Player $B$ sends back the value of $c$ to the player $A$ .

**Step 3.** Player $A$, using his secret-key, tests whether the decryption of $c$ is equal to 0 or not; if the decryption is 0, $A$ concludes that $S_A \bigcap S_B = \emptyset$; otherwise, if the decryption is non-zero, player $A$ concludes that there must be at least one element in the joint intersection.

**Theorem 1.** *The above protocol is a correct PIPE protocol that is secure in the semi-honest model under the* CPA *security of* $\langle K, E, D \rangle$*.*

Regarding efficiency, if $\ell$ is the security parameter, and the key-generation, encryption and decryption require $k(\ell), e(\ell), d(\ell)$ time respectively, it holds that (i) the total communication complexity is $2(N+1)\ell$ bits (ii) the time-complexity of player $A$ is $k(\ell) + Ne(\ell) + d(\ell)$, and (iii) the time-complexity of player $B$ is $N\ell + N_B\ell^2 + \ell^3 + e(\ell)$.

### 3.2 Two-Party PIPE Protocol # 2

In the protocol of this section, we will employ a type of a public-key encryption scheme that we call (two-player) superposed encryption. In this kind of encryption scheme, there are two parties each with a pair of public and secret-keys defined over the same public-parameters (e.g., the same prime modulus).

The functionality of the encryption scheme extends regular public-key encryption in the following way: given a plaintext $m'$ and a ciphertext $c$ that encrypts a plaintext $m$ under one player's public-key, one can superpose $c$ with $m'$ to obtain a "superposed ciphertext" $c'$ that can be decrypted by either player to a random ciphertext that hides the plaintext $m \cdot m'$ and can be decrypted by the other player. Formally, superposed encryption is a sequence of procedures $\langle K, K', E, E^{\mathsf{ext}}, D, D^{\mathsf{sup}}\rangle$ defined as follows:

- The key generation algorithm $K$ is comprised by an initial key generation step that produces the public parameter *param*, as well as $K'$ that produces the public-key and secret-key for each user (given the parameter *param*).

  Now given $param \leftarrow K(\ell)$ and $(pk_A, sk_A), (pk_B, sk_B) \leftarrow K'(param)$:
- The two encryption functions are specified as follows: $E_{pk_X} : P \rightarrow C$ and $E^{\mathsf{sup},X}_{pk_A,pk_B} : P \times C \rightarrow C^{\mathsf{sup}}$ for each player $X \in \{A, B\}$.
- The encryption function $E_{pk_X}$ is homomorphic for the plaintext $(P, +)$, randomness $(R, \oplus)$ and ciphertext group $(C, \odot)$. Moreover, $(P, +, \cdot)$ is a ring.
- The superposed encryptions $E^{\mathsf{sup},X}_{pk_A,pk_B}(m, E_{pk_{\overline{X}}}(m'))$ and $E^{\mathsf{sup},X}_{pk_A,pk_B}$ $(m', E_{pk_{\overline{X}}}(m))$ are indistinguishable for any fixed $m, m'$, where $X$ is a player, $X \in \{A, B\}$, and $\overline{X}$ is the other player, $\overline{X} \in \{A, B\} - \{X\}$.
- The decryption functions satisfy the following conditions:
  - $D_{sk_X}(E_{pk_X}(m)) = m$ if $X \in \{A, B\}$, for all $m \in P$.
  - For any fixed $c \in E_{pk_X}(m')$, it holds that if $c'$ is distributed according to $D^{\mathsf{sup}}_{sk_X}(E^{\mathsf{sup},\overline{X}}_{pk_A,pk_B}(m, c))$, then $c'$ is uniformly distributed over $E_{pk_{\overline{X}}}(m \cdot m')$. where $X \in \{A, B\}$ and $\overline{X}$ is the single element of $\{A, B\} - \{X\}$.

Next we define the appropriate notion of security for superposed encryption. Observe the differences from regular semantic security of public-key encryption: the adversary is allowed to *select a ciphertext and a public-key* over which the challenge plaintext will be superposed.

The superposed encryption CPA Game $G^{\mathcal{B}}_{\mathsf{cpa}}$ (denoted by $G^{\mathcal{B}}_{\mathsf{cpa}}(1^\ell)$):

```
1. $param \leftarrow K(1^\ell)$;
2. $(pk_A, sk_A) \leftarrow K'(param)$;
3. $\langle aux, pk_B, c, m_0, m_1\rangle \leftarrow \mathcal{B}(\mathsf{choose}, 1^\ell, param, pk_A)$
4. Choose $b \leftarrow_R \{0, 1\}$;
5. Set $c^* \leftarrow E^{\mathsf{sup},A}_{pk_A,pk_B}(m_b, c)$;
6. $b^* \leftarrow \mathcal{B}(\mathsf{guess}, aux, c^*)$;
7. if $b = b^*$ return $\top$ else return $\bot$;
```

Note that the above game assumes that player $B$ is the "attacker." The identical game where player $A$ is the attacker, will be denoted by $G^{\mathcal{A}}_{\mathsf{cpa}}(1^\ell)$.

**Definition 3.** *A superposed encryption scheme satisfies* CPA *security provided that for any* PPT *attackers* $\mathcal{A}, \mathcal{B}$ *it holds that* $2\mathbf{Prob}[G^{\mathcal{A}}_{\mathsf{cpa}}(1^\ell) = \top] - 1$ *and* $2\mathbf{Prob}[G^{\mathcal{B}}_{\mathsf{cpa}}(1^\ell) = \top] - 1$ *are negligible functions in the security parameter* $\ell$.

Below we show that superposed encryption CPA security implies regular CPA security of the underlying public-key encryption scheme.

**Theorem 2.** *Let $\langle K, K', E, E^{\mathsf{sup}}, D, D^{\mathsf{sup}}\rangle$ be a superposed encryption scheme that satisfies* CPA *security. Then, the underlying public-key encryption $\langle K'', E, D\rangle$ (where $K''$ is the composition of $K$ and $K'$) is a* CPA *pk encryption scheme.*

As a side note, the opposite does not hold necessarily and CPA security of the underlying public-key encryption does not appear to imply CPA security for the superposed scheme.

**Construction.** The scheme that we will use is based on ElGamal encryption and it is as follows:

- $K$, given $1^\ell$, samples a prime number $p$ such that $p = 2q+1$ with $q$ also prime. Then it selects a $g' \in \mathbf{Z}_p^*$ and sets $g \leftarrow (g')^2 (\text{mod} p)$ and selects $h \leftarrow_R \langle g\rangle$. The public parameter is $param := \langle p, q, g, h\rangle$. The user key generation operates as follows: given $\langle p, q, g\rangle$ it samples $x \leftarrow_R \mathbf{Z}_q$ and sets $y_X := g^x (\bmod p)$, with $pk_X = y_X$ and $sk_X = x$.
- The encryption function $E_{pk_X}$ selects $r \leftarrow_R \mathbf{Z}_q$ and returns $\langle g^r, y_X^r h^m\rangle$ (for $X \in \{A, B\}$). The ciphertext encryption function $E^{\mathsf{sup},A}_{pk_A,pk_B}$ takes a ciphertext pair $\langle G, H\rangle$ and the plaintext $m$, it samples $r, r' \leftarrow_R \mathbf{Z}_q$ and returns the triple $\langle g^r, G^m g^{r'}, y_X^r y_{\overline{X}}^{r'} H^m\rangle$. Likewise the ciphertext encryption $E^{\mathsf{sup},B}_{pk_A,pk_B}$ takes a ciphertext pair $\langle G, H\rangle$ and the plaintext $m$, it samples $r, r' \leftarrow_R \mathbf{Z}_q$ and returns the triple $\langle G^m g^r, g^{r'}, y_X^r y_{\overline{X}}^{r'} H^m\rangle$.

  The decryption function $D^{\mathsf{sup}}_{\mathsf{sk_A}}(c)$ (respectively $D^{\mathsf{sup}}_{\mathsf{sk_B}}(c)$) for $c = \langle G_A, G_B, Y\rangle$ it returns the ciphertext $\langle G_B, Y G_A^{-sk_A}\rangle$ (respectively $\langle G_A, Y G_B^{-sk_B}\rangle$).

To see that the above scheme is a superposed encryption observe:

$$D^{\mathsf{sup}}_{sk_A}(E^{\mathsf{sup},B}_{pk_A,pk_B}(m, E_{\mathsf{pk_A}}(m'))) = D^{\mathsf{sup}}_{sk_A}(\langle g^r, g^{r'}, y_A^r y_B^{r'} h^{m\cdot m'}\rangle) = \langle g^{r'}, y_B^{r'} h^{m\cdot m'}\rangle$$

**Theorem 3.** *The superposed encryption scheme presented above satisfies* CPA *security under the DDH assumption.*

**The PIPE protocol.** Suppose that $\langle K, E, E^{\mathsf{sup}}, D, D^{\mathsf{sup}}\rangle$ is a superposed encryption, and the two players possess the lists $S_A, S_B$ respectively that are subsets of $[N]$ with $N_A = \#S_A$ and $N_B = \#S_B$.

**Step 0.** The two players $A, B$ receive as public joint input $param \leftarrow K(1^\ell)$ and each one executes separately the key-generation algorithm to obtain $(pk_A, sk_A) \leftarrow K_A(param)$, $(pk_B, sk_B) \leftarrow K_B(param)$.

Player $A$ selects $\alpha_0, \ldots, \alpha_{N_A} \in \mathbf{Z}_q$ such that the polynomial $f(x) := \alpha_0 + \alpha_1 x + \ldots \alpha_{N_A} x^{N_A}$ has the property that $f(a) = 0$ if and only if $a \in S_A$.

Also player $A$ computes $c^* = E^{\mathsf{sup},B}_{pk_A,pk_B}(1, E_{pk_A}(1))$.

The following steps are repeated for $j = 1, \ldots, N_B$:

**Step $j$.1.** Player $A$ computes $c = D^{\mathsf{sup}}_{sk_A}(c^*)$. Player $A$ transmits to player $B$ the sequence of superposed ciphertexts,

$$\langle c^*_0, \ldots, c^*_{N_A} \rangle = \langle E^{\mathsf{sup},A}_{pk_A,pk_B}(\alpha_0, c), \ldots, E^{\mathsf{sup},A}_{pk_A,pk_B}(\alpha_{N_A}, c) \rangle$$

**Step $j$.2.** Player $B$ decrypts the superposed ciphertexts as follows:

$$\langle c_0, \ldots, c_{N_A} \rangle = \langle D^{\mathsf{sup}}_{sk_B}(c^*_0), \ldots, D^{\mathsf{sup}}_{sk_B}(c^*_{N_A}) \rangle$$

Observe that, if $j = 1$,

$$\langle c_0, \ldots, c_{N_A} \rangle \in \langle E_{pk_A}(\alpha_0), \ldots, E_{pk_A}(\alpha_{N_A}) \rangle$$

or for $j > 1$,

$$\langle c_0, \ldots, c_{N_A} \rangle \in \langle E_{pk_A}(f(b_1) \ldots f(b_{j-1}) \cdot \alpha_0), \ldots, E_{pk_A}(f(b_1) \ldots f(b_{j-1}) \cdot \alpha_{N_A}) \rangle$$

Following this, player $B$ computes

$$c' = E_{pk_A}(0) \cdot c_0 \cdot (c_1)^{b_j} \ldots (c_{N_A})^{b_j^{N_A}} \in E_{pk_A}(f(b_1) \ldots f(b_j))$$

Observe that, $c'$ is uniformly distributed over $E_{pk_A}(f(b_1) \ldots f(b_j))$. Then player $B$ computes an encryption of the ciphertext $c'$ using the superposed encryption function

$$c^* = E^{\mathsf{sup},B}_{pk_A,pk_B}(1, c')$$

and transmits $c^*$ to player $A$; the step $j + 1$ is executed now.

In the final round when $j = N_B$ the following modifications are made:

**Final Round.** Player $B$ in step $\mathbf{N_B}$**.2** does not compute $c^*$ using the superposed encryption $E^{\mathsf{sup},B}_{pk_A,pk_B}$; instead, he computes a regular ciphertext $c^{\mathsf{Fin}}$ as follows:

$$c^{\mathsf{Fin}} = (c')^{random} \cdot E_{pk_A}(0)$$

where $random \leftarrow_R \mathbf{Z}_q - \{0\}$. Observe that $c^{\mathsf{Fin}} \in E_{pk_A}(0)$ if $f(b_1) \ldots f(b_{N_B}) = 0$; otherwise, $c^{\mathsf{Fin}} \in E_{pk_A}(s)$ where $s$ is a random non-zero element of $\mathbf{Z}_q$.

When player $A$ receives $c^{\mathsf{Fin}}$, he computes $s = D_{sk_A}(c^{\mathsf{Fin}})$ and concludes that the intersection is empty if $s \neq 0$, or that there exists an intersection in case of $s = 0$.

**Theorem 4.** *The protocol described above is a correct* PIPE *protocol that is secure in the semi-honest model under the* CPA *security of the superposed pk-encryption.*

Regarding efficiency, we need $N_B$-rounds and for security parameter $\ell$ and if the key-generation, encryption and decryption require $k(\ell), e(\ell), d(\ell)$ time respectively, it holds that the total communication is $3N_B(N_A+2)\ell$ bits; the time-complexity of player $A$ is $\Theta(k(\ell) + N_B N_A e(\ell) + N_B d(\ell))$; the time-complexity of player $B$ is $\Theta(k(\ell) + N_B e(\ell) + N_B N_A d(\ell) + N_A N_B \ell^2 + N_A N_B \ell^2 \log N + \ell^3)$.

### 3.3 Two-Party PIPE Protocol # 3

Checking whether intersection exists for players that possess relatively small lists of essentially constant size (compared to the universe $[N]$) can be phrased in the context of multivariate polynomial evaluation to allow a protocol with optimal round complexity. Suppose now that players $A$ and $B$ have small lists of values $S_A = \{a_1, \ldots, a_{N_A}\}$ and $S_B = \{b_1, \ldots, b_{N_B}\}$ where $N_A \ll N$ and $N_B \ll N$. First, player $A$ selects $\langle \alpha_0, \alpha_1, \ldots, \alpha_{N_A} \rangle$ for the polynomial

$$Pol(z) = \sum_{u=0}^{N_A} \alpha_u z^u = \alpha_0 + \alpha_1 z + \alpha_2 z^2 + \cdots + \alpha_{N_A} z^{N_A}$$

so that $Pol(a) = 0$ iff $a \in S_A$.

As in the case of PIPE protocol #2, player $A$ wants to evaluate $F(b_1, b_2, \ldots, b_{N_B}) = \prod_{j=1}^{N_B} Pol(b_j)$ in order to find if $S_A \bigcap S_B$ is non-empty. If at least one $Pol(b_y)$ is equal to 0 (i.e., $b_y$ is in the $S_A \bigcap S_B$), then obviously $F(b_1, b_2, \ldots, b_t)$ will evaluate in 0.

A direct application of the techniques of [15] in this setting will result in a protocol of communication complexity $\Theta((N_A + 1)^{N_B})$, as the total number of coefficients of $F(b_1, b_2, \ldots, b_{N_B})$ is $(N_A + 1)^{N_B}$. In the remaining of the section, using the observation that many of coefficients are repeated, we will reduce the communication complexity to $\Theta(\binom{N_A+N_B}{N_B})$. Observe that the number of distinct coefficients in the computation of $F$ equals the number of possible multisets of size $N_B$ from the alphabet of size $N_A + 1$. Therefore, the number of distinct coefficients is $\binom{N_A+1+N_B-1}{N_B} = \binom{N_A+N_B}{N_B}$.

Let us denote the array of such coefficients as $Cf = \langle cf[1], \ldots, cf[\binom{N_A+N_B}{N_B}] \rangle$.

Let $\mathcal{I}$ contain all monotonically decreasing tuples $\langle i_1, \ldots, i_{N_B} \rangle$ of $\{0, \ldots, N_A\}^{N_B}$ so that $i_\ell \geq i_{\ell'}$ for $\ell > \ell'$. For $j = 1, \ldots, \binom{N_A+N_B}{N_B}$ we define $cf[j]$ to be $cf[j] = \alpha_{i_1} \alpha_{i_2} \ldots \alpha_{i_{N_B}}$ where $\langle i_1, \ldots, i_{N_B} \rangle$ is the $j$-th tuple of $\mathcal{I}$.

To describe the protocol, we need to specify an order in which the tuples $\langle i_1, \ldots, i_{N_B} \rangle$ are generated: we will use an inverse lexicographic order in $\mathcal{I}$, i.e., $\langle N_A, \ldots, N_A \rangle$ is the first element and the subsequent elements are in the order are defined by the function $next[\langle i_1, ..., i_{N_B} \rangle] = \langle i_1, ...., i_{t-1}, i_t - 1, i_t - 1, \ldots, i_t - 1 \rangle$ where $t = min\{1, ..., N_B\}$ with the property $i_{t+1} = \cdots = i_{N_B} = 0$. Note, that if $i_{N_B} \neq 0$ then $t = N_B$ (the last element).

The protocol description is as follows:

**Step 1.** Player $A$ executes the key generation algorithm for a public-key encryption $\langle K, E, D \rangle$ to obtain $pk, sk$.

Then, player $A$ sends to the player $B$ the encryption of $Cf$, as follows: $\langle \xi[1], \xi[2], \ldots, \xi[\binom{N_A+N_B}{N_B}] \rangle$, where $\xi[j] = E_{pk}(cf[j])$:

$$\boldsymbol{c} = \langle E_{pk}(cf[1]), E_{pk}(cf[2]), \ldots, E_{pk}(cf[\binom{N_A + N_B}{N_B}]) \rangle$$

**Step 2.** Let $oc_{\boldsymbol{i}}[j]$ equal the number of times the element $j$ occurs in the monotonically decreasing tuple $\boldsymbol{i} = \langle i_1, \ldots, i_{N_B} \rangle$. Observe that for such a tuple, the value $T_{\boldsymbol{i}} = \frac{N_B!}{oc_{\boldsymbol{i}}[0]!oc_{\boldsymbol{i}}[1]!\ldots oc_{\boldsymbol{i}}[N_A]!}$ corresponds to the number of times the coefficient $cf[\boldsymbol{i}]$ is repeated in $F$. Note, $oc_{\boldsymbol{i}}[0] + \cdots + oc_{\boldsymbol{i}}[N_A] = N_B$.

Note that a tuple $\boldsymbol{i} = \langle i_1, i_2, \ldots, i_{N_B}\rangle$ can be permuted $T_{\boldsymbol{i}}$ times in total; let $\langle i_1^{(j)}, i_2^{(j)}, \ldots, i_{N_B}^{(j)}\rangle$ denote the $j$-th permutation of this tuple. If $v = 1, \ldots, \binom{N_A+N_B}{N_B}$, let $\boldsymbol{i}[v]$ denote the $v$-th monotonically decreasing tuple of $\{0, \ldots, N_A\}^{N_B}$ that will be also denoted by $\langle i_1[v], \ldots i_{N_B}[v]\rangle$. Player $B$ upon receiving $\boldsymbol{c} = \langle \xi[1], \ldots, \xi[\binom{N_A+N_B}{N_B}]\rangle$, will perform the following: let $random \neq 0$ be some random number drawn uniformly from $P$ (the plaintext space); then, player $B$ computes:

$$F^{en} = ( \bigodot_{v=1}^{\binom{N_A+N_B}{N_B}} \xi[v]^{\sum_{j=1}^{T_{\boldsymbol{i}[v]}} \prod_{\ell=1}^{N_B} b_\ell^{i_\ell^{(j)}[v]}} )^{random} \odot E_{pk}(0)$$

The above expression is equivalent to :

$$F^{en} = E_{pk}(( \sum_{v=1}^{\binom{N_A+N_B}{N_B}} cf[v] \sum_{j=1}^{T_{\boldsymbol{i}[v]}} \prod_{\ell=1}^{N_B} b_\ell^{i_\ell^{(j)}[v]})random) = E_{pk}(F(b_1, \ldots, b_{N_B})random)$$

The evaluation of $F^{en}$ will result in 0 only if $S_A \bigcap S_B \neq \emptyset$. Player $B$ sends back the value of $F^{en}$ to the player $A$ .

**Theorem 5.** *The protocol for players with small sets is a correct PIPE protocol that is secure in the semi-honest model under the* CPA *security of $\langle K, E, D\rangle$.*

This protocol's total communication complexity is $\binom{N_A+N_B}{N_B} + 1$ ciphertexts. The time-complexity on the side of player $A$ is $\binom{N_A+N_B}{N_B}$ encryptions and one decryption; the complexity for player $B$ is $\binom{N_A+N_B}{N_B}$ ciphertext multiplications, one encryption, and $\binom{N_A+N_B}{N_B}$ exponentiations; note that the time of computing the exponents $\sum_{j=1}^{T_{\boldsymbol{i}[v]}} \prod_{\ell=1}^{N_B} b_\ell^{i_\ell^{(j)}[v]}$ for $v = 1, \ldots, \binom{N_A+N_B}{N_B}$ is proportional to $(N_A + 1)^{N_B}$ (equal to the total number of terms that should be computed for the multivariate polynomial); nevertheless, the values of these terms can be precomputed by player $B$ since they only involve the variables from the player's $B$ list. Thus, the online complexity will be proportional to $\binom{N_A+N_B}{N_B}$ steps.

## 4 Dealing with Malicious Parties

In this section we outline how the PIPE protocols presented in the previous sections can be modified so that they can be proven secure in the setting where either party is allowed to operate in a malicious way (as opposed to semi-honest). The basic tools that will be necessary for the transformation are universally composable commitments and zero-knowledge proofs of language membership (see section 2). Our general approach for all three protocols will be as follows:

players will provide UC-commitments for their private inputs and will use non-interactive zero-knowledge proofs of language membership to ensure that their computations are consistent with the UC-commitments. Security will be subsequently argued in the random oracle model.

**PIPE Protocol #1 for Malicious Parties.** Let $\mathsf{Com}_{\mathsf{crs}}(r, x)$ be of the form $\langle \gamma_1^r \gamma_2^x (\bmod p), \psi \rangle$ where the first component is a Pedersen-like discrete-log based commitment scheme that is made universable composable using verifiable encryption of [2], i.e., $\psi$ is a verifiable encryption of $r, x$; note that $\gamma_1, \gamma_2$ belong to the common reference string $\mathsf{crs}$. We will write $\mathsf{Com}_{\mathsf{crs}}(x)$ to denote the random variable defined by $\mathsf{Com}_{\mathsf{crs}}(r, x)$ over all possible random coin tosses $r$.

**Step 1.** Player $A$ executes the key generation algorithm for a public-key encryption $\langle K, E, D \rangle$ to obtain $pk, sk$. Player $A$ sends to player $B$ the ciphertexts $\langle c_1, c_2, \ldots, c_N \rangle$, where $c_j := E_{pk}(bit_j^A)$ as well as the commitments $\delta_j := \mathsf{Com}_{\mathsf{crs}}(bit_j^A)$ for $j = 1, \ldots, N$. Player $A$ accompanies each pair $(c_j, \delta_j)$ with a non-interactive proof of language membership $\mathsf{PK}(r, r', x : c_j = E_{pk}(r, x) \wedge \delta_j = \mathsf{Com}_{\mathsf{crs}}(r', x) \wedge x \in \{0, 1\})$.

**Step 2.** Player $B$ computes the ciphertext $c$ as in the semi-honest case, $c = (c_1^{bit_1^B} \odot \cdots \odot c_N^{bit_N^B})^{random} \odot E_{pk}(r, 0)$. where $r \leftarrow_R R$. Player $B$ also computes the commitments $\delta'_j = \mathsf{Com}_{\mathsf{crs}}(bit_j^B)$ for $j = 1, \ldots, N$ and $\delta'_{N+1} = \mathsf{Com}_{\mathsf{crs}}(random)$ as well as $\delta'_{N+2} = \mathsf{Com}_{\mathsf{crs}}(r)$ ; player $B$ transmits to player $A$ the values $c, \delta'_1, \ldots, \delta'_N$ as well as a non-interactive proof of knowledge $\mathsf{PK}(r'_1, \ldots, r'_{N+2}, x_1, \ldots, x_N, r', random : c = (c_1^{x_1} \odot \cdots \odot c_N^{x_N})^{random} \odot E_{pk}(r, 0) \wedge_{j=1}^{N} [\delta_j = \mathsf{Com}_{\mathsf{crs}}(r'_j, x_j) \wedge x_j \in \{0, 1\}] \wedge \delta'_{N+1} = \mathsf{Com}_{\mathsf{crs}}(r'_{N+1}, random) \wedge \delta'_{N+2} = \mathsf{Com}_{\mathsf{crs}}(r'_{N+2}, r))$. Note that such proof of knowledge can be constructed efficiently, cf. section 2.

**Step 3.** This is the same as in the semi-honest case: player $A$ tests whether the decryption of $c$ is equal to 0 or not.

We note that either player aborts the protocol in case some of the non-interactive proofs of knowledge do not verify.

**Theorem 6.** *The above protocol is a correct PIPE protocol that is secure in the malicious setting in the random oracle model.*

**PIPE protocol #2 and #3 for malicious parties.** Due to lack of space we omit from this extended abstract the transformation of PIPE protocols # 2 and # 3 in the malicious adversary setting. We remark that the transformation is based on the same principles as above, i.e., the employment of UC-commitments and the appropriate non-interactive proofs of language membership that show that players' moves are consistent with their commitments.

## References

1. Mihir Bellare and Oded Goldreich, *On Defining Proofs of Knowledge*, CRYPTO 1992: 390-420.
2. Jan Camenisch, Victor Shoup, *Practical Verifiable Encryption and Decryption of Discrete Logarithms*, CRYPTO 2003: 126-144

3. Ran Canetti and Marc Fischlin,*Universally Composable Commitments*, CRYPTO 2001: 19-40
4. David Chaum, Jan-Hendrik Evertse and Jeroen van de Graaf, *An Improved Protocol for Demonstrating Possession of Discrete Logarithms and Some Generalizations*, EUROCRYPT 1987: 127-141
5. D. Chaum and T. Pedersen *Wallet databases with observers*, In Advances in Cryptology – Crypto '92, pages 89-105, 1992.
6. Ronald Cramer and Ivan Damgard *Zero-Knowledge Proofs for Finite Field Arithmetic; or: Can Zero-Knowledge be for Free?*, CRYPTO 1998: 424-441
7. Ronald Cramer, Ivan Damgard, Berry Schoenmakers, *Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols*, CRYPTO 1994: 174-187
8. Ronald Cramer, Rosario Gennaro and Berry Schoenmakers, *A Secure and Optimally Efficient Multi-Authority Election Scheme*, EUROCRYPT 1997, pp. 103-118.
9. Ivan Damgard, *Efficient Concurrent Zero-Knowledge in the Auxiliary String Model* EUROCRYPT 2000: 418-430
10. Ivan Damgard, Jesper Buus Nielsen, *Perfect Hiding and Perfect Binding Universally Composable Commitment Schemes with Constant Expansion Factor*, CRYPTO 2002. pp. 581-596.
11. Alfredo De Santis, Giovanni Di Crescenzo, Giuseppe Persiano and Moti Yung, *On Monotone Formula Closure of SZK*, FOCS 1994: 454-465.
12. T. ElGamal. *A public-key cryptosystem and a signature scheme based on discrete logarithms*, IEEE Transactions on Information Theory, IT-31(4):469–472, July 1985.
13. Ronald Fagin, Moni Naor, and Peter Winkler. Comparing information without leaking it. Communications of the ACM, 39(5):77–85, 1996.
14. Amos Fiat and Adi Shamir *How to Prove Yourself: Practical Solutions to Identification and Signature Problems*, CRYPTO 1986: 186-194.
15. Michael Freedman, Kobbi Nissim and Benny Pinkas, *Efficient private matching and set intersection*, EUROCRYPT 2004.
16. Shafi Goldwasser, Silvio Micali and Charles Rackoff, *The Knowledge Complexity of Interactive Proof Systems* SIAM J. Comput. 18(1): 186-208 (1989)
17. Bart Goethals, Sven Laur, Helger Lipmaa and Taneli Mielikäinen, *On Secure Scalar Product Computation for Privacy-Preserving Data Mining.* , The 7th Annual International Conference in Information Security and Cryptology (ICISC 2004), December 2–3, 2004.
18. Oded Goldreich, *Secure Multi-Party Computation*, unpublished manuscript, 2002. `http://www.wisdom.weizmann.ac.il/ oded/pp.html`.
19. B. Kalyanasundaram and G. Schnitger. *The probabilistic communication complexity of set intersection*, SIAM Journal on Discrete Math, 5(5):545–557, 1992.
20. Helger Lipmaa. Verifiable homomorphic oblivious transfer and private equality test. In Advances in Cryptology, ASIACRYPT 2003, pp. 416–433.
21. D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella. *The Fairplay project*, http://www.cs.huji.ac.il/labs/danss/FairPlay.
22. Moni Naor and Benny Pinkas. Oblivious transfer and polynomial evaluation. In Proc. 31st Annual ACM Symposium on Theory of Computing, pages 245–254, Atlanta, Georgia, May 1999.
23. Alexander A. Razborov, *On the Distributional Complexity of Disjointness*, Theor. Comput. Sci. 106(2): 385-390 (1992)
24. Berry Schoenmakers and Pim Tuyls, *Practical Two-Party Computation Based on the Conditional Gate*, ASIACRYPT 2004, pp. 119-136.
25. A. C. Yao, *How to generate and exchange secrets*, In Proceedings of the 27th IEEE Symposium on Foundations of Computer Science, pages 162-167, 1986.

# RFID Traceability: A Multilayer Problem

Gildas Avoine and Philippe Oechslin

EPFL
Lausanne, Switzerland

**Abstract.** RFID tags have very promising applications in many domains (retail, rental, surveillance, medicine to name a few). Unfortunately the use of these tags can have serious implications on the privacy of people carrying tagged items. Serious opposition from consumers has already thwarted several trials of this technology. The main fears associated with the tags is that they may allow other parties to covertly collect information about people or to trace them wherever they go. As long as these privacy issues remain unresolved, it will be impossible to reap the benefits of these new applications. Current solutions to privacy problems are typically limited to the application layer. RFID system have three layers, application, communication and physical. We demonstrate that privacy issues cannot be solved without looking at each layer separately. We also show that current solutions fail to address the multilayer aspect of privacy and as a result fail to protect it. For each layer we describe the main threats and give tentative solutions.

**Keywords:** RFID, Privacy, Collision Avoidance, Communication Model.

## 1 Introduction

Often presented as a new technological revolution, Radio Frequency Identification (RFID) systems make possible the identification of objects in an environment, with neither physical nor visual contact. They consist of transponders inserted into the objects, of readers which communicate with the transponders using radio frequencies and usually of a database which contains information on the tagged objects. This technology is not fundamentally new. It has existed for several years, for example for ticketing on public transport, on motorway tollgates or ski-lifts and also for animal identification.

The boom which RFID technology is enjoying today rests essentially on the willingness to develop transponders which are cheap and of reduced size. The direct consequence of this step is a reduction in the capacity of the transponders, that is to say their computation, storage and communication capacities. This willingness is due (in part) to the Auto-ID Center[1] whose purpose is to standardise and promote very cheap RFID technology, by reducing the price

[1] The Auto-ID Center split in 2003, giving rise to the EPCGlobal Inc. [4] and the Auto-ID Labs [1].

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 125–140, 2005.
© Springer-Verlag Berlin Heidelberg 2005

of transponders to less than 5 cents. Because of their reduced capacities, these transponders, usually called *tags*, bring their share of problems, in particular with regard to privacy issues, whether it be information leakage or traceability.

Firstly, we will present the existing and potential applications related to RFID and we will give a brief description of the technology. Then we will present in Section 2 the privacy threats in RFID systems and we show that contrary to the three well-known cryptographic concepts, i.e., confidentiality, authenticity, and integrity, traceability cannot be ensured in the application layer only, but it must be ensured in each of the layers of the communication model. We will then analyse the traceability threats in each of the three layers of the radio frequency communication model and we will suggest some ideas in order to thwart them. We will go from the application layer in Section 3 to the physical layer in Section 5. We will finally summarise our analysis in Section 6.

### 1.1 RFID Applications

Advocates of RFID tags call them the super barcodes of the future. Based on a very different technology, identification by radio frequency represents a major innovation in relation to optical identification. In addition to the miniaturisation of the tags which allows them to be implanted within objects, it allows objects to be read en masse, without the need for visual contact. It should also be noted that each tag has a unique identifier: whilst a bar code represents a group of objects, an electronic tag represents a single object.

One area of application for RFID tags is the management of stock and inventories in shops and warehouses. The American mass-marketing giant, Wal-Mart, has recently placed a requirement on its suppliers that they use electronic tags on the palettes and packaging boxes that are delivered to it. This is a progressive policy and, at the beginning, it will only affect suppliers of pharmaceutical products.

The introduction of RFID tags in all articles could also directly benefit the consumer. One obsession of customers is cutting the waiting time at tills, by replacing the shop assistants with an entirely automated device: one would simply pass the contents of the trolley through a reading tunnel. This application will not see the light of day anytime soon, principally for technical reasons, but also for a less frequently thought about reason like fraud. Indeed, the electronic tags can be cloned or rendered ineffective through various processes, which clears the way for malicious activity. Even though barcodes can equally be cloned by a simple photocopy, this type of fraud is thwarted by a human presence when the goods are scanned at the till: in case of doubt, the shop assistant can verify the appropriateness of a product with the description corresponding to the barcode. Some visionaries go even further: the tags could contain information useful in the home, like washing, cooking or storing instructions. Thus maybe the washing machine that asks for confirmation before washing whites with reds or the refrigerator that discovers that a pot of “crème fraîche” stored on its shelves is no longer as fresh as its name suggests may no longer be science fiction?

These domestic applications are still experimental and should not detract from the very real applications which already surround us, e.g., the identification of pets by RFID is already a part of our everyday lives. In the European Union, this practice is already obligatory in some countries and will extend across the whole EU in a few years.

### 1.2 RFID Technology

Very cheap tags, electronic microcircuits equipped with an antenna, have extremely limited computation, storage, and communication capacities, because of the cost and size restrictions imposed by the targeted applications.

They have no microprocessors and are only equipped with a few thousand logic gates at the very most, which makes it a real challenge to integrate encryption or signature algorithms into these devices. This difficulty is reinforced by the fact that the tags are passive, meaning that they do not have their own energy source: they use the power supplied by the magnetic or electric field of the reader. Let us note, however, that promising research is being done at the moment, notably the implementation of AES for RFID tags proposed by Feldhofer, Dominikus and Wolkerstorfe [6].

The storage capacities of RFID tags are also extremely limited. The cheapest devices have only between 64 and 128 bits of ROM memory, which allows the unique identifier of the tag to be stored. Adding EEPROM memory remains an option for more developed applications. Whilst some memory zones can be made remotely inaccessible, the tags are not tamper-resistant, unlike smartcards made for secure applications (credit cards, pay TV, etc.).

The communication distance between tags and readers depends on numerous parameters, in particular the communication frequency. Two principal categories of RFID systems coexist: the systems using the frequency 13.56MHz and the systems using the frequency 860-960MHz, for which the communication distance is greater. In this latter case, the information sent by the reader can in practice be received up to a hundred meters, but the information returned from the tag to the reader reaches a few meters at most. These limits, resulting from standards and regulations, do not mean that the tags cannot be read from a greater distance: non-conforming equipment could exceed these limits, for example by transgressing the laws relating to the maximum authorised power.

## 2 Privacy Threats

Among the security issues related to RFID technology, one can distinguish those which threaten the functionality of the system from those which pose a threat to the privacy of its users, i.e., by divulging information about the user or allowing the user to be traced. The fact that the tags can be invisible, that they can be read remotely, and that they have a long life (considering that they do not have their own energy source), makes these privacy issues worse. Moreover, the ease of recording and automatically dealing with the logs obtained by RFID systems contributes to the desire for protection against the undesirable effects of these systems.

Beyond the usual denial of service attacks, threats to the functionality of RFID systems are linked to the falsification of the tags and their concealment. As discussed in the previous section, a cheap tag cannot benefit from protection mechanisms such as those enjoyed by smartcards made for secure applications. Therefore an adversary can obtain the memory content and create a clone of the tag. This operation is obviously simplified if the tag openly transmits all its data, as is the case in the common applications. Although reducing the reading distance reduces the risks of eavesdropping, it is not a satisfactory solution. High gain antennas and use of non conforming power levels may still make it possible to read a tag from greater distances. The possibility of neutralising the tags also prevents the correct functioning of the system. The totally automated trolley reader discussed in Section 1.1 is particularly vulnerable to this kind of attack, since foil or a simple chips packet can be enough to neutralise the tag by forming a Faraday cage.

Below we will concentrate on threats to the privacy of RFID tag carriers. These threats fall into two categories: information leakage and traceability.

### 2.1 Information Leakage

The disclosure of information arising during the transmission of data by the tag reveals data intrinsic to the object or its environment. For example, tagged pharmaceutical products could reveal data about the health of a person. An employer, insurer or other party could have a particular interest in knowing the state of health of a person that he is close to, and could so obtain sensitive information. The tags are not, however, made to contain or transmit large quantities of data. When a database is present in the system, the tag can send a simple identifier, so that only the people who have access to this database can match the identifier to the corresponding information. This is the principle adopted by systems using barcodes.

### 2.2 Traceability

The problem of traceability is more complex. Even if a tag only transmits an identifier, this information can be used to trace an object in time and space. If a link can be established between a person and the tags he is carrying, the tracing of objects can become the tracing of a person. An attacker may want to trace a given tag, either deterministically or probabilistically, starting from either an active or passive attack.

A simple approach for dealing with the problem of privacy is to prevent the readers from receiving data coming from the tags. Besides the difficulty of putting these techniques into practice, they have the pernicious side-effect that they can also be used by an adversary to harm the system. The first technique arising from the need to ensure privacy is to *kill the tag*. The technique is effective but has the major inconvenience that the tag can no longer be used. A less radical method consists of *preventing the tag from hearing the request* by enclosing the tag in a Faraday cage as we have already mentioned. This solution is only suitable for a few precise applications, e.g., money wallets, but is not for general use: animal identification is an example of an application which could not benefit from this technique. The third technique consists of *preventing the reader from understanding the reply.* The best illustration of this technique is surely the *blocker tag* [14] which aims to prevent a reader from determining which tags are present in its environment. Broadly, the blocker tag relies on the tree walking protocol (see Section 4.2) and simulates the full spectrum of possible identifiers.

Another approach is to design protocols which can identify the tags without compromising the privacy of their carriers. In spite of the huge interest that RFID technology has caused (and the fears of consumers), relatively few people have worked on such protocols. Sarma, Weis and Engels were the first to take a step in this direction [19]. Other protocols were then put forward, in particular by Juels *et al.* [13], Ohkubo *et al.* [16], Henrici and Müller [7], Feldhofer *et al.* [6], Molnar and Wagner [15], etc. Up until now, little work has been done to prove the security or to exhibit weaknesses of the proposed protocols. Only Avoine [2] and Saito *et al.* [18] paved the way by showing weaknesses in some existing schemes.

Unfortunately, we will show in Section 2.3 that even if an identification protocol is proven to ensure the privacy in the classical adversarial models, this does not mean that the protocol truly ensures privacy in practice.

### 2.3 Relationship Between Traceability and Layers

The three main concepts that are considered in cryptography are confidentiality, integrity and authentication. To analyse these concepts, a model of the adversary is defined, that is, the actions that the adversary may carry out on the entities and their communication channels in order to compromise confidentiality, integrity or authentication. This model is usually defined in theoretic notions like tamperproofness of the entities or timeliness of the channels without considering the exact nature of the underlying physical architecture.

The communication channels are usually devised using a layered approach (as in the OSI model [12]). By implementing a corresponding protocol at a given layer, confidentiality, integrity or authentication can be guaranteed independently from the characteristics of the lower layers. With regard to traceability, the problem is very different. Each layer can reveal information which can be used to trace a tag and we have to prove that the system is tracing-resistant at each layer. Thus, a protocol that is safe with regard to traceability in a classic adversary model may not be safe in practice. This is the case for all RFID protocols that have been described in the literature, since lower layers are never taken into consideration. It is thus of paramount importance to investigate traceability issues at each layer of the communication model. Below we refer to the model in Fig. 1 which is compatible with the ISO standard 18000-1 [10]. It is made of three layers, the application, the communication and the physical layer.

- The *application layer* handles the information defined by the user. This could be information about the tagged object (e.g., the title of a book) or more probably an identifier allowing the reader to extract the corresponding information from a database. To protect an identifier, an application protocol may transform the data before it is transmitted or deliver the information only if certain conditions are met.

- The *communication layer* defines the way in which the readers and tags can communicate. Collision avoidance protocols are found in this layer as well as an identifier that makes it possible to single out a specific tag for communication with a reader (this identifier does not have to be the same as the one in the application layer).
- The *physical layer* defines the physical air interface, that is to say, the frequency, modulation of transmission, data encoding, timings and so on.

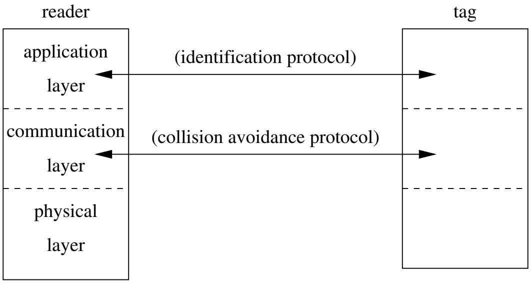


**Fig. 1.** Communication model

Up to now, very little of the work done has addressed this problem. We can cite Juels *et al.* [14], Molnar and Wagner [15], and Weis [21]. In the following sections we will analyse privacy issues (traceability) at each of the three layers of the communication model.

# 3 Traceability at the Application Layer

## 3.1 Identification Protocols

We explained in Section 2.3 that the traceability issue has to be considered in each of the three layers of the communication model. Up until now, only the application layer has been extensively studied (e.g., [7, 13, 15, 16, 19]). Broadly, all these works are based on the fact that the information sent by the tag to the reader changes at each identification. This information is either the identifier of the tag or an encrypted value of it. What differentiates the existing protocols is the way in which this information is refreshed between two identifications. Usually, during this process, the reader supplies the tag with the next value to send (new identifier or new ciphertext) or data allowing the tag to carry out the refreshment by itself. So, we can represent many of the RFID protocols by a 3-round protocol whose exchanged messages contain respectively the request, the identifier of the tag, and data to refresh the identifier.

Therefore, in order to avoid traceability, the information sent by the tag needs to be indistinguishable (by an adversary) from a *random* value and to be *used only once*. If the reader is involved in the refreshment process, it can voluntarily send information which is not indistinguishable from a random value. We characterise the RFID protocols according to whether the reader is or is not involved in the refreshment process.

In the first case, the tag must be able to generate new information by itself. For example, Ohkubo *et al.* [16] propose an RFID protocol where the tag can refresh its identifier by itself by using two hash functions. Obviously, the identifier is used only once since the tag changes it by itself as soon as an identification is completed. Whilst this scheme is proven secure from the point of view of privacy, it suffers from scalability issues. Avoine and Oechslin [3] however have shown that complexity can be significantly reduced using a time-memory trade-off.

In the case where the reader is involved in the regeneration of the information, we need to be sure that this information is indistinguishable (by an attacker) from a random value, but also that this information is used only once. Many of the existing protocols suffer from these two problems. This shows the difficulty of defining tracing-resistant RFID protocols if the tag depends on the reader for generating such values. To illustrate our point, we present below an attack against the protocol of Henrici and Müller [7].

### 3.2 Case Study: Protocol of Henrici and Müller

The principle of the protocol is as follows: after the personalisation phase, the tag contains its current identifier (ID), the current session number $i$ and the last successful session number $i^*$. When the system is launched, the database contains a list of entries, one for each tag it manages. Each entry contains the same data as is stored in the tag, augmented by a hash value of ID, $h(\text{ID})$, which constitutes the database primary key and other additional data. ID and $i$ are set up with random values and $i^*$ equals $i$. The identification process is as follows (see Fig. 2):

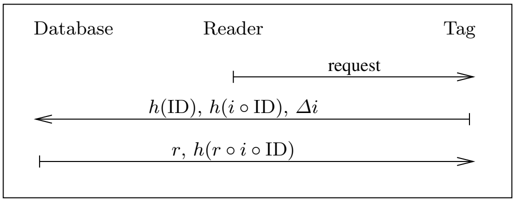

Database
Reader
Tag
request
$h(\text{ID})$, $h(i \circ \text{ID})$, $\Delta i$
$r$, $h(r \circ i \circ \text{ID})$

**Fig. 2.** Protocol of Henrici and Müller

1. The reader sends a request to the tag.
2. The tag increases its current session number by one. It then sends back $h(\text{ID})$, $h(i \circ \text{ID})$ and $\Delta i := i - i^*$ to the reader which forwards the values to the

database. Here $\circ$ is a “suitable conjunction function”; “A simple exclusive-or function is adequate for the purpose” [7]. $h(\text{ID})$ allows the database to recover the identity of the tag in its data; $h(i \circ \text{ID})$ aims at thwarting replay attacks and $\Delta i$ is used by the database to recover $i$ and therefore to compute $h(i \circ \text{ID})$.

3. The database checks the validity of these values according to its recorded data. If all is fine, it sends a random number $r$ and the value $h(r \circ i \circ \text{ID})$ to the tag, through the reader.
4. Since the tag knows $i$ and ID and receives $r$, it can check whether or not $h(r \circ i \circ \text{ID})$ is correct. If this is case, the tag calculates its new identifier $\text{ID}' := r \circ \text{ID}$ and $i^* := i$, which is used in the next identification. Otherwise it does not calculate $\text{ID}'$.

Note that due to resilience considerations, an entry is not erased when the database has replied to the tag, but a copy is kept until the next correct session: if the third step fails, the database will still be able to identify the tag the next time with the “old” entry. Thus two entries per tag are used in turn.

**Attack Based on the Non-randomness of the Sent Information.** The first attack consists of tracking a tag, in a probabilistic way, taking advantage of the side channel supplied by $\Delta i$. Indeed, since the tag increases its value $i$ every time it receives a request (Step 2), even if the identification finally fails, while $i^*$ is updated only when the identification succeeds (Step 4), an attacker may interrogate the tag several times to abnormally increase $i$ and therefore $\Delta i$. Thanks to the fact that this value is sent in the message from the tag to the reader, the attacker is then able to (probabilistically) recognise his target later according to this value: if the adversary later interrogates a tag that sends an abnormally high $\Delta i$, he concludes that this is his target.

**Attack Based on Refreshment Avoidance.** Another attack consists of corrupting the hash value sent from the reader to the tag. When this value is not correct, “the message is discarded and no further action is taken” [7], so the tag does not refresh its identifier. Note, however, that it is difficult to modify this message due to the fact that the communication channel is wireless. We therefore propose a practical variant of this attack: when a reader interrogates a tag, the attacker interrogates this tag as well before the reader carries out the third step. Receiving the request from the attacker, the tag increases $i$. Consequently, the hash value sent by the reader seems to be incorrect since $i$ has now changed. More generally, an attacker can always trace a tag between two correct identifications. In other words, this attack is possible because the signal to refresh the identifier comes from the outside of the tag, i.e., the reader.

**Attack Based on Database Desynchronisation.** A more subtle and definitive attack consists of desynchronising the tag and the database. In order to do this, when a reader queries a tag, the attacker performs the third step of the identification before the reader does it. The random value $r$ sent in the third step by the attacker is the neutral element of the operation $\circ$. Typically, if $\circ$ is the exclusive-or operation (according to [7]), the attacker replaces $r$ by the null bit-string and replaces $h(r \circ i \circ \mathrm{ID})$ by $h(i \circ \mathrm{ID})$ obtained by eavesdropping the second message of the current identification. We have trivially $h(\mathbf{0} \oplus i \oplus \mathrm{ID}) = h(i \oplus \mathrm{ID})$. Hence, the tag does not detect the attack and computes its new identity $\mathrm{ID}' = \mathbf{0} \oplus \mathrm{ID}$ (which is equal to its "old" identity) and it updates $i^*$. Therefore, in the next identification, the tag and the database will be desynchronised, since the tag computes the hash value using the "old" ID and the "new" $i^*$ whereas the database checks the hash value with the "old" ID and the "old" $i^*$: the test fails and the received message is discarded. Consequently, the database will never send the signal to refresh the tag's identifier and the tag is definitively traceable.

# 4 Traceability at the Communication Layer

## 4.1 Singulation Protocols

With several entities communicating on a same channel, we need to define some rules to avoid collisions and therefore to avoid information loss. This arises in RFID systems because when a reader sends a request, all the tags in its field reply simultaneously, causing collisions. The required rules are known as the *collision avoidance* protocol. The tags' computational power is very limited and they are unable to communicate with each other. Therefore, the readers must deal with the collision avoidance themselves, without the help of tags. Usually, they consist of querying the tags until all identifiers are obtained. We say that the reader performs the *singulation* of the tags because it can then request them selectively, without collision, by indicating the identifier of the queried tag in its request.

The collision avoidance protocols which are used in the current RFID systems are often (non-open source) proprietary algorithms. Therefore, obtaining information on them is difficult. Currently, several open standards appear and they are used more and more instead of proprietary solutions. We distinguish the EPC[2] family [4] from the ISO family [9]. Whether they are is EPC or ISO, there are several collision avoidance protocols. Choosing one of them depends (in part) on the used frequency. EPC proposes standards for the most used frequency, i.e., 13.56MHz and 860-930MHz. ISO proposes standards from 18000-1 to 18000-6 where 18000-3 corresponds to the frequency 13.56MHz, and 18000-6 corresponds to the frequency 860-960MHz. We have two main classes of collision avoidance protocols: the deterministic protocols and the probabilistic protocols. Usually, we use the probabilistic protocols for systems using the frequency 13.56MHz, and the deterministic protocols for systems using the frequency 860-960MHz because they are more efficient in this case. After describing both the deterministic and the probabilistic collision avoidance protocols in Section 4.2 and 4.3, we will then analyse the traceability issues of these protocols.

[2] Electronic Product Code.

### 4.2 Deterministic Protocols

Deterministic protocols rely on the fact that each tag has a unique identifier. If we want the singulation process to succeed, the identifiers must stay unchanged until the end of the process. In the current tags, the identifiers are set by the manufacturer of the tag and written in the tag's ROM. In the usual RFID systems, there is no exchange after the singulation because the reader has obtained the expected information, i.e., the identifiers of the tags which are in its field. Below, we use *singulation identifier* to denote such an identifier, or more simply *identifier* where there is no ambiguity with the identifier of the application layer. We give an example of deterministic collision avoidance protocol called *tree walking*.

Suppose tags have a unique identifier of bit-length $\ell$. All the possible identifiers can be visualised by a binary tree of depth $\ell$. A node at depth $d$ in this tree can be uniquely identified by a binary prefix $b_1b_2...b_d$. The reader starts at the root of the tree and performs a recursive depth-first search. So, at node $b_1b_2...b_d$, the reader queries all tags whose serial numbers bear this prefix, the others remain silent. The tags reply with the $d+1$-st bit of their serial number. If there is a collision, the reader restarts from the child of the prefix. When the algorithm reaches a leaf, it has detected a tag. The full output of the algorithm is a list of all tags within its field.

### 4.3 Probabilistic Protocols

The probabilistic protocols are usually based on a time-division multiple access protocol, called *Aloha*. We describe one of the variants of Aloha, namely the slotted Aloha. In the slotted Aloha, the access to the communication channel is split into time slots. In general, the number of slots is chosen randomly by the reader which informs the tags that they will have $n$ slots to answer to its singulation request. Each tag randomly chooses one slot among the $n$ and responds to the reader when its slot arrives. If $n$ is not sufficiently large with regard to the number of tags which are present, then some collisions occur. In order to recover the missing information, the reader interrogates the tags one more time. It can mute the tags which have not brought out collisions (*switched-off* technique) by indicating their identifiers or the time slots during which they transmitted. Also, according to the number of collisions, it can choose a more appropriate $n$.

Note that although all the usual tags have a (unique) singulation identifier, this condition is not fundamentally required for Aloha, but is desirable for efficiency reasons [11]. Without using these identifiers, the exchange of information of the application layer is carried out during the singulation because the reader cannot communicate anymore with the tag when the singulation process is completed. Note also that the singulation seems *atomic* from the tag's view: whilst a tag must reply to the reader several times when the tree walking is used, the tag can answer only once when no collision occurs with the Aloha protocol. In the case where the response brings out a collision, the reader restarts a new singulation process with possibly a larger $n$. On the other hand, if the switched-off technique is used, then the protocol is not atomic anymore.

### 4.4 Threats Due to an Uncompleted Singulation Session

It is clear that deterministic collision avoidance protocols relying on the static identifiers give an adversary an easy way to track the tags. To avoid traceability, the identifiers would need to be dynamic. However if the identifier is modified during the singulation process, singulation becomes impossible. So we introduce the concept of *singulation session* as being the set of exchanges between a reader and a tag which are needed to singulate the latter. When the session does not finish, due to failures or attacks, we say that the session stays *open*.

Since the singulation identifier cannot be changed during a session, the idea, to avoid traceability, is to use an identifier which is different for each session. The fact that the tag can be tracked during a session is not really a problem due to the shortness of such a session. In practice, the notion of singulation session already informally exists because the readers usually send a signal at the beginning and end of a singulation. Unfortunately, there is no reason to trust the readers to correctly accomplish this task. In particular, a malicious reader can voluntarily keep a session open to track the tag thanks to the unchanged identifier. This attack cannot be avoided when the signals come from the reader and not from the tag itself.

Contrary to what we usually think, using a probabilistic protocol based on Aloha does not directly solve the traceability problem at the communication layer. Because, apart from the (inefficient) Aloha-based protocols which do not use the switched-off technique, the concept of singulation session is also needed with probabilistic singulation protocols. Indeed, after having queried the tags, the reader sends an acknowledgement (either to each tag or to all the tags) to indicate which tags should retransmit (either the reader acknowledges the identifiers of the tags it has successfully read, or it indicates the numbers of the slots where a collision occurred). In the case where the identifiers are used, the fact that a singulation session stays open allows an adversary to track the tags. In the case where the acknowledgement does not contain the identifiers but contains instead the numbers of the slots where a collision occurred, then an attack relying on these slots is also possible, as follows: an adversary who is in the presence of a targeted tag sends it a (first) singulation request with the number of potential slots $n$. Assume the tag answers during the randomly chosen slot $s_{\text{target}}$. The tag being alone, the reader can easily link $s_{\text{target}}$ to the targeted tag. The reader keeps the session opened. Later, when the adversary meets a set of tags potentially containing its target, it interrogates the tags again, indicating that only tags which transmitted during $s_{\text{target}}$ must retransmit: if a tag retransmits, there is a high probability, depending on $n$ and the number of tags in the set, that it is the target of the adversary since another tag will respond to the (2nd) singulation request during $s_{\text{target}}$ if, and only if, its last session stayed opened and it transmitted during $s_{\text{target}}$.

Whether we consider deterministic or probabilistic protocols, it is fundamental that singulation sessions cannot stay open. The tag needs to be able to detect such sessions and to close them by itself. In other words, the signal needs to be internal to the tag.

Consequently, we suggest using an internal timeout to abort singulation sessions with abnormal duration. Thus, the tag starts the timeout when the singulation session begins (i.e., when it receives the first request of a singulation session). When the timeout expires, the current session is considered as aborted.

Implementation of such a timeout strongly depends on the practical system, e.g., the timeout could be a capacitor. When the tag receives the first request of a singulation session, it generates a fresh identifier and loads its capacitor. Then, each time it is queried (such that the request is not the first one of a session), it checks whether its capacitor is empty. If this is the case, the tag erases its identifier and does not answer until the next "first" request. If it is not the case, it follows the protocol. Note that the duration of the capacitor may be less than the duration of a singulation session if this capacity is reloaded periodically and the number of reloads is counted.

### 4.5 Threats Due to Lack of Randomness

Changing the identifier of the tag is essential but does not suffice because these identifiers need to be perfectly random not to supply an adversary with a source of additional information. The use of a cryptographically secure pseudo-random number generator (PRNG), initialised with a different value for every tag, is indispensable for avoiding traceability. Of course, singulation must rely only on this random identifier without requiring other characteristic data of the tag.

In the tree walking case, [5] proposes for instance using short singulation identifiers which are refreshed for each new singulation using a PRNG. The used identifiers are short for efficiency reasons since there are usually only few tags in a given field. However, if the number of tags in the field is large, the reader can impose the use of additional static identifiers, available in the tag, set by the manufacturer! The benefit of using PRNG is therefore totally null and void.

In the case of Aloha, if the singulation identifiers do not appear in the acknowledgement sent by the readers, they do not directly bring information to an adversary. On the other hand, they supply much information through a side channel if we analyse how the slot is chosen by the tag. If this is randomly picked, it will not supply useful information to the adversary, but a non uniform distribution can open the door to attacks. Unfortunately this is the case with the current existing standards and protocols.

In order to illustrate our point, we can analyse the collision avoidance protocol proposed by Philips for its tag ICode1 Label IC [17] using the 13.56MHz frequency. It contains a 64 bit identifier of which only 32 are used for the singulation process, denoted by $b_1 ... b_{32}$. Although the tag does not have a PRNG, the implemented collision avoidance protocol is probabilistic. The choice of the time slot depends on the identifier of the tag and data sent by the reader. When the reader queries a tag, it sends a request containing: the number of slots $n$ which the tags can use, where $n \in \{2^0, 2^1, ..., 2^8\}$, and a value $h \in 0, ..., 25$ called *hash value*. The selection of the time slot $s_i$ is done as follows:

$$s_i := \mathrm{CRC8}(b_{h+1}...b_{h+8} \oplus prev) \oplus n$$

where CRC8 is a *Cyclic Redundancy Check* with generator polynomial $x^8 + x^4 + x^3 + x^2 + 1$ and where *prev* is the output of the previous CRC8, initialised with `0x01` when the tag enters the field of a reader. Hence, an adversary can easily track a tag according to the slot chosen by the tag, if he always sends the same values $h$ and $n$. One way to proceed is as follows.

An adversary sends to his (isolated) targeted tag a request with the number of slots $n$ and the hash value $h$ . The tag responds during slot $s_{\mathrm{target}}$. When he meets a set of $m$ tags, the adversary wants to know if his target is here. In order to do this, he sends a singulation request containing the same $n$ and $h$. If no tag responds during $s_{\mathrm{target}}$ then the target is not included in the set of tags. However, the conditional probability that the tag is in the set given that at least one tag answers during slot $s_{\mathrm{target}}$ is

$$P(n, m, p) = \frac{p}{p + (1-p)(1 - (\frac{n-1}{n})^m)},$$

where $p$ is the probability that the target is in the set[3].

Consequently, choosing the identifier, in the case of the three walking-based protocols, and choosing the time slot, in the case of the Aloha-based protocols, must be done using a cryptographically secure PRNG. Otherwise, an adversary may take advantage of the distorted distribution in order to track his target in a probabilistic way, or worse, to recover its identifiers as with the ICode1 tag.

# 5 Traceability at the Physical Layer

The physical signals exchanged between a tag and a reader can allow an adversary to recognise a tag or a set of tags even if the information exchanged can not be understood. All efforts to prevent traceability in the higher layers may be rendered useless if no care is taken at the physical layer.

## 5.1 Threats Due to Diversity of Standards

The parameters of radio transmission (frequency, modulation, timings, etc) follow given standards. Thus all tags using the same standard should send very similar signals. Signals from tags using different standards are easy to distinguish. A problem arises when we consider sets of tags rather than a single tag. In a few years, we may all be walking around with many tags in our belongings. If several standards are in use, each person may have a set of tags with a

[3] Note that in the particular case of the ICode1 tag, where the CRC-8 is applied on a 8-bit word, we can actually recover 8 bits of the identifier by sending only one singulation request! Therefore, by sending 4 requests with respectively $h = 0$, $h = 8$, $h = 16$, and $h = 24$, the adversary will be able to recover the 32 bits of the tag's singulation identifier.

characteristic mix of standards. This mix of standards may allow a person to be traced. This method may be especially good at tracing certain types of persons, like military forces or security personnel.

To reduce the threats of traceability due to characteristic groups of tags it is thus of paramount importance to reduce the diversity of the standards used in the market. Note that even if it is possible to agree on a single standard to use when RFID tags become popular, there will be times when a standard for a new generation of tags will be introduced. During the period of transition it will be possible to trace people due to characteristic mixes of old and new tags.

### 5.2 Threats Due to Radio Fingerprinting

Radio fingerprinting is a technique that has been used in mobile telephony to recognise cloned phones. By recording characteristic properties of the transmitted signals it is possible to tell a cloned cell-phone from the original one. Small differences in the transient behaviour at the very beginning of a transmission allows for the identification of transceivers even if they are of the same brand and model [20]. In the case of RFID tags, there will be too many tags in circulation to make it possible to distinguish a single tag from all other tags of the same model. Nevertheless, there will be several manufacturers in the market and their tags will have different radio fingerprints. It will thus be possible to trace a person by a characteristic mix of tags from different manufacturers.

Preventing traceability through radio fingerprinting seems quite difficult. There is no benefit for the manufacturers to produce tags that use exactly the same technology, producing the same radio fingerprint. Much more likely, manufacturers will experiment with different technologies in order to produce tags that have either better performance, price or size.

## 6 Conclusion

As we have shown in this paper, until now privacy issues in RFID systems have only been considered in classical cryptographic models with little concern for the practical effects on traceability when the theory is put into practice. We have shown that, contrary to the three basic concepts of cryptography, i.e., confidentiality, authentication, and integrity, traceability has to be considered with regard to the communication architecture. Thus, to create a fully privacy-friendly RFID system, privacy has to be ensured at each of the three layers of the communication model. We have described the threats that affect each of these layers and we have given some practical examples in order to illustrate our theories. We have included recommendations or solutions for each of these layers, although we have found that ensuring both privacy and scalability at the application layer seems difficult without sacrificing the low cost constraint.

## Acknowledgments

The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

## References

1. Auto-ID Labs. `http://www.autoidlabs.org`.
2. G. Avoine. Privacy issues in RFID banknote protection schemes. *Smart Card Research and Advanced Applications – CARDIS*, pp. 33–48, Kluwer, 2004.
3. G. Avoine and Ph. Oechslin. A scalable and provably secure hash based RFID protocol. *International Workshop on Pervasive Computing and Communications Security – PerSec 2005*, pp. 110–114, IEEE, 2005.
4. Electronic Product Code Global Inc. `http://www.epcglobalinc.org`.
5. EPC. Draft protocol specification for a 900 MHz class 0 radio frequency identification tag. `http://www.epcglobalinc.org`, February 2003.
6. M. Feldhofer, S. Dominikus, and J. Wolkerstorfer. Strong authentication for RFID systems using the AES algorithm. *Workshop on Cryptographic Hardware and Embedded Systems – CHES 2004*, LNCS 3156, pp. 357–370, Springer, 2004.
7. D. Henrici and P. Müller. Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers. *Workshop on Pervasive Computing and Communications Security – PerSec 2004*, pp. 149–153, IEEE, 2004.
8. International Organization for Standardization. `http://www.iso.org`.
9. ISO/IEC 18000. Automatic identification – radio frequency identification for item management – communications and interfaces. `http://www.iso.org`.
10. ISO/IEC 18000-1. Information technology AIDC techniques – RFID for item management – air interface, part 1 – generic parameters for air interface communication for globally accepted frequencies. `http://www.iso.org`.
11. ISO/IEC 18000-3. Information technology AIDC techniques – RFID for item management – air interface, part 3 – parameters for air interface communications at 13.56 MHz. `http://www.iso.org`.
12. ISO/IEC 7498-1:1994. Information technology – open systems interconnection – basic reference model: The basic model. `http://www.iso,org`, November 1994.
13. A. Juels. "yoking-proofs" for RFID tags. *Workshop on Pervasive Computing and Communications Security – PerSec 2004*, pp. 138–143, IEEE, 2004.
14. A. Juels, R. Rivest, and M. Szydlo. The blocker tag: Selective blocking of RFID tags for consumer privacy. *Conference on Computer and Communications Security – ACM CCS*, pp. 103–111, ACM, 2003.
15. D. Molnar and D. Wagner. Privacy and security in library RFID: Issues, practices, and architectures. *Conference on Computer and Communications Security – ACM CCS*, pp. 210–219, ACM, 2004.
16. M. Ohkubo, K. Suzuki, and S. Kinoshita. Cryptographic approach to "privacy-friendly" tags. *RFID Privacy Workshop*, MIT, MA, USA, November 2003.
17. Philips. I-Code1 Label ICs protocol air interface, May 2002.
18. J. Saito, J.-C. Ryou, and K. Sakurai. Enhancing privacy of universal re-encryption scheme for RFID tags. *Embedded and Ubiquitous Computing – EUC 2004*, LNCS 3207, pp. 879–890, Springer, 2004.

19. S. Sarma, S. Weis, and D. Engels. RFID systems and security and privacy implications. *Cryptographic Hardware and Embedded Systems – CHES 2002*, LNCS 2523, pp. 454–469, Springer, 2002.
20. J. Toonstra and W. Kinsner. Transient analysis and genetic algorithms for classification. *IEEE WESCANEX 95. Communications, Power, and Computing*, volume 2, pp. 432–437, IEEE, 1995.
21. S. Weis. Security and privacy in radio-frequency identification devices (master thesis), May 2003.

# Information-Theoretic Security Analysis of Physical Uncloneable Functions

P. Tuyls, B. Škorić, S. Stallinga, A.H.M. Akkermans, and W. Ophey

Philips Research Laboratories, Prof. Holstlaan 4,
5656 AA Eindhoven, The Netherlands

**Abstract.** We propose a general theoretical framework to analyze the security of Physical Uncloneable Functions (PUFs). We apply the framework to optical PUFs. In particular we present a derivation, based on the physics governing multiple scattering processes, of the number of independent challenge-response pairs supported by a PUF. We find that the number of independent challenge-response pairs is proportional to the square of the thickness of the PUF and inversely proportional to the scattering length and the wavelength of the laser light. We compare our results to those of Pappu and show that they coincide in the case where the density of scatterers becomes very high. Finally, we discuss some attacks on PUFs, and introduce the Slow PUF as a way to thwart brute force attacks.

**Keywords:** Physical Uncloneable Function, entropy, speckle pattern, Challenge-Response Pair.

## 1 Introduction

### 1.1 Physical Uncloneable Functions

A 'Physical Uncloneable Function' (PUF) is a function that is realized by a physical system, such that the function is easy to evaluate but the physical system is hard to characterize [1, 2]. PUFs have been proposed as a cost-effective way to produce uncloneable tokens for identification [3]. The identification information is contained in a cheap, randomly produced (i.e. consisting of many random components), highly complicated piece of material. The secret identifiers are read out by performing measurements on the physical system and performing some additional computations on the measurement results. The advantage of PUFs over electronic identifiers lies in the following facts: (1) Since PUFs consist of many random components, it is very hard to make a clone, either a physical copy or a computer model, (2) PUFs provide inherent tamper-evidence due to their sensitivity to changes in measurement conditions, (3) Data erasure is automatic if a PUF is damaged by a probe, since the output strongly depends on many random components in the PUF. Additionally one can extract cryptographic keys from a PUF. This makes PUFs attractive for Digital Rights Management (DRM) systems.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 141–155, 2005.
© Springer-Verlag Berlin Heidelberg 2005

The physical system is designed such that it interacts in a complicated way with stimuli (*challenges*) and leads to unique but unpredictable *responses.* Hence, a PUF is similar to a keyed hash function. The key is the physical system consisting of many "random" components. In order to be hard to characterize, the system should not allow efficient extraction of the relevant properties of its interacting components by measurements. Physical systems that are produced by an uncontrolled production process, i.e. one that contains some randomness, turn out to be good candidates for PUFs. Because of this randomness, it is hard to produce a physical copy of the PUF. Furthermore, if the physical function is based on many complex interactions, then mathematical modeling is also very hard. These two properties together are referred to as *Uncloneability.* From a security perspective the uniqueness of the responses and uncloneability of the PUF are very useful properties. Because of these properties, PUFs can be used as unique identifiers for smart-cards and credit cards or as a 'cheap' source for key generation (common randomness) between two parties.

At the moment there are several main candidates: optical PUFs [3, 4], silicon PUFs [2, 5], coating PUFs [6] and acoustic PUFs [6]. Silicon PUFs make use of production variation in the properties of logical gates. When these are probed at frequencies that are out of spec, a unique, unpredictable response is obtained in the form of delay times. Coating PUFs are integrated with an IC. The IC is covered with a coating, which is doped with several kinds of particles of random size and shape with a relative dielectric constant differing from the dielectric constant of the coating matrix. An array of metal sensors is laid down between the substrate and the passivation layer. A challenge corresponds to a voltage of a certain frequency and amplitude applied to the sensors at a certain point of the sensor array. The response, i.e. the capacitance value, is then turned into a key. In an acoustic PUF, one measures the response of a token to an acoustic wave. An electrical signal is transformed to a mechanical vibration through a transducer. This vibration propagates as a sound wave through the token and scatters on the randomly distributed inhomogeneities. The reflections are measured by another transducer which converts the vibration back into an electric signal. It turns out that the reflections are unique for each token.

Optical PUFs contain randomly distributed light scattering particles. A picture of an optical PUF and its reading device is shown in Fig. 1. They exploit the

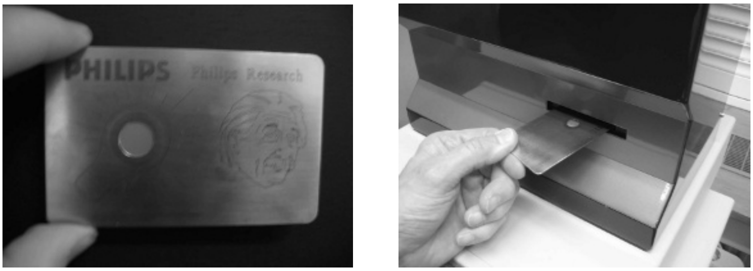


**Fig. 1.** Left: Card equipped with an optical PUF. Right: Reading device

uniqueness of speckle patterns that result from multiple scattering of laser light in a disordered optical medium. The challenge can be e.g. the angle of incidence, focal distance or wavelength of the laser beam, a mask pattern blocking part of the laser light, or any other change in the wave front. The response is the speckle pattern. An input-output pair is usually called a *Challenge-Response Pair* (CRP). Physical copying is difficult for two reasons: (i) The light diffusion obscures the locations of the scatterers. At this moment the best physical techniques can probe diffusive materials up to a depth of $\approx$10 scattering lengths [7]. (ii) Even if all scatterer locations are known, precise positioning of a large number of scatterers is very hard and expensive, and this requires a process different from the original randomized process. Modeling, on the other hand, is difficult due to the inherent complexity of multiple coherent scattering [8]. Even the 'forward' problem turns out to be hard[1].

The goal of this paper is to show how cryptographic tools based on (classical) physical functions can be modeled and rigorously analyzed in a cryptographic context. We derive an information-theoretic framework for PUFs and investigate the security level of optical PUFs. More in particular, we analyze the number of *independent* CRPs of a PUF, i.e. CRPs whose responses are not predictable using previously obtained CRPs. We derive a formula that gives the number of independent CRPs supported by an optical PUF in terms of its physical parameters. In section 2.1, we derive the model starting from the physics of multiple scattering. The security analysis, and in particular the computation of the number of independent CRPs, is presented in section 3. Finally, in section 4 we discuss brute force attacks. In particular, we introduce the 'slow PUF' as a way of thwarting these attacks.

### 1.2 Applications

Optical PUFs are well suited for identification, authentication and key generation [3, 6]. The goal of an identification protocol is to check whether a specific PUF is present at the reader. The goal of an authentication protocol is to ensure that received messages originate from the stated sender. For authentication it is therefore the objective to extract the same cryptographic key from the PUF as the one that is stored at the Verifier's database during enrollment, while for identification it is sufficient if the response is close to the enrolled response.

In order to use PUFs for above mentioned purposes they are embedded into objects such as smartcards, creditcards, the optics of a security camera, etc., preferably in an inseparable way, meaning that the PUF gets damaged if an attacker attempts to remove the PUF. This makes the object in which a PUF is embedded uniquely identifiable and uncloneable. Secret keys can be derived from a PUF's output [6] by means of protocols similar to those developed in the context of biometrics [10, 11].

[1] Given the details of all the scatterers, the fastest known computation method of a speckle pattern is the transfer-matrix method [9]. It requires in the order of $N_{\rm mod}^3 d/\lambda$ operations (see section 3.2 for the definition of $N_{\rm mod}$, $d$ and $\lambda$).

The usage of a PUF consists of two phases: enrolment and verification. During the enrolment phase, the Verifier produces the PUF and stores an initial set of CRPs securely in his database. Then the PUF is embedded in a device and given to a user. The verification phase starts when the user presents his device to a terminal. The Verifier sends a randomly chosen PUF challenge from his database to the user. If the Verifier receives the correct response from the device, the device is identified. Then this CRP is removed from the database and will never be used again.

If, additionally, the device and the Verifier need to exchange secret messages, a secure authenticated channel is set up between them, using a session key based on the PUF response. We present the following protocols.

**Identification Protocol:**

- User: Puts his card with PUF in the reader and claims its ID.
- Verifier: Randomly chooses a challenge C from his CRP database and sends it to the User.
- Reader: Challenges the PUF with the Challenge $C$, measures the Response $R$ and computes an identifier $S'$. $S'$ is sent back to the Verifier.
- Verifier: Checks whether $S'$ equals the identifier $S$ stored in his database during enrollment. Then he removes the pair $(C, S)$ from his database and never uses it again.

We note that the security of this protocol relies on the fact that an attacker who has seen $(C_1, S_1)$ cannot predict the identifier $S_2$ corresponding to the challenge $C_2$, and on the fact that the PUF supports a large number of CRPs.

**Authentication Protocol:**

- User: Puts his card with PUF in the reader and claims its ID.
- Verifier: Randomly chooses a challenge $C$ from his CRP database and sends it to the User, together with a random nonce $m$.
- Reader: Challenges the PUF with the Challenge $C$, measures the Response $R$ and computes a key $S'$. $M_{S'}(m)$ is sent to the Verifier, where $M_{S'}(m)$ denotes a MAC on $m$, using the key $S'$.
- Verifier: Computes $M_S(m)$ with the key $S$ stored in his database and compares it with $M_{S'}(m)$. If they are equal, then $S = S'$ with very high probability. The key $S$ is then used to MAC and/or encrypt all further messages.

The security of this scheme depends on the fact that (when the key S is unknown) the MAC $M_S(m)$ is unpredictable given that the attacker has seen the MAC on a message $m_1 \neq m$.

### 1.3 Notation

We introduce the following notation. The power of the laser is denoted by $P$ and its wavelength by $\lambda$. The thickness of the PUF is denoted by $d$. Scattering is assumed to be elastic[2], with mean free path[3] $\ell$. We further assume diffusive scattering, i.e. $\lambda \ll \ell \ll d$. The illuminated area of the PUF is $A = W^2$. For simplicity the output surface area is also taken to be $A$. The detector needs time $\triangle t$ to record a speckle pattern. The following numerical values will be used by way of example: $W = 1$ mm, $d = 1$ mm, $\ell = 10$ $\mu$m, $\lambda = 500$ nm, $P = 1$ mW, $\triangle t = 1$ ms. Note that the total area of the PUF ($A_{\mathrm{PUF}}$) can be much larger than the *illuminated area* $A$. We will use $A_{\mathrm{PUF}}$ =5cm$^2$. Throughout this paper we will mostly calculate properties of one specific volume $Ad$, and only afterwards adjust our results by a factor $A_{\mathrm{PUF}}/A$. This effectively amounts to treating the PUF of area $A_{\mathrm{PUF}}$ as a collection of independent PUFs of area $A$.

# 2 Information Theory of PUFs

## 2.1 General PUF Model

A PUF can be modeled as a function mapping challenges to responses. We denote the challenge space by $\mathcal{A}$ and the response space by $\mathcal{R}$. A PUF is then a parametrized function $\pi_K : \mathcal{A} \to \mathcal{R}$ whose behaviour is determined by the physical interactions. The parameter $K$ belongs to the parameter space $\mathcal{K}$ and is determined by a large number of random variables, namely the physical structure of the PUF. Hence, the space $\mathcal{K}$ models the space of all possible PUFs and there is a one to one correspondence between the elements of $\mathcal{K}$ and the set of PUFs. In order to express the uncertainy about the random variable $K$, described by the probability measure $\eta$, we use the Shannon entropy $\mathsf{H}_\eta(K)$

$$\mathsf{H}_\eta(K) = -\sum_{i=1}^{|\mathcal{K}|} \eta(K_i) \log \eta(K_i), \tag{1}$$

where $|\mathcal{K}|$ denotes the size of $\mathcal{K}$. Sometimes we will also need the conditional entropy $\mathsf{H}(K|R)$, representing the uncertainty about $K$ given that one knows a response $R$. The mutual information between $K$ and $R$ is denoted as $\mathbf{I}(K;R)$. For the precise definitions of these notions we refer the reader to textbooks on information theory, e.g. [12]. The notation “log” denotes the logarithm with base 2.

One of the important quantities used in this paper is the size of the parameter space $\mathcal{K}$, representing the information content of a PUF. Therefore we have to make a precise definition of this quantity. To this end, we start by defining some abstract notions and later we make those notions concrete by means of an example. First, we present a brief computation that motivates the definitions that we introduce. The amount of information about the PUF that is revealed by one response is given by the mutual information $\mathbf{I}(K;R) = \mathsf{H}(K) - \mathsf{H}(K|R)$.

[2] Elastic scattering means that the photons do not lose energy when they are scattered.

[3] The mean free path is the average distance travelled by the photons between two scattering events.

We show that the mutual information is actually equal to $\mathsf{H}(R)$. First we observe that $\mathsf{H}(K) = \mathsf{H}(K, R)$, since given the PUF, the speckle pattern is fixed. Using the identity $\mathsf{H}(K, R) = \mathsf{H}(R) + \mathsf{H}(K|R)$ we obtain

$$\mathbf{I}(K; R) = \mathsf{H}(R). \tag{2}$$

### 2.2 Definitions

The information content of a PUF ($\mathsf{H}_\eta(K)$) and of its output ($\mathsf{H}(R)$) depends on the measurements that can be performed on the system. This is formalized as follows. We identify a measurement with its possible outcomes.

**Definition 1.** *A measurement $\mathcal{M}$ is a partition $\{R_1, \cdots, R_m\}$ of $\mathcal{K}$.*

Here $R_j$ is the set in $\mathcal{K}$ containing all PUFs that produce outcome $j$ upon measurement $\mathcal{M}$, and $m$ is the number of possible outcomes. Two measurements give more (refined) information than one. The composition of two measurements is denoted as $\mathcal{M}_1 \vee \mathcal{M}_2$ and is defined as follows:

$$\mathcal{M}_1 \vee \mathcal{M}_2 = \{R_i^{(1)} \cap R_j^{(2)}\}_{i,j=1}^m. \tag{3}$$

$R_j^{(i)}$ is the set of all PUFs that produce outcome $j$ upon measurement $\mathcal{M}_i$. By induction this definition extends to composition of more than two measurements.

**Definition 2.** *Let $\eta$ denote a probability measure on $\mathcal{K}$. The information obtained about a system $K \in \mathcal{K}$ by performing measurement $\mathcal{M}$ is defined as*

$$\mathsf{h}_{\mathcal{M}}(\mathcal{K}) = -\sum_{i=1}^{m} \eta(R_i) \log \eta(R_i).$$

We note that the following monotonicity property can easily be proven

$$\mathsf{h}_{\mathcal{M}_1 \vee \mathcal{M}_2} \geq \mathsf{h}_{\mathcal{M}_1}, \tag{4}$$

which corresponds to the fact that finer measurements give more information. Due to the physics, one will often only have a finite set $\mathcal{A}$ of challenges available. This set restricts the amount of information that can be obtained.

**Definition 3.** [4] *Given the set $\mathcal{A}$ of possible measurements, the total amount of information that can be obtained about a system $\mathcal{K}$ is*

$$\mathsf{h}_{\mathcal{A}}(\mathcal{K}) = \sup_{\mathcal{M}_1, \ldots, \mathcal{M}_q \in \mathcal{A};\ 0 < q \leq |\mathcal{A}|} \mathsf{h}_{\mathcal{M}_1 \vee \ldots \vee \mathcal{M}_q}(\mathcal{K}).$$

[4] We note that this definition is in agreement with the theory of dynamical systems and dynamical entropy [13].

It follows from the monotonicity property (4) that $\mathsf{h}_{\mathcal{A}}(\mathcal{K}) \le \mathsf{H}(K)$, i.e. the maximum amount of information that can be obtained about a system is upper bounded by the amount of uncertainty one has in the measure $\eta$. If $\eta$ is given by the random measure $\eta(K_i) = 1/|\mathcal{K}|$, we find that $\mathsf{H}(K) = \log(|\mathcal{K}|)$. In the remainder of this text, we will assume that $\eta$ is given by this measure.

Definitions 1 and 2 are very general and apply to many kinds of PUFs. In this framework, the couple $(\mathcal{K}, \mathcal{A})$ has to be specified for a well-defined notion of PUF security. We consider two extreme cases to illustrate the definitions. If $\mathcal{A}$ contains a CRP measurement that distinguishes PUFs perfectly, then the PUF supports only one independent CRP. The opposite extreme case is a set of measurements $\mathcal{A} = \{\mathcal{M}_j\}_{j=1}^{n}$ that can be represented as an extremely coarse partitioning of $\mathcal{K}$, say $|M_1^{(j)}| = |M_2^{(j)}| = |\mathcal{K}|/2$, where the combined measurements $(\mathcal{M}_1 \vee \ldots \vee \mathcal{M}_n)$ suffice to distinguish all elements of $\mathcal{K}$. In this case a minimum of $\log|\mathcal{K}|$ measurements is needed to reveal all details of the PUF. For good PUFs, all available measurements are fuzzy, revealing little about the physical structure.

### 2.3 Optical PUFs

We illustrate Definition 2 for optical PUFs. As the probing light has wavelength $\lambda$, it follows from the theory of electromagnetism [14] that details of size smaller than $\lambda$ are difficult to resolve. It is natural to divide the volume into elements ('voxels') of volume $\lambda^3$. The number of voxels is $N_{\text{vox}} = Ad/\lambda^3$. In the example of section 1.3 we have $N_{\text{vox}} = 8 \cdot 10^9$ and a total number of $4 \cdot 10^{12}$ voxels in the whole PUF. For the sake of simplicity, we assume that light can only distinguish whether a voxel contains a scatterer or not. Hence, the information content of a voxel is at most 1 bit, and the PUF can be represented as a bit string of length $N_{\text{vox}}$. The entropy derived from the probability distribution $\eta$ is[5] $\mathsf{H}(K) = N_{\text{vox}}$.

$\mathcal{A}$ is the full set of non-compound measurements that can be performed by means of a beam of monochromatic light. Combining all these available measurements, the maximum amount of information $\mathsf{h}_{\mathcal{A}}(\mathcal{K})$ that can be extracted from the PUF is $\mathsf{H}_\eta(K) = N_{\text{vox}}$. The couple $(\mathcal{K}, \mathcal{A})$ as defined here is used in the remainder of the text.

## 3 Security Analysis

### 3.1 Security Parameter *C*

The main goal of this paper is to estimate the number of independent CRPs. This number is denoted as $C$. It represents the minimal number of CRP measurements that an attacker has to perform to characterize the PUF.

[5] It is possible to refine this model, taking into account the number of photons taking part in the measurement process. This gives rise to an extra factor proportional to the log of the number of photons. We will not discuss this refinement.

**Definition 4.** *Measurements $\mathcal{M}_1, \ldots, \mathcal{M}_t$ are mutually independent iff*

$$\mathsf{h}_{\mathcal{M}_1 \vee \ldots \vee \mathcal{M}_t} = \mathsf{h}_{\mathcal{M}_1} + \cdots + \mathsf{h}_{\mathcal{M}_t}.$$

Note that $\mathsf{h}_{\mathcal{M}_1 \vee \ldots \vee \mathcal{M}_t} = t \cdot \mathsf{h}_{\mathcal{M}_1}$ if all measurements give the same amount of information, which by symmetry arguments is a reasonable assumption.

Independent measurements are also called *independent CRPs* since responses are implicitly incorporated in definition 4. In words, knowledge of independent CRPs $\{\mathcal{M}_j\}_{j \neq i}$ does not give any information about the response to the $i$'th challenge. The number of independent CRPs is hence naturally defined as

$$C = \frac{\mathsf{h}_{\mathcal{A}}(\mathcal{K})}{\mathsf{h}_{\mathcal{M}}(\mathcal{K})} = \frac{\mathsf{h}_{\mathcal{A}}(\mathcal{K})}{\mathsf{H}(R)}, \tag{5}$$

where $\mathcal{M} \in \mathcal{A}$ and $\mathsf{H}(R)$ denotes the information content of a response. The second equality in (5) follows from (2). As we have already argued that $\mathsf{h}_{\mathcal{A}}(\mathcal{K}) = N_{\text{vox}}$, the remainder of this section focusses on the computation of $\mathsf{H}(R)$.

In practice the independent challenges may turn out to be very complicated combinations of basic challenges. However, for the security analysis it is not necesary to have precise knowledge about them. The number $C$ provides a basic security parameter which is not affected by technological and computational advances. An "adaptive chosen plaintext" attack (in the PUF context: trying to model a PUF by collecting responses to self-chosen challenges) requires at least $C$ speckle pattern measurements, irrespective of the attacker's capabilities.

In practice many mutually *dependent* challenges may be used safely by the verifier. Even if some mutual information exists between the responses, it is computationally hard to exploit it, since that would require a characterisation of the physical function. It is not a priori clear how much mutual information between responses can be tolerated before the system becomes insecure, only that the answer depends on the capabilities of the attacker and that the 'safe' number of challenges is proportional to $C$. Therefore, the best available measure of the security level offered by a PUF is the parameter $C$, the number of challenges that can be used safely if the attacker has infinite computation power.

### 3.2 Speckle Pattern Entropy

In order to define the information content $\mathsf{H}(R)$ of a speckle pattern, we investigate the physics of multiple coherent scattering and speckle formation. Based on the physics, we turn this problem into a counting problem of the distinguishable photon states in the light leaving the PUF. First, we show that the PUF can be modeled as a strongly scattering waveguide of thickness $d$, cross-section $A = W^2$ and scattering length $\ell$, satisfying $\lambda \ll \ell \ll d$. The waveguide allows a number of transversal modes $N_{\text{mod}}$. The scattering process is represented by an $N_{\text{mod}} \times N_{\text{mod}}$ random scattering matrix $S_{ab}$, specifying how much light is scattered from incoming mode $b$ to outgoing mode $a$. Given a single incoming mode, the speckle pattern is fully determined by one column of the $S$-matrix. Hence the question is how much information is contained in one such column.

Then we calculate the speckle pattern entropy in the case where all $S$-matrix elements are independent. This yields an upper bound on $\mathsf{H}(R)$. In this calculation, the finiteness of the speckle pattern entropy is ultimately based on the discretisation of light in terms of photons. Finally, we take correlations between the matrix elements into account to compute a lower bound on $\mathsf{H}(R)$.

**Wave Guide Model**

First, we compute the number of incoming and outgoing modes $N_{\text{mod}}$. The complex amplitude of the electric field at the PUF surface can be represented as

$$E(\boldsymbol{r}) = \int_{|\boldsymbol{q}|\leq k} \frac{\mathrm{d}^2 q}{(2\pi)^2} \tilde{E}(\boldsymbol{q}) e^{i\boldsymbol{q}\cdot\boldsymbol{r}} \ ; \ \tilde{E}(\boldsymbol{q}) = \int_{|x|,|y|\leq W/2} \mathrm{d}^2 r \ E(\boldsymbol{r}) e^{-i\boldsymbol{q}\cdot\boldsymbol{r}}, \qquad (6)$$

where $\boldsymbol{r} = (x, y)$ denotes the position and $\boldsymbol{q} = (q_x, q_y)$ the lateral wave vector. A mode is propagating if the longitudinal ($z$) component of the wave, $q_z = \sqrt{k^2 - \boldsymbol{q}^2}$, is real (where $k = 2\pi/\lambda$). Hence the integration domain is a circle in $\boldsymbol{q}$-space with radius $k$. Note that both $E(\boldsymbol{r})$ and $\tilde{E}(\boldsymbol{q})$ are band-limited functions. Applying the Shannon-Whittaker sampling theorem [14] to the expression for $\tilde{E}(\boldsymbol{q})$ in (6), it follows that $\tilde{E}(\boldsymbol{q})$ can be characterized by discrete samples,

$$\tilde{E}(\boldsymbol{q}) = \sum_{a_x,a_y=-\infty}^{\infty} \tilde{E}(a_x \frac{2\pi}{W}, a_y \frac{2\pi}{W}) \frac{\sin(q_x W/2 - a_x \pi)}{q_x W/2 - a_x \pi} \frac{\sin(q_y W/2 - a_y \pi)}{q_y W/2 - a_y \pi}.$$

Next, we use the fact that the electric field is band-limited in $q$-space as well. The integers $a_x, a_y$ have to satisfy $(a_x^2 + a_y^2)(2\pi/W)^2 \leq k^2$. The number of modes is therefore finite and is given by the number of pairs $(a_x, a_y)$ satisfying the momentum constraint $|\mathbf{q}| \leq k$. Denoting the transverse modes as $\boldsymbol{q}_a$, we have[6]

$$\boldsymbol{q}_a = \frac{2\pi}{W}(a_x, a_y) \ ; \quad N_{\text{mod}} = \# \left\{ (a_x, a_y) \text{ with } |\boldsymbol{q}_a| \leq k \right\} = \frac{\pi A}{\lambda^2}. \qquad (7)$$

The integers $a_x$, $a_y$ lie in the range $(-W/\lambda, W/\lambda)$. In the example of section 1.3 there are $N_{\text{mod}} = 1.3 \cdot 10^7$ transversal modes. The angular distance between outgoing modes corresponds to the correlation length present in the speckle pattern as derived by [15]. The scattering process can be represented as a complex random matrix $S$, whose elements map incoming states to outgoing states,

$$\tilde{E}_a^{\text{out}} = \sum_{b=1}^{N_{\text{mod}}} S_{ab} \tilde{E}_b^{\text{in}}. \qquad (8)$$

We take the distribution function of $S$ to be symmetric in all modes. We introduce $T_{ab} = |S_{ab}|^2$, the transmission coefficient from mode $b$ to mode $a$, which

[6] If polarisation is taken into account, the number of modes doubles. In this paper we will not consider polarisation.

specifies how much light *intensity* is scattered. Given a basic challenge, consisting of a single incoming mode $b$, a speckle pattern corresponds to an $N_{\mathrm{mod}}$-component vector $\boldsymbol{v}$, namely the $b$'th column of the $T$-matrix,

$$v_a = T_{ab},\ b \text{ fixed.} \tag{9}$$

Hence, the entropy of the response is given by $\mathsf{H}(\boldsymbol{v})$. Because of the mode symmetry in the distribution of $S$, the entropy does not depend on $b$. In the more general case where the challenge is a linear combination of basic challenges, one can always perform a unitary transformation on the modes such that the challenge is given by a single mode in the new basis. The response is then a single column of the transformed matrix $S'$. Since $S'$ has the same probability distribution as $S$, the entropy contained in one column of $S'$ is the same as the entropy of one column of $S$. Hence, $\mathsf{H}(\boldsymbol{v})$ (9) is valid for composite challenges as well.

**Weak PUFs: Upper Bound on $\mathsf{H}(\boldsymbol{R})$**

Here we derive an upper bound for the entropy of a speckle pattern. We start with a simplified situation, assuming the outgoing modes to be independent. This is generally not true but it gives an upper bound on $\mathsf{H}(R)$ and hence a lower bound on $C$. For this reason we refer to such a PUF as a *weak* PUF. It is clear that a speckle pattern cannot carry more information than the light leaving the PUF. We therefore derive an upper bound on the information content of $N_{\mathrm{mod}}$ light intensity states. Although the physics of multiple scattering is classical, we need the quantum description of light in terms of photons for our computation.[7] We have to count the number of *distinguishable* ways in which $N_\varphi$ photons can be distributed over $N_{\mathrm{mod}}$ outgoing modes. To this end we estimate the number of distinguishable photon states (energy levels) $N_{\mathrm{states}}$ in one mode. The energy in the mode is $Nh/\lambda$, [8] where $N$ is the number of photons in the mode. We restrict ourselves to the case of photon number statistics governed by $\langle N^2\rangle - \langle N\rangle^2 = \langle N\rangle$ without thermal noise. This Poisson relation holds for lasers and thermal light at room temperature. The more general case is treated in [14]. The energy states have a width of approximately $2\sqrt{N}$. Given the level density $1/(2\sqrt{N})$, the number of distinguishable energy levels with photon number lower than $N$ is

$$N_{\mathrm{states}} \approx \int_0^N \frac{\mathrm{d}x}{2\sqrt{x}} = \sqrt{N}. \tag{10}$$

The energy level of the $i$'th mode is denoted by the integer $L_i$ and the corresponding number of photons by $n_i \approx L_i^2$. We assume that all configurations $\{n_i\}$ have the same probability of occurring, as long as they satisfy the conservation $\sum_i n_i = N_\varphi$. From (10) we see that this translates to $\sum_i L_i^2 = N_\varphi$. Hence, the number of distinguishable configurations is given by the area of a section of an

[7] A similar situation arises in statistical mechanics, where a discretisation of classical phase space, based on quantum physics, is used to count the number of microstates.

[8] $h$ denotes Planck's constant.

$N_{\rm mod}$-dimensional sphere of radius $\sqrt{N_\varphi}$ (the section with positive $L_i$ for all $i$). The area of an $n$-sphere is $2\pi^{n/2}r^{n-1}/\Gamma(n/2)$. Our upper bound on $\mathsf{H}(R)$ is

$$\mathsf{H}_{\rm up}(R) \approx \log\left[(\tfrac{1}{2})^{N_{\rm mod}} 2\pi^{\frac{1}{2}N_{\rm mod}} \sqrt{N_\varphi}^{N_{\rm mod}-1}/\Gamma(\tfrac{1}{2}N_{\rm mod})\right]. \qquad (11)$$

Since $N_{\rm mod}$ is large, we can use Stirling's approximation and obtain

$$\mathsf{H}_{\rm up}(R) \approx \tfrac{1}{2}N_{\rm mod}\log\left(\tfrac{1}{2}\pi e N_\varphi/N_{\rm mod}\right). \qquad (12)$$

We have assumed $N_\varphi > N_{\rm mod}$, so the log in (12) is positive. The entropy increases with the number of photons, but only in a logarithmic way. Hence, errors in estimating $N_\varphi$ will have a small effect on $\mathsf{H}_{\rm up}(R)$. The number of participating photons is proportional to the measurement time $\triangle t$,

$$N_\varphi = P\triangle t\cdot\lambda/(hc), \qquad (13)$$

where $c$ is the speed of light. In principle, it is possible to completely characterize the PUF by performing a single very long measurement. However, as seen from (13) and (12), substituting $\mathsf{H}_{\rm up}(R)\rightarrow\mathsf{H}(K)$, $\triangle t$ is then exponential in $\mathsf{H}(K)$. Information can be extracted from the PUF much faster, namely linearly in $\triangle t$, by doing many fast measurements. Using the example numbers of section 1.3, we have $N_\varphi = 2.5\cdot 10^{12}$ and the upper bound is $\mathsf{H}_{\rm up}(R) < 1.2\cdot 10^8$.

**Strong PUFs: Lower Bound on $\mathsf{H}(R)$**

In multiple scattering PUFs, the modes at the outgoing surface are correlated. In [16] a correlation function was obtained for the elements of the $T$-matrix,

$$\frac{\langle\delta T_{ab}\delta T_{a'b'}\rangle}{\langle T_{ab}\rangle\langle T_{a'b'}\rangle} = D_1\delta_{\triangle\boldsymbol{q}_a,\triangle\boldsymbol{q}_b}F_1(\frac{d}{2}|\triangle\boldsymbol{q}_b|) \qquad (14)$$

$$+\frac{D_2}{4gN_{\rm mod}}\left[F_2(\frac{d}{2}|\triangle\boldsymbol{q}_a|)+F_2(\frac{d}{2}|\triangle\boldsymbol{q}_b|)\right]+\frac{D_3}{(4gN_{\rm mod})^2}$$

where $\delta T_{ab} = T_{ab}-\langle T_{ab}\rangle$, $\langle\cdot\rangle$ is the average over all scatterer configurations and

$$F_1(x) = x^2/\sinh^2 x\ ; \quad F_2(x) = 2/(x\tanh x) - 2/\sinh^2 x \qquad (15)$$

with $\triangle\boldsymbol{q}_a = \boldsymbol{q}_{a'}-\boldsymbol{q}_a$, $g$ the transmittance $N_{\rm mod}^{-1}\sum_{ab}T_{ab}\approx\ell/d$, and $D_i$ constants of order unity. Due to the correlations, the number of degrees of freedom $N_{\rm dof}^{\rm out}$ in the speckle pattern is less than $N_{\rm mod}$. We calculate $N_{\rm dof}^{\rm out}$ following the approach of [8], but we make use of (14). We sum over the correlationsin the vector $\boldsymbol{v}$ to obtain the *effective cluster size* $\mu$. $\mu$ represents the number of variables correlated to a given $v_a$ (for arbitrary $a$). The vector $\boldsymbol{v}$ can be imagined to consist of uncorrelated clusters of size $\mu$, where each cluster contains exactly one degree of freedom. This means that we approximate $\boldsymbol{v}$ by a vector of $N_{\rm dof}^{\rm out} = N_{\rm mod}/\mu$ independent cluster-size entries. Denoting the variance of $v_a$ as $\sigma_a$ and neglecting the $D_3$ term, the correlations within $\boldsymbol{v}$, obtained from (14), are given by

$$C_{aa'} = \langle\delta v_a\ \delta v_{a'}\rangle/(\sigma_a\sigma_{a'}) = \delta_{aa'}+D_2/(D_1 4gN_{\rm mod})\left[\tfrac{4}{3}+F_2(\tfrac{1}{2}d|q_a-q_{a'}|)\right]. \quad (16)$$

The $D_2$ term consists of the sum of a short-range ($F_2$) term and a long-range contribution (4/3). From (16) we obtain $\mu$ and the number of degrees of freedom,

$$\mu = \sum_a C_{aa'} \approx \frac{D_2}{3D_1}\frac{d}{\ell}\,; \quad N_{\rm dof}^{\rm out} = \frac{N_{\rm mod}}{\mu} \approx \frac{3D_1}{D_2}\frac{\pi A}{\lambda^2}\frac{\ell}{d}. \tag{17}$$

Here we have neglected the summation over the $F_2$-term, since $F_2(x)$ asymptotically falls off as $2/x$ for large $x$. We have also neglected the contribution $\sum_a \delta_{aa'} = 1$ with respect to $d/\ell$.

The speckle entropy is calculated by repeating the level counting computation of the 'Weak PUFs' section, but now with modified parameters. Every output mode within a cluster of size $\mu$ emits exactly the same amount of light. Consequently, the problem of distributing $N_\varphi$ photons over $N_{\rm mod}$ modes is equivalent to the problem of distributing $N_\varphi/\mu$ bunches of $\mu$ photons over $N_{\rm mod}/\mu$ clusters. Performing the substitution $\{N_{\rm mod} \to N_{\rm mod}/\mu, N_\varphi \to N_\varphi/\mu\}$ into (12) we obtain

$$\mathsf{H}_{\rm low}(R) = \frac{N_{\rm dof}^{\rm out}}{2}\log\left(\frac{\pi e}{2}\frac{N_\varphi}{N_{\rm mod}}\right) = \frac{3\pi D_1}{2D_2}\frac{A\ell}{\lambda^2 d}\log\left(\frac{\pi e}{2}\frac{N_\varphi}{N_{\rm mod}}\right). \tag{18}$$

Substituting into (18) relation (13) and the numbers given in section 1.3, we have $\mathsf{H}_{\rm low}(R) \approx 4 \cdot 10^6$. By assuming that several modes carry the same photon state, we have underestimated $N_{\rm dof}^{\rm out}$. Therefore, the result (18) is indeed a lower bound on $\mathsf{H}(R)$. Furthermore, we have assumed that all the information present in the outcoming light is recorded by an ideal detector, capturing all the light in a sphere surrounding the PUF. This is the optimal situation for an attacker. Hence we err on the side of safety.

### 3.3 The Security Parameter

We now use the results of section 3.2 to estimate the security parameter. We assume that we are in the regime where the PUF can be probed to such an extent that all bits can be determined by measurements. In this regime we have $C = \mathsf{H}(K)/\mathsf{H}(R)$, and after substitution of the upper bound (12) and the lower bound (18) for $\mathsf{H}(R)$ we find that $C$ lies in the interval

$$(\min\left\{\frac{2}{\pi}\cdot\frac{1}{\log(\frac{\pi e}{2}\frac{N_\varphi}{N_{\rm mod}})}\cdot\frac{d}{\lambda},\ N_{\rm mod}\right\}, \min\left\{\frac{2}{3\pi}\cdot\frac{1}{\log(\frac{\pi e}{2}\frac{N_\varphi}{N_{\rm mod}})}\cdot\frac{d^2}{\lambda\ell},\ N_{\rm mod}\right\}) \tag{19}$$

The $\min\{\cdots, N_{\rm mod}\}$ function reflects the fact that there are no more than $N_{\rm mod}$ basic challenges. The result (19) has the following properties:

- $C$ grows with increasing $d/\lambda$, since the PUF entropy is proportional to $d/\lambda$.
- In addition, the upper bound on $C$ grows with increasing $d/\ell$. This is a measure for the number of scattering events $N_{\rm sc}$ taking place before a photon exits the PUF. (Assuming a random walk, $d/\ell \propto \sqrt{N_{\rm sc}}$). Hence, multiple scattering increases the cryptographic strength of a PUF.

- (19) refers to one illuminated area $A = W^2$. By shifting the laser over a distance equal to the diameter of the laser spot, one illuminates a new subvolume of the PUF with the same number of challenges. This means that the total number of independent challenges $C_{\text{tot}}$ is given by

$$C_{\text{tot}} = C \cdot A_{\text{PUF}}/A. \tag{20}$$

Using the numbers from section 1.3, (19) gives $3 \cdot 10^4 \leq C_{\text{tot}} \leq 1 \cdot 10^6$. In order to achieve this many distinct challenges in practice, an angular accuracy of laser positioning is required of the order of 1 mrad. This is easily achievable.

We emphasize once more that the security parameter has been computed from an *information-theoretic* point of view. This means that an attacker who has gathered $C_{\text{tot}}$ CRPs in principle knows the complete CRP behaviour. He is *in principle* able to compute the response to a new challenge (one for which he has not seen the response before). *In practice* the security might be much stronger and is based on the following assumptions: (i) the so-called *forward problem* (computing the response for a given challenge) is difficult and (ii) interpolation techniques do not allow for an accurate prediction of a response, given responses to nearby challenges. This means that one can use more than $C_{\text{tot}}$ CRPs safely.

Finally, we compare our result (19) to [3, 4]. Their approach is based on the memory angle $\delta\theta \propto \lambda/d$ [16] and does not take the density of scatterers into account. Dividing the half-sphere into pieces of solid angle $\delta\theta^2$, they obtain a number of CRPs proportional to $d^2/\lambda^2$, representing the number of obtainable responses that look mutually uncorrelated. This number is larger than our upper bound for $C$ by a factor $\propto \ell/\lambda$. The two approaches give comparable results only in the limit of extremely strong scattering, $\ell \approx \lambda$.

## 4 Attacks and Countermeasures

We discuss the following threat. An attacker steals somebody's PUF and tries to characterize the PUF without being noticed by the owner. In particular this means that the PUF has to be returned to the owner within a short time period.

### 4.1 Brute Force

It follows from the definition of $C$ that *in principle* only $C$ measurements are required to fully characterize a PUF. However, an attacker faces the problem that CRP intrapolation is difficult. Consequently, a brute force attack may be more feasible. The brute force attack is an attempt to exhaustively record the full set of CRPs. The responses are stored in a database. Let us assume that a challenge takes the form of a single incoming transverse momentum mode; it is clear that the number of possible challenges is of order $N_{\text{mod}}$. The required storage space is relatively small, since it is not necessary to store complete speckle patterns, but only the keys/identifiers derived from them. The measurement duration for this attack is $N_{\text{mod}} \triangle t \cdot A_{\text{PUF}}/A = \pi A_{\text{PUF}} \triangle t/\lambda^2$. Using the wavelength and the

PUF area from the example in section 1.3, and taking $\triangle t$ of the order of 10ms, we have a total duration in the order of hundreds of days. This is too long for the attack to go unnoticed in the scenario sketched above.

We emphasize the necessity of enforcing "long" measurement times $\triangle t$.

### 4.2 The Slow PUF

A long integration time in the detector can be achieved by attaching a gray filter (irremovably) to the PUF. Let us denote the transmission of the combined PUF and gray filter by $\eta_{\rm PUF}$. In the detector the incoming photons are converted to electrons with quantum efficiency $\eta_{\rm Q}$. The actual signal of each detector cell is the number of electrons $N_{\rm e}$ collected in time $\triangle t$. The number of cells in the detector is denoted as $N_{\rm cells}$. The generation of photo-electrons is a Poisson process,

$$\left\langle N_{\rm e}^2\right\rangle - \left\langle N_{\rm e}\right\rangle^2 = \left\langle N_{\rm e}\right\rangle = \frac{\eta_Q N_\varphi}{N_{\rm cells}} = \frac{\eta_{\rm Q}\eta_{\rm PUF}P}{N_{\rm cells}(hc/\lambda)}\triangle t. \tag{21}$$

The signal to noise ratio SNR can at most be equal to $\left\langle N_{\rm e}\right\rangle^2/(\left\langle N_{\rm e}^2\right\rangle - \left\langle N_{\rm e}\right\rangle^2) = \left\langle N_{\rm e}\right\rangle$, since there may be other noise sources which are not taken into account here. Hence, (21) gives a lower bound on $\triangle t$, proportional to the SNR. According to Shannon's theorem [12], the number $b$ of useful bits that can be extracted from any signal is limited in the following way,

$$b \leq \tfrac{1}{2}\log(1+\mathrm{SNR}). \tag{22}$$

In our case, $b$ represents the number of bits used for gray level representation. Combining (21) and (22) we obtain

$$\triangle t \geq \frac{(hc/\lambda)N_{\rm cells}}{\eta_{\rm Q}\eta_{\rm PUF}P}(2^{2b}-1). \tag{23}$$

For example, taking $\eta_{\rm Q} = 0.3$, $\eta_{\rm PUF} = 0.001$, $N_{\rm cells} = 3\cdot 10^6$ and $b = 4$ in the example of section 1.3, we get $\triangle t \geq 1$ms.

This gives a fundamental physical lower bound on the integration time, which can therefore never be reduced by technological advances. Given a challenge-response setup with fixed $P$ and $\eta_{\rm Q}$, (23) shows that the integration time can be increased in the following ways: (i) by decreasing the transmission $\eta_{\rm PUF}$, (ii) by increasing the number of gray levels $2^b$ that should be detected in order to correctly process the speckle pattern and (iii) by increasing the number of pixels that should be resolved by the detector to ensure correct processing. All three methods have their limitations.

An attacker can of course use the best detector in existence (high $\eta_{\rm Q}$). He can also use any laser that he desires (high $P$). Especially the use of a high-intensity laser can dramatically shorten the integration time. This can be prevented by adding to the PUF a photo-layer which darkens (permanently or temporarily) when irradiated by light above a certain threshold intensity.

## 5 Conclusions

We have introduced a general theoretical framework to study the secret key capacity of PUFs. We propose to use the number $C$ of independent CRPs as a security parameter. This parameter represents a security measure that is not affected by technological or computational advances. In practice one may use many more CRPs safely, under the assumption that correlations between CRPs are hard to exploit computationally.

For optical PUFs we have derived an analytical expression for $C$. In order to make brute force attacks difficult, long measurement times have to be enforced. This has been analyzed for the case of optical PUFs.

## References

1. B. Gassend et al., *Controlled Physical Random Functions*, Proc. 18th Annual Computer Security Applications Conf., Dec. 2002.
2. B. Gassend et al., *Silicon Physical Unknown Functions*, Proc. 9th ACM Conf. on Computer and Communications Security, Nov. 2002.
3. R. Pappu, *Physical One-Way Functions*, Ph.D. thesis, MIT 2001.
4. R. Pappu et al., *Physical One-Way Functions*, Science Vol. 297, Sept 2002, p.2026.
5. B.L.P. Gassend, *Physical Random Functions*, Master's Thesis, MIT 2003.
6. P. Tuyls, B. Škorić, *Secret Key Generation from Classical Physics*, Proceedings of the Hardware Technology Drivers for Ambient Intelligence Symposium, Philips Research Book Series, Kluwer, 2005.
7. M. Magnor, P. Dorn and W. Rudolph, *Simulation of confocal microscopy through scattering media with and without time gating*, J.Opt.Soc.Am. B, Vol. 19, no. 11 (2001), pp 1695–1700.
8. J. F. de Boer, *Optical Fluctuations on the Transmission and Reflection of Mesoscopic Systems*, Ph D thesis, 1995, Amsterdam.
9. H. Furstenberg, *Noncommuting Random Matrices*, Trans. Am. Math. Soc. 108, 377, 1963.
10. J.P. Linnartz, P. Tuyls, *New Shielding Functions to enhance Privacy and Prevent Misuse of Biometric Templates*, Proc. 4th International Conference on Audio and Video based Biometric Person Authentication, LNCS2688, Guildford UK, June 9-11, 2003.
11. E. Verbitskiy, P. Tuyls, D. Denteneer, J.P. Linnartz, *Reliable Biometric Authentication with Privacy Protection*, Proc. of the 24th Symposium on Information Theory in the Benelux.
12. T.M. Cover, J.A. Thomas, *Elements of Information Theory*, Wiley ans Sons, New York, 1991.
13. K. Petersen, *Ergodic Theory*, Cambridge University Press, 2000.
14. D. Gabor, *Light and Information*, in E. Wolf, Ed., Progress in Optics Vol. I, North-Holland, Amsterdam 1961.
15. J. W. Goodman, *Statistical properties of laser speckle patterns*, in Laser Speckle and Related Phenomena, 2nd ed., J. C. Dainty, Ed. New York: Springer-Verlag, 1984.
16. S. Feng, C. Kane, P.A. Lee and A.D. Stone, *Correlations and Fluctuations of Coherent Wave Transmission through Disordered Media*, Phys.Rev.Lett. Vol.61 No.7 (1988), pp 834–837.

# Risk Assurance for Hedge Funds Using Zero Knowledge Proofs

Michael Szydlo

mike@szydlo.com

**Abstract.** This work introduces a new tool for a fund manager to verifiably communicate portfolio risk characteristics to an investor. We address the classic dilemma: *How can an investor and fund manager build trust when the two party's interests are not aligned?* In addition to high returns, a savvy investor would like a fund's composition to reflect his own risk preferences. Hedge funds, on the other hand, seek high returns (and commissions) by exploiting arbitrage opportunities and keeping them secret. The nature and amount of risk present in these highly secretive portfolios and hedging strategies are certainly not transparent to the investor.

This work describes how to apply standard tools of cryptographic *commitments* and *zero-knowledge proofs*, to financial engineering. The idea is to have the fund manager describe the portfolio contents indirectly by specifying the asset quantities with cryptographic commitments. Without de-committing the portfolio composition, the manager can use zero knowledge proofs to reveal chosen features to investors - such as the portfolio's approximate sector allocation, risk factor sensitivities, or its future value under a hypothetical scenario.

The investor can verify that the revealed portfolio features are consistent with the committed portfolio, thus obtaining strong assurance of their correctness - any dishonest portfolio commitment would later serve as clear-cut evidence of fraud. The result is a closer alignment of the manager's and investor's interests: the investor can monitor the fund's risk characteristics, and the fund manager can proceed without leaking the exact security composition to competitors.

**Keywords:** Hedge fund, zero-knowledge, commitment scheme, investor trust.

## 1 Introduction

This paper describes a novel application of zero-knowledge techniques to the relationship between and investor and a portfolio manager. The interest of the fund manager is in earning high returns, so he may want to keep his exact portfolio and trading strategy secret. An investor, on the other hand, also requires mechanisms to ensure the honesty of the managers, and to check that the fund's risk characteristics are in line with his own risk preferences. We address the fundamental problem of how to control the flow of risk information to serve these

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 156–171, 2005.
© Springer-Verlag Berlin Heidelberg 2005

distinct interests. We suggest that the tool described in this paper is particularly suited to *hedge funds*, which tend to be highly secretive, more loosely regulated, and potentially very lucrative.

Cryptography has been applied to financial transactions before, in both generic ways (privacy, integrity), as well as in ways specific to transactions (digital cash, and privacy-preserving auctions). Rather than focus on the transactions themselves, our approach uses cryptography to allow a more finely controlled release of financial information to an investor.

Our idea is to use cryptographic commitments and zero knowledge proofs in a remarkably simple way: The fund manager describes the portfolio contents indirectly by specifying the asset quantities with cryptographic commitments. Then, without de-committing the portfolio composition, the manager can use zero knowledge proofs to reveal chosen features to the investor. This technique differs from traditional topics in financial cryptography, since it applies the tools of cryptography directly to mainstream *financial engineering.*

The main cryptographic tools we require are standard: *Pedersen Commitments* and *Interval Proofs*. We review the mechanics of these tools and show how to assemble them into (zero knowledge) statements which are meaningful to the investor. We stick to such well-known building blocks in this paper in order to retain the focus on the new finance application.

We have implemented a prototype of the protocol to demonstrate its feasibility. Despite the potential for efficiency improvements, the basic construction is already good enough to serve in practice. This shows that it is possible for a fund to communicate interesting risk information for large and complicated portfolios on a daily basis.

The rest of this paper is organized as follows: In Section 2 we provide some background on hedge funds, and the risks associated to them. In Section 3 we review the cryptographic building blocks we require, and in Section 4 we describe the mechanics of the protocol. We continue with some detailed applications in Section 5. In Section 6 we describe the results of our prototype implementation, and discuss efficiency concerns. We conclude in Section 7, and provide an appendix with some further technical details on the cryptographic construction.

## 2 Finance Background

For the non-finance professional, and to motivate our work, we first review some basic finance material, highlighting the roles of information and risk. We focus on the differing interests of the investor and fund manager with respect to release of information to motivate the need for our risk communication protocol. Including the background on common methods employed in the industry to measure risk also helps show that most of the meaningful risk statements used in practice are compatible with our protocol. Much of this material is present in introductory finance textbooks, e.g., see [5], which emphasize quantative methods.

## 2.1 Hedge Funds and Risk

**Portfolios and Risk:** An investment portfolio is just a collection of assets designed to store or increase wealth. In a *managed fund*, the *investor* turns over capital to a *fund manager*, an investment professional who buys, sells, and otherwise maintains the portfolio in return for a fee or commission. The assets often contain publicly traded *securities* such as stocks, bonds, commodities, options, currency exchange agreements, mortgages, "derivative" instruments, as well as less liquid assets such as real estate, or collectibles. Examples of managed funds are pension funds, 401K plans, mutual funds, and hedge funds.

Every type of investment contains uncertainty and risk. Ultimately, the risk inherent in investments derives from the fact that the future market value[1] depends on information which is not available: information concerning either unknown future events, or information concerning past events which has not been publicly disclosed or effectively analyzed. The charter of the fund manager is to manage these risks in accordance with the preferences of the investor.

**Risk Factors:** The finance profession has developed a plethora of models to define and estimate portfolio risks. A first description of a portfolio's risks includes a breakdown of the types of assets in the fund such as the proportion of capital invested in equity, debt, foreign currency, derivatives, and real estate. A further breakdown specifies the allocation by industry type or *sector*, or region for foreign investments.

The future value of an investment depends on such future unknown factors as corporate earnings for stocks, interest rates and default likelihood for bonds, monetary policy and the balance of trade for foreign currency, regional political stability for any foreign investment, re-financing rates for securitized mortgages, housing demand for real estate, etc.

Risk models identify such measurable *risk factors*, and study the dependence of the asset's value on each such factor. Such *factor exposures*, are estimated with statistical regression techniques, and describe not only the sensitivity to the factor but also how the variance, or *volatility* of a security depends on such correlated factors. Assembling such analysis for all securities in a portfolio, the fund manager has a method for quantatively understanding the relative importance of the risk factors his portfolio is exposed to. Another important tool, *scenario analysis* estimates the future value of a portfolio under a broad range of hypothetical situations.

**Hedge Funds:** To *hedge* against a risk is to effectively buy some insurance against an adversarial event. When two assets depend oppositely on the same risk factor, the combined value of the pair is less sensitive to that factor. A *Hedge Fund* is just a type of portfolio designed to have certain aggregate risk characteristics. Hedge funds may use leveraging techniques such as *statistical*

[1] Economists like to point out that there is no robust intrinsic definition of value outside a market.

*arbitrage*, engaging in long and short positions in similarly behaving securities, hoping to earn a profit regardless of how the correlated securities behave.

Hedge funds are often large private investments and are more loosely regulated than publicly offered funds. (Only in 2006 must hedge funds register with the SEC at all). Such extra flexibility affords the possibility of exceeding the performance of more standard funds. For example, hedge funds often take a position contrary to the market consensus, effectively betting that a certain event will happen. When accompanied by superior information or analysis such bets can indeed have high expected value. Of course, highly leveraged funds can be extremely sensitive to a particular risk factor, and are thus also susceptible to extreme losses.

The high investment minimums, lax regulation and secrecy or “black box” nature of hedge funds has fostered an aura of fame and notoriety through their spectacular returns, spectacular losses, and opportunities for abuse. Recently, though, there has been interest in marketing hedge funds as viable opportunities for the average investor.

### 2.2 The Role of Information

**Information and Asset Prices:** A *market* assigns a value to an asset based on the prices in a steady steam of transactions. It is the pieces of information which are perceived to be relevant to the asset’s value which are compared to existing expectations and drive the supply, demand, and market price. The pivotal role of information is embodied in the *efficient market hypothesis* which states that under the assumption of perfect information distribution, the collective brainpower of investors will reduce arbitrage opportunities, and force the market price to an equilibrium.

In the real world, information distribution is not perfect, and the *information asymmetries* among parties significantly affect the behavior of asset prices in the market. The situation is worse for illiquid assets, for which one must rely on some ad-hoc *fundamental analysis* to estimate the value. Similarly, it is difficult to assign a robust value to an investment fund with opaque risk characteristics (such as a hedge fund). An increasing sharing of the actual risk profile of hedge funds would increase their usefulness in *funds of funds*, for example.

**The Importance of Secrets:** Certain investments, such as passive funds which track an index may have no requirement to protect the portfolio contents or trading patterns. Actively traded funds, on the other hand, have good reasons to maintain secrets. For example, revealing in advance an intention to purchase a large quantity of some security would drive the price up. A parallel can be made with corporations: Sharing technological, financial, and trade secrets would undermine the competitive advantage of a firm.

Especially relevant to our focus, if a hedge fund were exploiting a subtle but profitable arbitrage opportunity, revealing this strategy would quickly destroy the benefit, as other funds would copy the strategy until it was no longer profitable. Thus, a rational investor will support such constructive use of secrets.

**The Importance of Transparency:** Secrecy is also dangerous. The actions of a fund manager might not always represent the goal of creating value for the investor! The danger of too much secrecy is that it also reduces barriers to theft, fraud, and other conflicts of interest. An example of corrupt behavior that might be discouraged by increased transparency is the practice of engaging in unnecessary trading motivated by brokerage commissions. To combat this risk, individual investors require enough access to information about a company or fund to help ensure honest management, consistent with the creation of value.

Another kind of problem will arise if the investor is not aware of the kinds of risks his portfolio is exposed to. In this case it is impossible to tell if these risks are in line with his preferences. A fund manager might be motivated by a fee structure which encourages him to take risks that are not acceptable to the investor. When the fee structure or actual level of risk in the portfolio is not evident to the investor, a fund manager may legally pursue actions consistent with interests other than the investor's.

**Aligning Interests:** The above discussion about the differing views concerning just how much risk information should be kept secret and how much should be revealed shows how difficult it is in practice to perfectly align the interests of investors and fund managers. The traditional approaches to mitigating this problem involve financial regulatory bodies such as the SEC, which seeks to institute reporting laws and support capital requirements that protect the investor, ideally without imposing too large a burden on the financial institution. In the case of hedge funds, the position of the SEC is that the interests of the investor are not adequately protected [1]. Indeed, it has not been able to eliminate all fraud and conflict of interests arising in the context of hedge funds.

There are several requirements for a good set of mechanisms to align the interests of investors and managers. These include methods for the investor to ensure the honesty of the fund manager, methods for the investor to be aware of the fund's evolving risks, and contractual agreements and fee structures which discourage the manager from adding hidden risks. Finally, the mechanisms should not discourage the fund manager from fully exploiting any competitive advantage or superior analysis which he might have.

### 2.3 Finance and Cryptography

**Previous Work:** There are many existing applications of cryptography to financial infrastructure. The most significant practical applications involve well known aspects of securing the transactions themselves: providing authenticity of the parties, integrity and non-repudiation of the transactions, and confidentiality among the parties. Such applications all use cryptography in a generic way, not tailored to any particular requirements of finance.

More interesting advanced finance-related applications of cryptography include fair exchange, secure auctions, and digital anonymous cash. These applications use cryptography as a building block to compose cryptographic protocols which protect some aspect of a transaction, preserving some secret, or prove the correctness of a protocol step. The technique of sending non-interactive proofs relative to previously committed values is pervasive in protocol design.

The present application to finance is not directly focused on the transactions, but instead on the release of information about the evolving portfolio's composition and risks. This kind of application has not previously appeared.

**New Contributions:** Our contribution is the proposal of an additional mechanism which will help achieve a better balance of information sharing between fund managers and investors. We present a protocol which can precisely control the level of transparency in an investment fund. The result is that the investor can ensure that an appropriate level and type of risk is taken, yet the fund can pursue competitive strategies which would not be possible if the restriction of perfect transparency were imposed.

Cryptographic commitments, and zero knowledge proofs provide versatile tools for precisely controlling the delivery of partial and verifiable pieces of information. Our work is the first to exploit these methods in the context of financial risk management. When our protocol is used to communicate the amounts and types of risk in a portfolio, the interests of each party will be better served. In addition to outlining the basic approach, the technical applications we describe below serve as specific examples of how various types of risks can be communicated within our framework.

## 3 Cryptographic Building Blocks

The cryptographic tools we require in our construction are all standard. Namely we require commitments with a homomorphic property, and zero knowledge proofs that a committed integer lies in a interval. In this section, we review the most well-known versions of these constructions. Throughout this paper, we let $p$ denote a large prime and $q$ a prime such that $q|p-1$. Let $\mathbf{G} = \mathbf{Z}$ denote the group of mod-$p$ integers, and let $g \in \mathbf{G}$ and $h \in \mathbf{G}$ be group elements of order $q$ such that the discrete log, $log_g(h)$ is unknown. We also let $\mathsf{hash}$ denote a cryptographic hash function with range $[0, q-1]$.

**Pedersen Commitment:** A cryptographic commitment is a piece of data which binds its creator to a unique value, yet appears random until it is de-committed. A *Pedersen commitment* [8] to $x$ with randomness $r$ is the group element $C_r(x) = g^x h^r$, and can be de-committed by revealing the $r$ and $x$. This commitment is computatationally binding and unconditionally hiding. Since a commitment can only feasibly de-commit to the original value of $x$, we also say $C_r(x)$ "corresponds" to $x$.

**Linearity Property:** We make essential use of the linear (homomorphic) properties which Pedersen commitments enjoy:

$$C_r(x)^a = C_{ar}(ax) \tag{1}$$

$$C_r(x)C_{r'}(x') = C_{r+r'}(x+x') \tag{2}$$

Thus, without knowing the values $x$ and $x'$ that two commitments hide, any party can compute a commitment to any fixed linear combination of $x$ and $x'$.

**Proof of Knowledge:** A *zero knowledge proof of knowledge* allows a prover to demonstrate knowledge of hidden values without actually revealing them. A proof of knowledge of a (Pedersen) committed integer $x$ [10] demonstrates knowledge of some $x$ and $r$ such that $C_r(x) = g^x h^r$. We focus on *non-interactive* proofs of knowledge, for which the proof is concentrated in a single piece of data and can be later verified without any further participation of the prover.

One can also prove that a committed value $x$ satisfies some condition $\phi(x)$ without revealing it, and we use the notation $POK(x, r \mid C = g^x h^r, \phi(x))$ to denote a zero knowledge proof of knowledge of $(x, r)$ satisfying both $C = g^x h^r$ and the predicate $\phi(x)$.

**Schnorr OR Proofs:** The well known *Schnorr OR proof* [6, 10].

$$POK(x, r \mid C = g^x h^r, x \in \{0, 1\}) \tag{3}$$

can be used to prove that $x \in \{0, 1\}$, (provided this is true), without leaking whether $x$ is 0 or 1. The proof data consists of the five values $\{C, r_1, r_2, c_1, c_2\}$ such that $c_1 + c_2 = \mathsf{hash}\ (a_1, a_2) \pmod{q}$, where $a_1 = h^{r_1} C^{-c_1}$, and $a_2 = h^{r_2} (C/g)^{-c_2}$. Any verifier can efficiently check these conditions. In Appendix A, we review the *completeness*, *zero-knowledge*, and *soundness* properties of this construction.

**Interval Proofs:** We will need proofs that a committed integer satisfies an inequality such as $x \geq A$. One way to accomplish this is to prove that $x$ lies in an interval $[A, B]$ for a large enough $B$. We now review the classic interval proof [4, 7, 6], based on bounding the bit length of an integer.

$$POK(x, r \mid C = g^x h^r, x \in [0, 2^k - 1]). \tag{4}$$

The proof is constructed as follows: First expand $x$ in binary: $x = \sum_0^k 2^i a_i$, and produce a commitment $C_i = C_{r_i}(a_i)$ for each digit. The commitment to the last digit is set to be $C / \Pi_1^k (C_i^{2^i})$, so that the relation $C = \Pi_0^k (C_i^{2^i})$ holds[2]. Finally, for each digit $a_i$ compute a Schnorr OR proof demonstrating that $a_i \in \{0, 1\}$. This proof is verified by checking the list of $k$ Schnorr proofs, and checking that $C = \Pi_0^k (C_i^{2^i})$ holds.

To construct a proof that $x$ is in the range $[A, 2^k - 1 + A]$, one simply follows the same procedure, replacing $C$ with $C/g^A$. These proofs are reasonably efficient in practice, as long as the interval is not too large. See [3] for alternate constructions of interval proofs designed for time and space efficiency.

[2] An alternative to adjusting the last digit's commitment is to add a proof that $C$ and $\sum_0^k 2^k C_i$ commit to the same number.

### 3.1 Further Notation

For our application we will need to make commitments to a large set of quantities (assets) and prove statements about linear combinations of them. We consider a universe of asset types $\{\mathsf{A_i}\}$, and let $\mathsf{b_i}$ denote an amount of asset type $\mathsf{A_i}$ , and $\mathsf{C_i}$ a commitment to this value.

By virtue of the homomorphic property of Pedersen commitments, for any list of coefficients $\{\mathsf{m_i}\}$, the product $\Pi\,\mathsf{C_i}^{\mathsf{m_i}}$ is a commitment to $\Sigma\mathsf{m_i}\mathsf{b_i}$, and can thus be publicly computed from the $\{\mathsf{C_i}\}$ and $\{\mathsf{m_i}\}$. By using the interval proof technique reviewed above, the creator of the commitments can prove that $\Sigma\mathsf{m_i}\mathsf{b_i} \in [\mathsf{Q}, \mathsf{Q} + 2^k - 1]$, for any threshold integer $\mathsf{Q}$. Since all of the zero-knowledge proofs we use are with respect to the same $\mathsf{C_i}$, hiding $\mathsf{b_i}$ we abbreviate

$$POK(x, r \mid \Sigma\mathsf{m_i}\mathsf{C_i} = g^x h^r, x \in [\mathsf{Q}, \mathsf{Q} + 2^k - 1]) \qquad (5)$$

to the more succinct expression which also de-emphasizes the interval length

$$ZKP_k(\Sigma\mathsf{m_i}\mathsf{b_i} \geq \mathsf{Q}). \qquad (6)$$

Similarly, a zero knowledge proof that an expression is bounded above is denoted $ZKP_k(\Sigma\mathsf{m_i}\mathsf{b_i} \leq \mathsf{Q})$. To summarize, this proof data (6) allows any verifier with the $\{\mathsf{C_i}\}$, $\{\mathsf{m_i}\}$ and $\mathsf{Q}$ to check that $\Sigma\mathsf{m_i}\mathsf{b_i} \geq \mathsf{Q}$ for the $\mathsf{b_i}$ hidden in the $\mathsf{C_i}$.

## 4 The Risk-Characteristic Protocol

### 4.1 The Basic Approach

The process we describe provides the investor with a new tool to verify claims made by the fund manager, and there are both contractual and cryptographic aspects of the mechanism. Additionally, the involvement of a third party enhances the effectiveness of the scheme.

As part of the financial design phase, a universe of possible asset types is chosen, and the kinds of risk information to be verifiably communicated are identified. Such parameters are incorporated into the contract governing the fund. The more interactive component of the scheme involves a periodic delivery of risk assertions and accompanying proofs to the investor.

**Contractual Aspects:** The legal document governing the investment, the *prospectus* specifies the rights and obligations of the investor and the fund, including the mechanics of the contributions, payments, withdrawals, and fees. The prospectus may also specify or limit the types of investments made within the fund.

With our scheme, the architect of the fund chooses the risk profile and management strategy that he will follow, and incorporates the investment restrictions he is willing to guarantee into the prospectus. As part of a legal agreement, the fund would already be legally obligated to respect these conditions. However, such guarantees become much more meaningful when there is a mechanism for the investor to verify them in real time. The following steps facilitate this.

Within the prospectus a list of *allowable assets* is specified. The assets $\mathsf{A_i}$ can be directly identified by symbol if the security is market traded, and if not, described via their characteristics. Illiquid or private assets such as real estate, commercial mortgages, private bonds, or reinsurance contracts, can still be identified by descriptive categories. The units must be specified for each security, or asset type, since the rest of the protocol requires that the quantities be represented as integers. The *risk conditions* must also be expressed in the contract, and need to be expressed in a specific form to be compatible with the framework of our protocol. The conditions on the quantities $\mathsf{b_i}$ of assets $\mathsf{A_i}$ must take the form

$$\Sigma \mathsf{m_i b_i} \leq \mathsf{Q} \;\; \texttt{or} \;\; \Sigma \mathsf{m_i b_i} \geq \mathsf{Q} \tag{7}$$

where the set of coefficients $\{\mathsf{m_i}\}$ and bound $\mathsf{Q}$ determine the nature of the condition. We denote the list of conditions incorporated into the contract by $\mathsf{Limit_j}$. It is easy to see how such conditions might be used to limit the amount invested in a single security, asset type, or sector.

In Section 5, we discuss how such conditions can also be used to bound total exposure to a specific risk factor, or expected value under a hypothetical scenario. Thus, the linear form of the conditions is not too restrictive. The applications using factor exposures or scenario analysis should also place additional data in the contract. The data which must be placed in the prospectus is thus:

1. The list of asset types $\mathsf{A_i}$.
2. The list of conditions $\mathsf{Limit_j}$.
3. (Optional) The list of risk factors $\mathsf{F_j}$.
4. (Optional) The list of factor exposures $\mathsf{e_{i,j}}$.
5. (Optional) The list of scenarios $\mathsf{S_j}$.
6. (Optional) The list of scenario valuations $\mathsf{v_{i,j}}$.

### 4.2 The Protocol Steps

Once the prospectus has been fully designed, the fund manager may solicit funds from investors and invest the capital in a manner consistent with the contractual restrictions. As often as specified in the contract, (e.g. daily), the fund manager will commit to the portfolio, and produce statements and proofs for each of the contractual risk-limitations. The commitments may also be sent to a third party to facilitate resolution of disputes. The protocol takes the following form:

1. The fund manager commits to $\mathsf{b_i}$ with $\mathsf{C_i}$.
2. The fund manager delivers commitments $\{\mathsf{C_i}\}$ to the investor, and optionally to a third party.
3. (Optional) The fund manager also sends a de-commitment of the committed quantities $\{\mathsf{b_i}\}$ to the third party.
4. The fund manager asserts that conditions $\mathsf{Limit_j}$ are fulfilled, computes proofs $ZKP_k(\Sigma \mathsf{m_i b_i} \leq \mathsf{Q})$, or $ZKP_k(\Sigma \mathsf{m_i b_i} \geq \mathsf{Q})$, and sends them to the investor.
5. The investor verifies the completeness of the correctness of the proofs.

6. In case of dispute, the commitments may be opened or revealed by the third party. If the actual portfolio holdings do not match the committed holdings, the commitments serve as direct evidence of fraud.

We now elaborate on several aspects of this protocol.

**Trading Behavior:** In order to respect the contractual risk conditions, the fund manager must be sure to check that the risk profile would remain sound before effecting any transaction.

**Commitment Step:** Using the commitment scheme reviewed above, the number of units, $\mathsf{b_i}$, of each $\mathsf{A_i}$ is committed to. The package of committed asset values is digitally signed and timestamped, and sent to the investor.

The commitments are binding - once made they can not be de-committed to a different value. This serves as a strong incentive against deliberate misstating of the portfolio. Of course, it is impossible to rule out the possibility that the fund manager lies about the asset quantities $\mathsf{b_i}$ in order to misrepresent the status of the fund. However, the quantity held of a particular asset at a given point in time is an objective piece of information. Making such a false statement would clearly be fraud.

**Third Parties:** We suggest the use of a third party to increase the effectiveness of the fund's incentive to commit honestly to the portfolio. For example, the committed portfolio might also be sent directly to the SEC, or to a different regulatory organization.

When the corresponding de-commitments are included in the message to the SEC, or other third party, this organization can also act as a trusted third party, confirming the correctness of the commitments, against independent information procured about the fund's contents, for example, by examining exchange records, and brokerage transactions. In this manifestation, the investor will have an even stronger guarantee, despite still never learning the asset quantities himself.

An alternative to the SEC would be another independent organization, such as a data storage firm, which would timestamp the commitment data, keep the de-commitments (if included) private, and readily provide the data to the court in case it is subpoenaed. If the protocol is implemented without sending the de-commitments to the third party, the commitments still serve as evidence should the court order them to be opened. A final option is to employ multiple third parties, and use the technique of secret splitting [11] so that two or more entities need to cooperate to obtain the data.

**Computing the Proofs:** The proofs of the form $ZKP_k(\Sigma\ \mathsf{m_i}\ \mathsf{b_i}\ \geq\ \mathsf{Q})$, $ZKP_k(\Sigma \mathsf{m_i b_i} \leq \mathsf{Q})$ are computed according to the process reviewed in Section 3. One technical detail to consider is the choice of the interval length, $k$. The interval should be large enough so that a proof may always be found if the inequality $\Sigma \mathsf{m_i b_i} \geq \mathsf{Q}$, or $\Sigma \mathsf{m_i b_i} \leq \mathsf{Q}$ holds. An upper bound for the required $k$ can be obtained by considering the minimum and maximum possible values of $\Sigma \mathsf{m_i b_i}$.

**Verification Step:** The verification process also follows the process reviewed in Section 3. During the process the investor should also consult the prospectus to obtain the authenticity and completeness of the parameters $\mathsf{m}_\mathsf{i}$ and $\mathsf{Q}$ behind the restrictions $\mathsf{Limit}_\mathsf{j}$. One the proof data is verified to be complete and correct, the investor will know that the claimed statements constraining the assets are correct, relative to the assumption that the commitments themselves were not fraudulently created.

**Failures and Disputes:** If any verification step fails, then the investor knows that a condition of the investment contract has been breached- this should never happen if the fund manager respects the fund composition restrictions. If there is a legitimate reason for the manager to violate a constraint specified in the contract, the manager should not publish a proof-attempt that will fail, but rather address the problem directly. In case of a legal dispute, the commitments can serve as evidence of the claimed portfolio, and as mentioned above, third parties can assist in such a process.

### 4.3 Discussion

It is clear that the fund manager and investor will need appropriate infrastructure to fully benefit from this mechanism, so it may be most applicable to large institutional investors. A hedge fund which is able to offer this kind of additional assurance would be compensated with ability the attract greater business, and the service might be reflected in the fee that the fund is able to charge.

The scheme increases the accountability of the fund manager, as the investor will have continuous confirmation that the fund has not left the acceptable risk range. The mechanism we describe is certainly stronger than the reputation and *post-facto* legal based approaches in place today. Through the deliberate specification of acceptable risk bounds in the fund prospectus, the mechanism provides strong incentive for the fund manager to manage the portfolio in a manner which is more closely aligned with the investors' risk preferences. Conversely, it discourages investment behavior that concentrates enormous risk on an unlikely scenario, unless the investor agrees to this kind of gamble.

## 5 Applications

Portfolio risk depend on the evolving allocation among security types, so we now turn our attention to the task of designing meaningful risk constraints within our framework. These constraints take the form of linear combinations of the asset quantities $\mathsf{A}_\mathsf{i}$, and include direct limitations on the portfolio composition, as well as constraints based on factor exposures and scenario analysis. Clearly, not all portfolio risks can be specified in advance (or with linear constraints), so our framework leaves open the possibility of revealing additional portfolio risk information not stipulated in the prospectus.

**Individual Asset Bounds:** These are simple constraints of the form $\mathsf{b_i} \leq \mathsf{Q}$, which serve to limit the amount invested in a particular single asset $\mathsf{A_i}$. By using this simple constraint for every potential asset, assurance can be obtained that the fund is not placing a significant bet on the performance of a single security.

**Asset Class and Sector Allocation:** Organizing the list of assets into sectors, a bound on the total investment in a particular sector can be expressed as $\Sigma \mathsf{m_i b_i} \leq \mathsf{Q}$, where $\mathsf{m_i}$ are non-zero for the assets within the sector, and represent a weighting according to the asset's price at the fund's inception. Sector allocation statements and proofs relative to *updated* asset prices can also be made, but these bounds can not be contractually guaranteed in the same way.

**Asset Features, Short Positions:** Following the same technique as for sector allocation, the assets can be grouped in any way desired, and an inequality can be constructed bounding the value invested in such a subgroup. An important example of this might be to group the short positions into a group, and bound the amount of asset shorting. This can be accomplished by listing the short positions as distinct assets, or by using constraints of the form $\Sigma \mathsf{m_i b_i} \geq -\mathsf{Q}$. Bounding the acceptive complementary short and long positions limits the risks associated with such extreme leveraging, including *liquidity risk.*

**Current Minimum Value:** An estimation of current value can be communicated by setting the $\mathsf{m_i}$ to be the current price, and the statement $\Sigma \mathsf{m_i b_i} \geq -\mathsf{Q}$ can be proved for any value of $\mathsf{Q}$ less than the actual sum $\Sigma \mathsf{m_i b_i}$. Since such a statement depends on current prices it can not be rigorously guaranteed in the contract, but it may still be a useful piece of information to relate.

**Factor exposures:** These bounds rely on risk models which assign each asset $\mathsf{A_i}$ a factor exposure $\mathsf{e_{i,j}}$ to a particular factor $\mathsf{F_j}$. According to such models, the exposure is an estimation of the sensitivity, $d(value)/d(factor)$, to the factor. To use this kind of constraint, the exposures $\mathsf{e_{i,j}}$ for factor $\mathsf{F_j}$ should be published in the contract. The aggregate sensitivity of the portfolio to $\mathsf{F_j}$ is then $\Sigma \mathsf{e_{i,j} b_i}$, which may be positive or negative. A bound $-\mathsf{Q'_j} \leq \Sigma\ \mathsf{e_{i,j}}\ \mathsf{b_i} \leq \mathsf{Q_j}$, provides a guarantee that the portfolio is not too sensitive to the factor $\mathsf{F_j}$. For example, such constraints might be used to limit the interest rate risk that the portfolio is allowed to take, or the amount of credit risk.

**Scenario analysis:** This kind of bound extends the benefit obtained by considering a single risk factor in isolation. First a set of *scenarios* are selected, denoted $\mathsf{S_j}$, which define a set of potential future trajectories of various economic factors. Next, some model must be used to estimate the value $\mathsf{v_{i,j}}$ of each asset under each scenario. The prospectus lists the battery of scenarios, and also lists the expected value of each asset under each scenario, and makes reference to the modeling technique used. Finally, an "acceptable risk" is agreed upon by listing the portfolio's minimum future value under each scenario described in the contract. The expected future value of the portfolio under scenario $\mathsf{S_j}$ is simply $P_j = \Sigma \mathsf{v_{i,j} b_i}$, so the bound we are interested in takes the form

$$\Sigma \mathsf{v}_{\mathsf{i,j}} \mathsf{b}_{\mathsf{i}} \geq SV_j. \tag{8}$$

Note that the validity of this approach does not dependent on the choice of model: the values $\mathsf{v}_{\mathsf{i,j}}$ must be published, and the investor must find them reasonable to accept the contract. Of course, the manager can not guarantee future portfolio values, but he can guarantee that he will never take a position which will assume less than the contractual minimum value under any of the listed hypothetical scenario, however unlikely he feels that the scenario is.

Such scenarios are idealized, discreet, future possibilities, and the actual outcome is unlikely to closely follows an actual scenario listed. Nevertheless, such bounds are very useful since they force the fund to maintain a composition for which it is not expected to lose too much value under an adversarial scenario.

**Trading volume:** A final type of bound may be useful to detect a certain type of fraud masquerading as "market timing", where redundant trades are made not to improve the portfolio's position, but to earn brokerage fees associated with each trade. To allow a bound on the total trading activity within a fund would require a minor tweak: we provide commitments to the amounts of each asset purchased and sold (these are considered separately, and must each be positive). Then bounds on the total amount of sales (purchases) over some period can also be expressed as linear conditions, and the same types of zero knowledge proofs employed.

## 6 Implementation

To demonstrate the feasibility of our proposal, we implemented a prototype of our scheme using C, and Shoup's NTL package [12]. For this prototype we generated parameters $p$ and $q$ to be 1024 bits and 160 bits respectively, and used SHA1 as the hash function. With these parameters, each commitment was 1024 bits long, and each $k$-bit interval proof was $1664k$ bits long. We set the interval proof length, $k$, to be 30 bits, which is sufficient for the inequalities we would like to prove. This assumes a precision for $\mathsf{m}_{\mathsf{i}}$ and $\mathsf{b}_{\mathsf{i}}$ of about 15 bits each; increased precision would unlikely significantly add to the risk information conveyed.

Each interval proof with parameter $k = 30$ requires a few seconds to compute, and can be reduced to less than 1 second when optimized on a standard 2005 model PC. Assuming a portfolio with several thousand assets $\mathsf{A}_{\mathsf{i}}$ and 1000 constraints $\mathsf{Limit}_{\mathsf{j}}$, the commitments and zero knowledge proofs can be computed in less than twenty minutes, if we assume a cost of 1 second per constraint proof. Of some concern, the proof material does require a substantial amount of space - about 6 megabytes for the parameters [k=30, 1000 constraints]. Elliptic curves, or the techniques in [3] may improve efficiency.

The main conclusion we draw for this experiment is that for a reasonably complex portfolio and set of constraints, the computation can be completed in a matter of minutes, and stored at a reasonable cost. This means that it is feasible to generate and publish the proof data at least once per day, for example, after the major US exchanges are closed.

## 7 Conclusions

This work has introduced, for the first time, the applications of zero knowledge techniques to the release of investment risk material. It is surprising that the relatively simple and well established cryptographic tools of commitment and interval proofs suffice to construct a mechanism to make portfolio composition assertions which can already communicate the most important types of portfolio risks. This follows from the observation that most of the relevant risk assertions (sector allocation, factor exposure, and scenario analysis) are linear in nature.

The premise behind this work is that a verifiable mechanism to communicate risk will increase the trust between an investor and a fund manager, and ultimately create overall economic value. The scheme we describe lies at the crossroads of cryptography, risk management, law, and trust assessment, and is is novel technique to increase accountability of fund managers to investors. The proposed mechanism consists of a contract between the investor and manager, through which the manager agrees to describe the evolving portfolio in a verifiable way. Effectively, the investor will have a new tool to monitor the manager's trades, and to check that the fund characteristics satisfy the risk preferences specified in the contract.

We contend that hedge funds would be more compelling investments, if their performance were not perceived as a magic black-box, often delivering spectacular returns, but occasionally declaring bankruptcy. Indeed, many hedge fund strategies involve taking large positions in oppositely correlated securities, a configuration designed to achieve probable high returns yet only reveal the risks in case of disaster! Despite the fact that the scheme limits the hedge fund manager's choices, he may be motivated to employ our scheme to attract investors who demand real-time risk-exposure information and additional legal assurance.

## Acknowledgments

The author would like to thank Dr. Andrew Lo for motivating discussions.

## References

1. Securities and exchange commission : Web article, 2003. URL: http://www.sec.gov/answers/hedge.htm.
2. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *1st ACM Conference on Computer and Communications Security*, pages 62–73. ACM Press, 1993.
3. F. Boudot. Efficient proofs that a committed number lies in an interval. In Bart Preneel, editor, *Advances in Cryptology - EuroCrypt '00*, pages 431–444, Berlin, 2000. Springer-Verlag. Lecture Notes in Computer Science Volume 1807.
4. E. F. Brickell, D. Chaum, I. B. Damgåard, and J. van de Graaf. Gradual and verifiable release of a secret. In Carl Pomerance, editor, *Advances in Cryptology - Crypto '87*, pages 156–166, Berlin, 1987. Springer-Verlag. Lecture Notes in Computer Science Volume 293.

5. J. Campbell, A. Lo, and C. MacKinlay. *The Econometrics of Financial Markets.* Princeton University Press, New Jersey, 1997.
6. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Y.G. Desmedt, editor, *CRYPTO '94*, pages 174–187. Springer-Verlag, 1994. LNCS no. 839.
7. W. Mao. Guaranteed correct sharing of integer factorization with off-line shareholders. In H. Imai and Y. Zheng, editors, *Proceedings of Public Key Cryptography*, pages 60–71. Springer-Verlag, 1998.
8. T. P. Pedersen. A threshold cryptosystem without a trusted party (extended abstract). In Donald W. Davies, editor, *Advances in Cryptology - EuroCrypt '91*, pages 522–526, Berlin, 1991. Springer-Verlag. Lecture Notes in Computer Science Volume 547.
9. D. Pointcheval and J. Stern. Security proofs for signature schemes. In Ueli Maurer, editor, *Advances in Cryptology - EuroCrypt '96*, pages 387–398, Berlin, 1996. Springer-Verlag. Lecture Notes in Computer Science Volume 1070.
10. Claus P. Schnorr. Efficient identification and signatures for smart cards. In *Proceedings on Advances in cryptology*, pages 239–252. Springer-Verlag New York, Inc., 1989.
11. A. Shamir. How to share a secret. *Communications of the Association for Computing Machinery*, 22(11):612–613, November 1979.
12. V. Shoup. Ntl: A library for doing number theory, 2003. URL: http://www.shoup.net/ntl.

# A Cryptography Details

## A.1 Schnorr OR Proof Properties

We review the security properties of the Schnorr OR Proof. These are completeness, zero-knowledge and special soundness. The non-interactive version of the proof, also called a sigma protocol [6], is made non-interactive with the Fiat-Shamir transform. Replacing the role of the verifier with a hash function, the non-interactive proofs are proved secure in the Random Oracle Model [2]. There is no known attack on this proof when the random oracle is replaced with a good (one-way, and collision-free) hash function such as SHA1.

**Completeness:** For any commitment $C_r(0)$, or $C_r(1)$, such a proof can always be efficiently calculated as follows: If $x = 1$ (so $C = g^1h^r$), let $r_1, c_1, u_2$ be random (mod $q$). Let $a_1 = h^{r_1}C^{-c_1}$ (mod $p$), $a_2 = h^{u_2}$ (mod $p$), $c$=hash$(a_1, a_2)$, $c_2 = c - c_1$, and $r_2 = u_2 + c_2r$ (mod $q$). In the case where $x = 0$, (so $C = g^0h^r$), let $r_2, c_2, u_1$ be random (mod $q$), $a_2 = h^{r_2}C/g^{-c_2}$ (mod $p$), $a_1 = h^{u_1}$ (mod $p$), $c$=hash$(a_2, a_1)$, $c_1 = c - c_2$, and $r_1 = u_1 + c_1r$ (mod $q$).

**Zero Knowledge:** The interactive proof is special honest verifier zero knowledge. For any C,c a simulator which chooses $r_1, c_1, r_2, c_2$ at random such that $c = c_1 + c_2$, and computes $a_1 = h^{r_1}C^{-c_1}$ and $a_2 = h^{r_2}C/g^{-c_2}$ perfectly simulates the honest protocol interaction. The non-interactive proof is zero knowledge in the random oracle model.

**Special Soundness:** This sketch shows that two accepting protocol interactions $(a_1, a_2; c; , r_1, r_2, c_1, c_2)$ and $(a_1, a_2; c'; r'_1, r'_2, c'_1, c'_2)$ for a fixed $C$ with different challenges $\{c_1, c_2\} \neq \{c'_1, c'_2\}$ can be used to compute a witness $(x, r)$ for $C = g^x h^r$. Suppose the challenges differ, so either $c_1 \neq c'_1$ or $c_2 \neq c'_2$. In the first case, $h^{(r_1 - r'_1)/(c'_1 - c_1)} = C$, and in the second, $h^{(r_2 - r'_2)/(c'_2 - c_2)} = C/g$. Either way a pair $(x, r)$ satisfying $C = g^x h^r$ is found. By the forking lemma [9], the non-interactive proof is thus sound in the random oracle model.

# Probabilistic Escrow of Financial Transactions with Cumulative Threshold Disclosure

Stanisław Jarecki[1] and Vitaly Shmatikov[2]

[1] University of California, Irvine
[2] University of Texas at Austin

**Abstract.** We propose a scheme for privacy-preserving escrow of financial transactions. The objective of the scheme is to preserve privacy and anonymity of the individual user engaging in financial transactions until the cumulative amount of all transactions in a certain category, for example all transactions with a particular counterparty in any single month, reaches a pre-specified threshold. When the threshold is reached, the escrow agency automatically gains the ability to decrypt the escrows of all transactions in that category (and only that category).

Our scheme employs the *probabilistic polling* idea of Jarecki and Odlyzko [JO97], amended by a novel robustness mechanism which makes such scheme secure for malicious parties. When submitting the escrow of a transaction, with probability that is proportional to the amount of the transaction, the user reveals a share of the key under which all his transactions are encrypted. Therefore, the fraction of shares that are known to the escrow agency is an accurate approximation of the fraction of the threshold amount that has been transacted so far. When the threshold is reached, with high probability the escrow agency possesses all the shares that it needs to reconstruct the key and open the escrows. Our main technical contribution is a novel tool of *robust probabilistic information transfer*, which we implement using techniques of optimistic fair 2-party computation.

## 1 Introduction

Increasing demands by law enforcement and regulatory agencies to monitor financial transactions are gradually eroding individual and organizational privacy. A common legal requirement is that all transactions exceeding a certain threshold (*e.g.*, $10,000 for currency transactions in the U.S.) must be reported to the financial authorities. Moreover, if a financial institution suspects that a customer is engaged in "structuring" his transactions so as to avoid the reporting requirement (*e.g.*, making regular cash deposits just under the reporting threshold), the institution is required to report these transactions, too. This may lead to an unnecessary loss of privacy, since the transactions in question may be innocuous.

Building on the transaction escrow scheme of [JS04], we propose an efficient technical solution to the problem of reporting structured transactions. Our goal is to balance individual privacy with the legally mandated cumulative threshold

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 172–187, 2005.
© Springer-Verlag Berlin Heidelberg 2005

disclosure requirement, *e.g.*, "all transactions of any individual totaling $T$ or more must be disclosed". Our scheme guarantees the following properties:

**Privacy:** With high probability, an individual whose transactions total less than the pre-specified threshold $T$ enjoys *provable* anonymity and privacy. In particular, a malicious escrow agency cannot feasibly open the escrowed transactions whose cumulative amount is less than this threshold.

**Cumulative Threshold Disclosure:** Once the total amount of some individual's escrowed transactions exceeds the pre-specified threshold $T$, then with high probability the escrow agency is able to (i) efficiently identify these transactions among all escrows it has collected, and (ii) automatically open these (and only these) escrows without help from their creator.

We achieve these properties assuming a trusted third party (TTP), which is only invoked *optimistically.* The role of the TTP can be naturally played by the Key Certification Authority, whose presence is required in any case in any realistic transaction escrow system. Our protocols are *optimistic* in the sense that the TTP is contacted only if one of the parties notices that the other one misbehaves. The effect of interaction with the TTP is equivalent to interaction with an honest counterparty in the protocol, hence there is no incentive for either player to diverge from the protocol specification. Therefore, in practice the TTP should only be invoked in the (rare) cases of certain communication failures.

Both privacy and cumulative threshold disclosure properties in our scheme are *probabilistic*: (1) there is a small *probability of erroneous disclosure*, *i.e.*, that some individual's transactions will be revealed to the escrow agency even though they total less than the pre-specified threshold, and (2) there is also a small *probability of erroneous non-disclosure*, *i.e.*, that some individual's transactions will not be disclosed even though they total more than the threshold. Both probabilities decrease sharply with the distance separating the cumulative transaction amount and the threshold $T$ (*i.e.*, it is highly unlikely that privacy of some individual will be compromised if the cumulative amount of his transactions is significantly below $T$, or that he will avoid disclosure if it is significantly higher than $T$). Our scheme provides a tradeoff between the computation and communication complexity of interaction between the user and the escrow agency, and the sharpness of the slope of these functions.

*Overview of Transaction Escrow.* The concept of verifiable transaction escrow was introduced in [JS04], but the escrow scheme in [JS04] does not support *cumulative* disclosure conditions which are the focus of the present paper. Following [JS04], we refer to the individual performing the transaction, *e.g.*, a bank transfer or a stock purchase, as the *user* ($U$), and the escrow agency collecting the escrows as the *agency* ($A$). We assume that $U$ and $A$ communicate over an anonymizing channel. In particular, $U$ may send information to and engage in zero-knowledge protocols with $A$ through a proxy, without revealing $U$'s true identity. We refer to the full description of any transaction as the transaction *plaintext.* We'll say that transactions are *related* if they belong to the same *category,* and to simplify the exposition, we'll equate the category with the user's identity. In real applications, the category of a transaction might be more fine-grained, and determined not only by the user's identity, but also by any predicate on the transaction plaintext, such as the type of the transaction, the payee's identity, the jurisdiction of the payee's financial institution, *etc.*

A transaction escrow scheme, as introduced in [JS04], must ensure **category-preserving anonymity**: the only information the escrow agency can learn from any two escrowed transactions is whether or not they originate from the same user. Importantly, the agency does not learn which user this is.[1] The scheme of [JS04] can also support **simple threshold disclosure**: the agency can efficiently identify and de-escrow all transactions that belong to the same category once the *number* of such transactions reaches a pre-specified threshold.

*Cumulative Disclosure Conditions for Financial Transactions.* A simple threshold disclosure condition described above cannot efficiently support monitoring of financial flows, because financial oversight laws usually call for transactions of a certain type to be reported to the monitoring agency based on the total *value* of the transactions and not just their *number*. Indeed, this objective is difficult to achieve with any system in which disclosure is based just on the number of transactions. No matter how we set the limit which determines when a single transaction needs to be escrowed and the number of transactions that should lead to automatic disclosure, the person performing the transactions can divide his transactions into small pieces, each of which stays below the threshold level.

## 2 Overview of Escrow with Cumulative Disclosure

Let $T$ be the pre-specified cumulative disclosure threshold for transactions originating from a single individual (*e.g.*, $10,000 for financial transactions in the U.S.). Conceptually, we split the threshold $T$ into $d$ parts, *e.g.*, $d = 20$ (in section 6, we discuss how to choose $d$ and we describe the trade-off between efficiency and the probability of erroneous disclosure or non-disclosure). All transactions that belong to the same category are encrypted with the same key, using a verifiable anonymous encryption scheme. The key itself is split by the user into $d$ shares using standard verifiable secret sharing techniques [Fel87].

Our scheme follows the "probabilistic polling" idea proposed by Jarecki and Odlyzko for a micropayment scheme [JO97]. Whenever the user performs a transaction for some amount $t \leq \frac{T}{d}$ (higher amounts need to be subdivided into pieces of at most $\frac{T}{d}$ value) and submits the corresponding escrow to the agency, the user must also reveal one share of his encryption key with probability exactly equal to $\frac{d}{T} * t$. If the probability of submitting a share is set in this way, then regardless of the size $t$ of the individual transactions that make up a cumula-

[1] Note that this requirement precludes the traditional escrow solutions where plaintext data is encrypted under escrow agency's public key, as the escrow agency would then in principle be always able to decrypt all the escrowed data.

tive amount $A$, the expected number of shares generated by a user who escrows $n = \frac{A}{t}$ transactions will be $n(\frac{d}{T} * t) = \frac{A}{T}\ d$, which is independent of $t$.

When total amount reaches $A = T + \delta$, regardless of the pattern of transactions, with probability that grows steeply with $\delta$ the escrow agency will have obtained $d$ shares, enabling it to reconstruct the key and open all escrows of that user. Because the agency cannot feasibly decrypt the escrows until it collects $d$ shares, all transactions of a user whose cumulative transaction value $A$ is $A = T - \delta$ will stay secret with probability that again increases sharply with $\delta$.

*Robust Probabilistic Information Transfer with a Fair Coin Toss.* To guarantee that the share is transferred from the user to the agency with the required probability, we develop a joint coin tossing protocol between the user and the agency based on fair exchange of random contributions (encrypted under the trusted third party's public key) using the standard techniques of optimistic fair exchange of secrets (see, *e.g.*, [ASW00]). In addition to committing to his random contribution, the user verifiably commits to a share of the escrow key, using the verifiable encryption scheme of Camenisch and Shoup [CS03], and "signs" (see below) the transcript of the protocol up to that point. The parties then de-commit their contributions to the joint coin toss, and if the resulting coin toss indicates that the share must be revealed, the user is expected to open his commitment. If the user refuses to de-commit his random contribution correctly, or refuses to reveal the share itself, the agency can appeal to a trusted third party, who will open the escrow and reveal the user's share if the joint coin toss should indeed result in a transfer of the share. Thus neither the user, nor the agency can skew the probability with which the key share is transferred between them.

Note that the agency must be able verify the user's signatures without learning his identity. Since the TTP is allowed to know the user's identity, we combine the unlinkable credentials technique of [CL01] with the verifiable encryption of [CS03] and have the user issue signatures under a public key which is itself encrypted under the TTP's public key. The escrow agency does not learn the user's public key, but can verify that (1) some CA-certified valid public key was used and the TTP will be able to identify it, and (2) the transcript was signed under that key, and the TTP will be able to verify it.

There is a small privacy leak in our scheme since the escrow agency must know the probability with which the information is to be transferred. Since this probability is proportional to the transaction value, the agency essentially learns this value. This leak appears harmless in practice since the agency does *not* learn the identity of the user, or anything else about the transaction plaintext, except that the transaction must be related to some other previously escrowed transactions, and thus that they all originate from the same (unknown) user.

*Related Work.* Our scheme employs Shamir's polynomial secret sharing in such a way that user's revelation of enough shares enable the escrow agency to recover the user's keys and decrypt his/her escrowed data. Similar idea was proposed for secure metering of web accesses by Naor and Pinkas [NP98], but in our scheme this idea is is extended so that (1) it can be used in conjunction with a PKI system, so that the secret sharing is determined by the user's private key, (2) the generated shares must be linkable to each other but unlinkable to their originator, and (3) the shares need to be generated only with some probability, and this probabilistic generation must be fair.

Our notion of a probabilistic information transfer owes much to works on 2-party coin tossing [Blu82] and two-party secure computation in general [Can00]. Our implementation of this functionality utilizes the techniques and the model of the 2-party computation with off-line trusted third party, used *e.g.*, by the secret exchange protocol of Asokan, Shoup, and Waidner [ASW00], and by the general fair 2-party computation protocol of Cachin and Camenish [CC00].

## 3 Model and Definitions

A transaction escrow system involves an *Escrow Agency* and any number of *Users.* Users engage in financial transactions such as stock purchases, wire transfers, *etc.* For the purposes of this paper, we will focus on one application, in which the transactions are wire transfers (or more properly, wire transfer requests) and the counterparties of these transactions (*i.e.*, the entities the perform them on users' behalf) are banks and other financial services providers. As mentioned in the introduction, each transaction is fully described by its *plaintext*, and we define the *category* of the transaction as simply the user's identity. To make this identity unambiguous, we assume a global PKI with a trusted Certification Authority who issues a unique public key credential to every user.

In our scheme the user, knowing the plaintext of his intended transaction, first performs a protocol with the escrow agency in which he sends to the agency a transaction *escrow* and in return receives the agency's receipt. The user then engages in the transaction with the counterparty, and the counterparty verifies that the user holds a valid receipt for this transaction. Note that we have no hope of escrowing transactions in which a counterparty aids the user in avoiding the escrow protocol by foregoing the verification of the escrow receipt. Simply speaking, if some user and counterparty want to conduct an un-monitored transaction, they can. The transaction escrow scheme can help in monitoring only transactions in which at least one of the participants, the user or the counterparty, enables this monitoring. Similarly, the user's privacy against the escrow agency can only be protected for transactions with honest counteparties. A dishonest counterparty can always forward the transaction plaintext to the agency.

We call a transaction escrow scheme $(\alpha_{T,t}, \beta_{T,t})$**-probabilistic cumulative threshold escrow** if it satisfies the following properties, where $\alpha_{T,t}$ and $\beta_{T,t}$ are both functions from real values to the $[0, 1]$ interval, $T$ is the global pre-specified cumulative privacy threshold, and $t$ is the minimum allowed transaction size.

$\alpha_{T,t}$**-probabilistic cumulative threshold disclosure.** Independently for every user, regardless of his transaction pattern, if the user escrows transactions whose total cumulative value equals $A = T + \delta$, then with probability at least $1 - \alpha_{T,t}(\delta)$ (minus a negligible amount), all transactions of *this* user can be efficiently identified and de-escrowed by the agency.

$\beta_{T,t}$**-probabilistic amount-revealing privacy.** For any two escrows $e, e'$ of two transactions conducted with some honest counterparties, the only thing that a (potentially malicious) escrow agency learns about these transactions is (1) whether or not they originate with the same user, and (2) the numerical amounts $val(e)$, $val(e')$ transacted in each case. Moreover, regardless of the user's transaction pattern and of the actions of the escrow agency, if the escrows correspond to transactions whose total cumulative value equals $A = T - \delta$, then all transactions of *this* user are revealed to the agency protection with probability at most $\beta_{T,t}(\delta)$ (plus a negligible amount).

Unlike in [JS04], disclosure depends probabilistically on the cumulative transacted amount. With probability of at least $1 - \alpha(\delta)$, which approaches 1 as $\delta = A - T$ increases, the escrow agency can open all escrowed transactions of a user whose transactions add up to $A$. Therefore, $\alpha$ represents the risk of not being able to open some user's escrows even though their cumulative transacted amount is higher than the threshold. Also, there is an additional privacy relaxation: $\beta$ represents the risk of privacy violation for users whose cumulative transaction amount does not yet reach the pre-specified threshold.

Our escrow scheme is actually a family of schemes, each of which is an $(\alpha_{T,t}, \beta_{T,t})$-probabilistic cumulative threshold escrow scheme for some functions $\alpha_{T,t}$, and $\beta_{T,t}$. As the number of shares increases (and the scheme becomes less efficient), the "accuracy" of probabilistic disclosure gets better in the sense that for any value $t$, the two functions decrease more sharply, which reduces the risk of both erroneous disclosure and erroneous non-disclosure. Both functions decrease slower (and hence get worse) when the minimum transaction size $t$ decreases. However, the impact of $t$ on both these functions seem very small, and we conjecture that $\alpha$ and $\beta$ will stay approximately the same even for very small values of $t$, thus eliminating the need for the minimum transaction size restriction.

## 4 Basic Threshold Escrow

Before summarizing the transaction escrow scheme of [JS04], we'd like to emphasize the difference between that scheme and the scheme proposed in this paper. In [JS04], the disclosure condition is, roughly, as follows: "If the number of transactions, *each* of which originates from the same user and satisfies a particular condition, is greater than some threshold $d$, then open the corresponding escrows." In this paper, the disclosure condition is as follows: "If the transactions *jointly* satisfy a particular condition (namely, the total transacted amount is above some threshold $T$), then open the corresponding escrows." One of the contributions of this paper is to build on the techniques of [JS04] to support a disclosure condition that spans *multiple* transactions of the same user.

### 4.1 Cryptographic Toolkit

Our constructions rely on the hardness of the Decisional Diffie-Hellman (DDH) problem in subgroup $QR_p$ of quadratic residues in $\mathbb{Z}_p^*$, where $p, q$ are large primes such that $p = 2q+1$, and $g$ is a generator of $\mathbb{Z}_p^*$. Our basic cryptographic tool is a *verifiable random function* (VRF) family, implemented in the Random Oracle Model and based on DDH. Let $H : 0, 1^* \rightarrow \mathbb{Z}_p^*$ be an ideal hash function. The VRF family is defined by

(i) the key generation algorithm that picks a secret key $k \in \mathbb{Z}_q^*$ and the corresponding public key $pk = g^{2k} \bmod p$,

(ii) the evaluation algorithm $\mathsf{Eval}_k(x)$ which outputs $y = H(x)^{2k} \bmod p$ and a non-interactive zero-knowledge proof $\pi$ of equality of discrete logarithms $x = \mathsf{DL}_h(y) = \mathsf{DL}_h(pk)$, which proves that the function was computed correctly (such proof can be accomplished in ROM with a few exponentiations using standard techniques, *e.g.*, [CP92]), and

(iii) the verification algorithm for verifying proof $\pi$ of discrete-log equality.

### 4.2 Basic Transaction Escrow with Simple Threshold Disclosure

We assume that every user $U$ is initialized with a secret key $k_U$, chosen at random in $\mathbb{Z}_q^*$, and that the corresponding public key $pk_U = g^{2k_U}$ is signed by the Certification Authority. We assume that the escrow agency has been initialized with the public/private key pair of an *unlinkable* CMA-secure signature scheme of Camenisch-Lysyanskaya [CL01], and that the disclosure threshold $d$ is a global constant. We say that two transactions $m$ and $m'$ belong to the same *category* if and only if they are originate with the same user.[2]

Suppose user $U$ wishes to perform a transaction described by plaintext $m$ with some counterparty $C$. Before carrying out the transaction, $C$ demands that the user present a *receipt* from the escrow agency, proving that the latter has received a correctly formed escrow of the transaction. $U$ starts by picking a unique $(d-1)$-degree secret-sharing polynomial $f$. The coefficients are computed as $k_i = H(k, i)$ where $H : 0, 1^* \rightarrow \mathbb{Z}_q$ is a pseudorandom function, and the polynomial is defined as $f(x) = k_0 + k_1 x + \ldots + k_{d-1}x^{d-1}$ *mod* $q$. Values $\{C_0, \ldots, C_d\}$ where $C_i = g^{2k_i}$ *mod* $p$ serve as commitments to the coefficients.

The user sends to the escrow agency (via an anonymizing channel):

(i) *Tag* $t = \mathsf{Eval}_k(1)$, which allows the escrow agency to separate escrows into categories (note that $t$ is constant for all transactions of the same category);
(ii) *Ciphertext* $c' = (c, \{C_i\}_{i=0,\ldots,d-1}, f(x))$. Here $c = (\mathsf{pad}^H(m|r))^{2k_0}$ *mod* $p$ is the Pohlig-Hellman encryption of (padded) $m$ under the key $k_0$, $\{C_i\}$ are the commitments to the polynomial coefficients, $x = H(c)$ is point in $\mathbb{Z}_q^*$ assigned for $c$, and $f(x)$ is the value of the polynomial on $x$;
(iii) *Anonymous signature* $s$ on $(t, c')$ computed as $s = \mathsf{Eval}_k(t, c')$.

[2] We simplify the scheme of [JS04] by assuming that all transactions are of the same type, and so only the user's identity determines the transaction category.

The escrow agency verifies that $(x, f(x))$, for $x = H(c)$, is a true data point on the polynomial committed to in $\{C_i\}$ by checking that $g^{2f(x)} = C_0 * C_1^x * \ldots * C_{d-1}^{(x^{d-1})}$ *mod* $p$. If there already exist escrows in the same category (*i.e.*, the escrow agency has previously received escrows with the same tag $t$), the agency checks that the commitments $\{C_i\}$ are the same as those supplied with previous escrows in the category. If the checks are successful, the escrow agency signs tuple $(t, s, c, C_0)$ using the unlinkable signature scheme of [CL01], and returns the signature to the user as the escrow *receipt*. We omit the details of the protocol from [JS04] by which user $U$ proves to the counterparty (who knows transaction plaintext $m$ and the $U$'s public key $pk$, but not $U$'s secret $k$) that $U$ possesses the receipt on a correctly formed escrow for this transaction, which implies that $U$ must have given the correctly formed escrow to the escrow agency.

Provided that the escrow receipts are verified by honest counterparties, the above scheme provides automatic and unavoidable threshold disclosure. With each escrow, the user must submit a new *share* $f(x)$ of the $(d-1)$-degree polynomial $f$, and each escrow contains an encryption of the transaction plaintext under the same key $k_0 = f(0)$. Once $d$ escrows have been collected in the same category, the escrow agency can interpolate the polynomial $f$ from the $d$ shares, compute $f(0)$ and decrypt all these escrows. Otherwise, this user's escrows remain secret to the agency.

## 5 Escrow with Cumulative Threshold Disclosure

To replace simple threshold disclosure with *cumulative* disclosure, we need to change the basic protocol of section 4.2 in which the user supplies a single secret-share $s = f(x)$ of the key $k_0$ that encrypts all of his transactions. As explained in section 2, $s$ must be transferred to the agency with probability equal to $\theta = \frac{d}{T} * t$ where $t$ is the value associated with this transaction, a.k.a. *transaction size*. We achieve this using a novel tool we call *robust probabilistic information transfer.*

### 5.1 Probabilistic Information Transfer: Definition

A probabilistic information transfer protocol is a protocol between two parties, user $U$ and agency $A$. The public input is the probability $\theta \in [0, 1]$ with which information transfer should take place, the user's private input is the information that might be transfered, which in our case is the share $s = f(x)$, and the public input is a commitment to this information, which in our case is $C_s = g^s \bmod p$. Because we are interested in protocols that assume a trusted third party, we allow for this protocol to involve the *third* party, $TTP$. Even though a probabilistic information transfer will thus be a protocol between three parties $U$, $A$, and $TTP$, our secure implementation of that notion will involve the $TTP$ party only in case one of the parties is faulty, and thus the protocol we propose works in the "optimistic off-line trusted third party" model, similarly to, *e.g.*, the fair-exchange protocol of [ASW00] or the general fair 2-party computation protocol of [CC00]. As in [CC00], we will assume that $T$ has as the secret input its secret key

$sk_{TTP}$, while $pk_{TTP}$ is publicly known. Finally, we assume that $A$, the agency, has a private/public key pair $(k_A, pk_A)$, too, where $k_A$ is its private key for a VRF function and $pk_A$ is its verification counterpart. Additionally, we allow for an auxiliary public input $aux$, which represents the reference to some transaction to which this transfer refers. In our probabilistic escrow application, we will use $aux$ to represent the escrow $(t, c', s)$ (see section 4.2 above) on account of which this instance of the probabilistic information transfer protocol is executed.

*Ideal Functionality for Probabilistic Information Transfer.* The simplest way to describe our desired security property is to specify the *ideal functionality* $I$ for the protocol, following the secure function evaluation paradigm (*e.g.*, [Can00]). We define a secure probabilistic information transfer protocol as a protocol that securely realizes this ideal functionality $I$ in the *static* adversarial model where the adversary can corrupt (statically) either the user $U$ or the agency $A$, but the trusted third party $TTP$ is never corrupted.[3]

As mentioned above, we assume that the public input in this protocol consists of commitment $C_s = g^s \bmod p$ on $U$'s information $s$, $A$'s public VRF verification key $pk_A$, $TTP$'s public encryption key $pk_{TTP}$, the probability $\theta$, and the auxiliary public input $aux$. Given these public inputs, the ideal functionality $I$ for probabilistic information transfer proceeds as follows. $U$ can contribute some value $s$ or a special symbol $\perp$, which designates $U$'s refusal to participate. $A$ contributes his private key $k_A$ and either of the two special symbols: $\diamond$, which designates $A$'s acquiescence, or $\perp$, his refusal to participate. The $TTP$ party contributes his private key $sk_{TTP}$. The ideal functionality responds as follows. If either party contributes symbol $\perp$, or if $C_s \neq g^s \bmod p$, or if $k_A$ does not correspond to $pk_A$, the ideal functionality $I$ outputs $\perp$ to both $U$ and $A$. Otherwise, $I$ casts a coin $r$ uniformly distributed in $[0, 1]$ and hands $r$ to both $U$ and $A$. Moreover, if $r < \theta$ then $I$ also gives $s$ to $A$. The outputs of $TTP$ are null in every case.

This ideal functionality implements a secure probabilistic information transfer of $s$ from $U$ to $A$ with probability $\theta$, with the following caveats:

(1) The commitment $C_s$ to the information $s$ is known beforehand to $A$, and this commitment could contain some information about $s$.
(2) Whenever both parties start the protocol, but one of them decides to withdraw (by contributing the $\perp$ input), the other party learns about this;
(3) If $U$ decided to proceed, then $A$ learns if the odds came to his disadvantage and the message $s$ has not been transferred to him; and
(4) $U$, too, learns whether the information has been transferred to $A$ or not, and thus this probabilistic transfer protocol is *non-oblivious.*

**Definition 1.** *We call a protocol between $U$, $A$, and $TTP$, a* (statically) secure probabilistic information transfer protocol in the trusted third party model *if it*

[3] We note that Cachin and Camenish [CC00] who define a general notion of *fair* 2-party computation in the same optimistic third party model as we have here, allow for $TTP$ to be corrupted as well, but this extra corruption ability seems unnecessary.

*securely implements the above ideal functionality in the adversarial model where the adversary (statically) corrupts either the $U$ or the $A$ party, but never the $TTP$. We call such protocol* optimistic *if the $TTP$ party is contacted only in case either $U$ or $A$ is corrupted.*

*Contributory Protocols.* In the sequel we will consider only a special class of protocols that realize such functionality securely, namely a "contributory coin-toss" protocols, where the two players $U$ and $A$ make *c*ontributions $r_U$ and $r_A$ which are uniquely defined by the messages sent by each player, where the resulting coin toss is then computed as a deterministic function of these contributions, *e.g.*, $r = r_U \oplus r_A$, and where the information $s$ is transferred if and only if $r < \theta$.

*No Strategic Advantage for $U$.* If a probabilistic information transfer protocol is a secure implementation of the above ideal functionality, then such protocol offers no strategic advantage to $U$ in the following sense. If $U$ ever decides to withdraw from the protocol, he may only do so *before* he learns $A$'s contribution to the joint coin toss, and thus the likely outcome of the protocol. Clearly this is the case for the ideal functionality, and thus $U$'s withdrawal at a midpoint of the protocol must be equivalent to a refusal to engage in the protocol in the first place, and thus can only happen before the coin toss $r$ is decided. Consequently, $U$ cannot gain any advantage by stopping and re-running the protocol on new inputs, since he can stop only when he is still oblivious to the outcome.

Technically, this suggests that there should be a communication round in the protocol which we can call "$U$'s commitment point," such that: (1) If $U$ does not execute this round correctly we say that "$U$ stops before the commitment point", and this is equivalent to $U$ contributing the $\perp$ surrender sign in the ideal world. As discussed above, before this commitment point $U$ has only a negligible advantage in predicting the coin toss that determines whether $s$ is going to be transferred to $A$ or not. (2) If $U$ does send this message correctly, this is equivalent to $U$ actually contributing the correct $s$ input in the ideal world. Therefore, if $U$ stops or diverts from the protocol *after* the commitment point, then an honest $A$ must still get the correct result: a fair coin toss $r$ and the $s$ value if the $(r < \theta)$ condition is satisfied. Most likely, $A$ will have to rely on the trusted third party to retrieve the fairly generated $r$ and, depending on the outcome, the correct $s$, using the messages $U$ sent before (and including) the commitment point.

### 5.2 Probabilistic Information Transfer: Additional Properties

*Observable Accountability.* In any escrow scheme, but especially in our case where the agency learns the monetary values of all escrowed transactions, a corrupt agency may stage a directed denial of service attack against some user by refusing to issue receipts on his escrows. (While the agency does not know the user's identity, all escrows of that user are linkable.) While such a DoS attack cannot be prevented, it should at least be made detectable by an independent observer, say, a journalist. Then a user who believes that he is being denied service can ask the journalist to observe a (re)run of the escrow protocol. If the agency does not reply with a valid receipt, the journalist can observe that the agency is at fault. This "observable accountability" should be satisfied not just by the probabilistic information transfer subprotocol, but also by the entire escrow protocol. (We note, however, that observability in the larger escrow protocol requires some slight modifications to the protocol presented in section 4.2.)

**Observable accountability:** All actions performed by both parties in the execution of the probabilistic information transfer protocol can be verified without revealing any long-term private information.

*Verifiably Deterministic Coin Contribution for $A$.* While giving any outside observer the ability to verify whether the parties follow the protocol correctly can work as a hedge against the denial of service attacks by a malicious agency, it is not sufficient. Suppose that a malicious agency refuses to serve some user if the coin comes out to the agency's disadvantage, but when the user re-runs the protocol, possibly accompanied by an outside observer, the agency performs correctly. This simple cheating strategy for the agent effectively transfers the information $s$ to the agent with probability $1-(1-\theta)^2$, which is greater than $\theta$. To prevent this attack, we will require the following property:

**Verifiable deterministic coin contribution for $A$:** In the algorithm specified by the probabilistic information transfer protocol, $A$'s contribution to the coin toss is a *deterministic function* of (1) $U$'s message which commits $U$'s contribution to the coin toss, and (2) the auxiliary input $aux$ (which in our escrow application will be instantiated with an escrow instance on account of which the probabilistic transfer is taking place). Moreover, if a malicious $A$ attempts to compute its contribution differently, this deviation will be detected by $U$ with an overwhelming probability.

If $A$'s contribution to the coin toss is a deterministic function of $U$'s contribution, and if the protocol is observably accountable, then $A$ gains no advantage by first abandoning the protocol when the coin comes out to its disadvantage, and then agreeing to re-run it. However, $A$'s contribution should be the same only when applied to the same instance $aux$ in the context of which this protocol instance was invoked, thus facilitating only genuine re-runs. Otherwise, a malicious $U$, once discovering a winning combination between his contribution $r_U$ and $A$'s contribution $r_A$ could try to use the same $r_U$ (and hence induce the same lucky $r_A$ response) for many different instances of the protocol.

Note that determinism of $A$'s contribution does *not* imply that $U$ is able to efficiently predict $A$'s contribution to the joint coin toss. In our construction described in section 5.3, $A$'s coin is computed using a verifiable random function (VRF) applied to $U$'s inputs to the protocol. Because $U$ does not know $A$'s private VRF key, the output of the function appears random to $U$, yet the function is deterministic, and $A$ is able to prove that it was computed correctly.

### 5.3 Probabilistic Information Transfer: Implementation

Even though any ideal functionality can be securely realized using secure 2-party computation [Yao82], such general techniques do not seem to yield a practical protocol in our case. Instead, we design an efficient (4-round, small constant number of exponentiations for both parties) protocol which securely achieves our ideal functionality assuming the presence of an *offline* Trusted Third Party (TTP). Thus, following the "optimistic" paradigm in two-party securecomputation, the TTP is only involved in case of some active faults in the protocol. In our application the role of the TTP can be naturally played by the Key Certification Authority, because a trusted KCA is required in our escrow scheme anyway.

Our protocol is observably accountable and "verifiably deterministic" for $A$. Note that any probabilistic protocol for $A$ can be transformed into a deterministic one by simply giving $A$ a private key and asking that all its random choices are computed via a pseudorandom generator or pseudorandom function based on that key. To achieve observable accountability, $A$'s randomness will be generated by a *verifiable* random function (VRF) keyed with $A$'s private key. In our protocol, the other party ($U$) can verify that the pseudorandomness involved in $A$'s crucial moves is computed correctly using this VRF.

*Cryptographic Setup.* Recall that the user $U$ has a private/public VRF keys $(k_U, pk_U)$ (see section 4.1), and message $s \in \mathbb{Z}_q$ to (probabilistically) send to $A$. We assume that commitment $C_s = g^s \bmod p$ to $s$ was made public before the protocol starts. We amend the key generation procedure of the escrow scheme so that $A$ generates a private/public key pair $(k_A, pk_A)$ for the same VRF function. We assume that $U$ knows $A$'s public key $pk_A$. (However, recall that $A$ does *not* know $U$'s public key $pk_U$.) The Key Certification Authority which plays the role of the TTP picks a private/public key pair $(sk_{TTP}, pk_{TTP})$ of a verifiable encryption scheme of Camenish-Shoup [CS03], with the plaintext space including elements of $\mathbb{Z}_q$. We will use the CS encryption to allow $U$ to prove to $A$ that the plaintext $s$ corresponding to an encrypted value $c_s = Enc_{PK_{TTP}}(s)$ satisfies an equation $g^s = C_s \bmod p$. Such proof takes only a few exponentiations and is non-interactive in the random oracle model (see [CS03] for more details). We assume that the required probability $\theta$ can be rounded up as $\theta = i/2^l$ for some integers $l$ and $i \in [0, 2^l]$. We will assume a second hash function $H' : \{0,1\}^* \rightarrow \{0,1\}^l$, which we will also model as a random oracle.

*Robust Probabilistic Information Transfer Protocol with Off-Line TTP:*

1. $U$ picks a random $r'_U \in \mathbb{Z}_p^*$, computes $c_U = Enc_{PK_{TTP}}(r'_U)$ and $c_s = Enc_{PK_{TTP}}(s)$, and sends $(c_U, c_s)$ to A. $U$ also sends a non-interactive zero-knowledge proof [CS03] that the plaintext $s$ encrypted in ciphertext $c_s$ satisfies relation $g^s = C_s \bmod p$.
2. After verifying the proof, $A$ computes $r'_A = \mathsf{Eval}_{k_A}(c_U, aux)$ and sends $c_A = Enc_{PK_{TTP}}(r'_A)$ to $U$.
3. $U$ sends back to $A$ a MAC value $h = \mathsf{Eval}_s(aux, \theta, C_c, c_s, c_U, c_A)$ on the transcript so far using $s$ as the MAC key, together with a zero-knowledge

proof that $h$ is computed correctly under the key $s$ committed to in $C_s = g^s \bmod p$. Note that if $s$ is treated as a VRF key, then $C_s$ is its corresponding verification key, and thus this is the same VRF verification as discussed in section 4.1. This communication round is the "commitment point" for $U$ in the protocol.

4. If everything verifies, $A$ opens $c_A$ as an encryption of $r'_A$ by sending $r'_A$ to $U$ together with the random coins used in this encryption. $A$ also proves that $r'_A$ is correctly computed as $r'_A = \mathsf{Eval}_{k_A}(c_U, aux)$.
5. If $A$'s de-commitment and the proof are correct, $U$ similarly opens to $A$ his ciphertext $c_U$ as an encryption of $r'_U$. $U$ also computes $r = (r_U \oplus r_A)/2^l$ where $r_U = H'(r'_U)$ and $r_A = H'(r'_A)$. If $r < \theta$ then $U$ also sends $s$ to $A$.
6. If $U$'s de-commitment is correct, $A$ computes $r$ the same way as $r = (r_U \oplus r_A)/2^l$ where $r_U = H'(r'_U)$ and $r_A = H'(r'_A)$. If $r < \theta$ and $A$ doesn't get $s$ from $U$, or $g^s \neq C_s \bmod p$, then $A$ hands $(aux, \theta, C_s, c_s, c_U, c_A, h)$ to TTP, together with the proof that $r'_A = \mathsf{Eval}_{k_A}(c_U, aux)$.
7. TTP decrypts $s = Dec_{sk_{TTP}}(c_s)$, $r'_U = Dec_{sk_{TTP}}(c_U)$, $r'_A = Dec_{sk_{TTP}}(c_A)$, verifies $A$'s proof that $r'_A$ is computed as $A$'s VRF on input $(c_U, aux)$, checks if $h = \mathsf{Eval}_s(aux, \theta, C_s, c_s, c_U, c_A)$. If any verification fails, TTP sends $\perp$ back to $A$ and stops. Otherwise, TTP recomputes $r_U = H'(r'_U)$, $r_A = H'(r'_A)$, $r = (r_A \oplus r_U)/2^l$. If $r < \theta$, then TTP sends $(r, s)$ to $A$, else sends $r$.

**Theorem 1.** *The above protocol is a robust probabilistic information transfer protocol in the optimistic trusted third party model. This is a a contributory protocol which is also observably accountable, and has a verifiably deterministic coin contribution for A.*

We postpone the proof to the post-proceedings version of the paper.

*Performance.* We estimate our scheme's performance by counting the number of cryptographic operations that the user and the agency must execute in each session. Let $C_e$ be the cost of a single full exponentiation modulo 1KBit modulus. In our setting, the cost of Camenisch-Shoup encryption is approximately $10.5C_e$, and the cost of the associated proof is approximately $13.5C_e$. Assuming that each multi-exponentiation costs between $1.15C_e$ and $1.3C_e$, we estimate that the user has to perform the equivalent of $52.3C_e$ in each protocol session, while the escrow agency's cost is $29C_e$ ($30C_e$ if a share is transferred).

## 6 Accuracy of Probabilistic Threshold Escrow

To estimate accuracy, we are interested in the probability $\alpha_{T,t}$ of *erroneous non-disclosure*, *i.e.*, that the total transacted amount exceeds threshold $T$, but the escrow agency has not accumulated enough shares to reconstruct the decryption key, and the probability $\beta_{T,t}$ of *erroneous disclosure*, *i.e.*, that the escrow agency accumulates enough shares to reconstruct the decryption key even though the transacted amount is still under the threshold.

Suppose the decryption key is split into $d$ shares ($d$ is a parameter of the system). We'll call $s = \frac{T}{d}$ *share size*. This is the amount "corresponding" to one share of the key. Suppose that the user transacts some total amount $A$, and, for simplicity, assume that all transactions are of equal size $t$. If $t < s$, then for each instance of the probabilistic escrow protocol, the probability of revealing a share is simply $\frac{t}{s}$. If $t = is + x$ where $i > 0$ and $x < \frac{T}{n}$, the escrow agency demands $i$ shares straight away, and then engages in the probabilistic escrow protocol in which the probability of revealing an additional share is $\frac{x}{s}$.

W.l.o.g., assume that $t < s$. Let $n = \frac{A}{t}$ be the number of transactions performed. Since for each transaction the probability of obtaining a share $\frac{t}{s} = d\frac{t}{T}$, the probability of obtaining exactly $d$ shares after $n$ transactions is the binomial probability $\binom{n}{d}(d\frac{t}{T})^d(1 - d\frac{t}{T})^{n-d}$, where $\binom{n}{d}$ is the binomial coefficient $\frac{n!}{(n-d)!d!}$. The probability that the escrow agency obtains fewer than $d$ shares is the "tail" of the binomial probability distribution $p_{nd} = \sum_{i=0}^{d-1} \binom{n}{i}(d\frac{t}{T})^i(1 - d\frac{t}{T})^{n-i}$. The probability of disclosure is $p_d = 1 - p_{nd}$. Unfortunately, for realistic applications the number of trials $n$ is insufficiently large to approximate the binomial distribution with a normal or Poisson distribution. Therefore, we do not attempt to derive a closed formula approximating $p_{nd}$.

*Probability of Error.* Probability of error is equal to $p_{nd}$ if the total transacted amount is greater than or equal to the threshold, and to $p_d$ if the total amount is less than the threshold. In fig. 1, we set the disclosure threshold $T = \$10,000$,

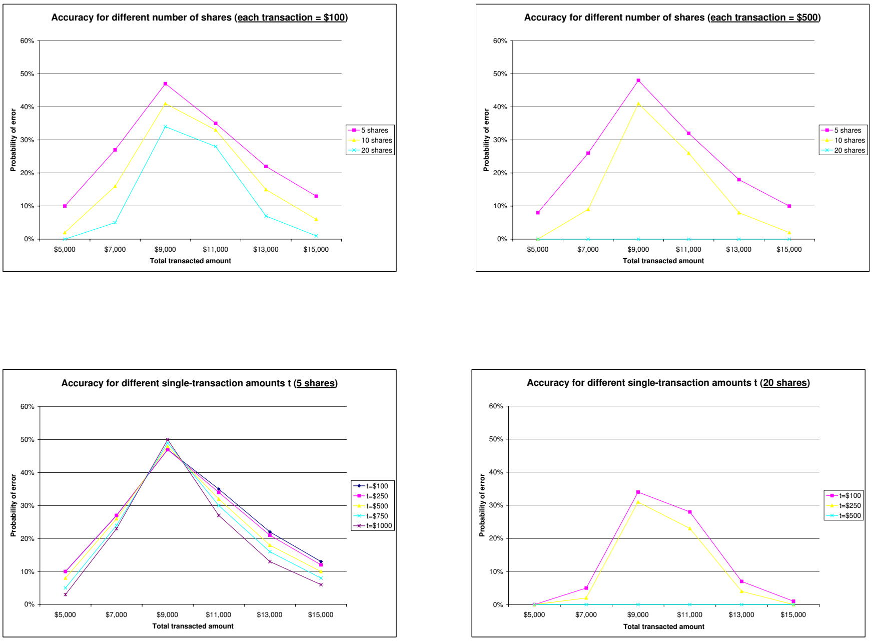


**Fig. 1.** Accuracy of probabilistic threshold disclosure

and calculate the probability of error as a function of the total transacted amount for different transaction sizes $t$ and different number of shares $d$.

Figure 1 illustrates the basic efficiency-accuracy tradeoff of our probabilistic escrow scheme. For a larger number of shares, accuracy is better because (a) for any given transaction size $t$, both $\alpha$ and $\beta$ functions (respectively, left and right sides of the "bell curve") become *steeper*, *i.e.*, the likelihood of erroneous disclosure or non-disclosure decreases sharply with the difference between the total transacted amount and the threshold, and (b) absolute probability of error decreases with the increase in the number of shares. The larger the number of shares, the less efficient the scheme is from the user's viewpoint, due to the difficulty of maintaining a large number of shares.

For a fixed number of shares and total transacted amount, lower single-transaction amounts are associated with higher probabilities of error, as demonstrated by fig. 1. Therefore, the best strategy for a malicious user who would like to transact an over-the-threshold amount without disclosure is to split the amount into lots of small transactions. Note, however, that the curve flattens as transaction sizes decrease. We conjecture that the marginal benefit to the cheating user from choosing ever smaller transactions is negligible. We also argue that for any minimum transaction size $t$, the spending pattern modeled in the tables (*i.e.*, total amount $A$ is split into equal transactions, each of the minimum permissible size) is the worst-case scenario, and that for different transactions patterns probabilities of erroneous disclosure or non-disclosure will be better than those shown in the figures.

*Future Directions.* We are currently investigating an extension of the scheme which is *oblivious* to the user, *i.e.*, he does not learn if information transfer has been successful. The user won't be able to "game" the system by adjusting his behavior depending on the number of shares already accumulated by the agency.

## References

[ASW00] N. Asokan, V. Schoup, and M. Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communications*, 18:593–610, 2000.

[Blu82] M. Blum. Coin flipping by phone. *In Proc. 24th IEEE Computer Conference (CompCon)*, 15(1):93–118, 1982.

[Can00] Ran Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[CC00] C. Cachin and J. Camenisch. Optimistic fair secure computation. In *Proc. CRYPTO '00*, pages 93–111, 2000.

[CL01] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Proc. EUROCRYPT '01*, pages 93–118, 2001.

[CP92] D. Chaum and T. Pedersen. Wallet databases with observers. In *Proc. CRYPTO '92*, pages 89–105, 1992.

[CS03] J. Camenisch and V. Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *Proc. CRYPTO '03*, pages 126–144, 2003.

[Fel87] P. Feldman. A practical scheme for non-interactive verifiable secret sharing. In *Proc. FOCS '87*, pages 427–438, 1987.

[JO97] S. Jarecki and A. Odlyzko. An efficient micropayment scheme based on probabilistic polling. In *Proc. Financial Cryptography '97*, pages 173–192, 1997.

[JS04] S. Jarecki and V. Shmatikov. Handcuffing Big Brother: an abuse-resilient transaction escrow scheme. In *Proc. EUROCRYPT '04*, pages 590–608, 2004.

[NP98] M. Naor and B. Pinkas. Secure and efficient metering. In *Proc. EUROCRYPT '98*, 1998.

[Yao82] A. Yao. Protocols for secure computations. In *Proc. FOCS '82*, 1982.

# Views, Reactions and Impact of Digitally-Signed Mail in e-Commerce

Simson L. Garfinkel[1], Jeffrey I. Schiller[1], Erik Nordlander[1], David Margrave[2],
and Robert C. Miller[1]

[1] MIT, Cambridge, MA 02139, USA
{simsong, jis, erikn,rcm}@mit.edu
[2] Amazon.com, Seattle, WA
DavidMA@amazon.com

**Abstract.** We surveyed 470 Amazon.com merchants regarding their experience, knowledge and perceptions of digitally-signed email. Some of these merchants (93) had been receiving digitally-signed VAT invoices from Amazon for more than a year. Respondents attitudes were measured as to the role of signed and/or sealed mail in e-commerce. Among our findings: 25.2% of merchants thought that receipts sent by online merchants should be digitally-signed, 13.2% thought they should be sealed with encryption, and 33.6% thought that they should be both signed and sealed. Statistically-significant differences between merchants who had received the signed mail and those who had not are noted. We conclude that Internet-based merchants should send digitally-signed email as a "best practice," even if they think that their customers will not understand the signatures, on the grounds that today's email systems handle such signatures automatically and the passive exposure to signatures appears to increase acceptance and trust.

## 1 Introduction

Public key cryptography can be used to *sign* a message so that the recepient can verify that the message has not been modified after was sent. Cryptography can also be used to *seal* the contents of an electronic message so that it cannot be deciphered by anyone who does not have a corresponding key — presumably anything other than the intended recipient.

These two cryptographic primitives—signing and sealing—have been at the root of public key cryptography since its invention in the 1970s. Over the past two decades the Internet community has adopted three standards—Privacy Enhanced Mail, OpenPGP, and S/MIME—all of which are designed to allow Internet users to exchange email with integrity and privacy protections. Support for two of these standards, OpenPGP and S/MIME, has been widely available since 1997. Nevertheless, email messages that are either digitally-signed or sealed are a rarity on the Internet today. [1]

The lack of cryptographic participation is all the more surprising when one considers the real need for this technology in today's electronic marketplace:

- Email can easily be modified in transit, misdelivered to the wrong recipient, or copied without the knowledge of the correspondents.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 188–202, 2005.
© Springer-Verlag Berlin Heidelberg 2005

- In recent years Internet users have been beset by a deluge of so-called “phishing” email messages—messages that purport to be from a respected bank or other financial institution that direct the recipients to bandit websites that exist for the purpose of stealing usernames and passwords. [2]
- Many email messages and computer viruses attempt to trick the recipient into opening the message by using a forged “From:” address.

Ironically, these are the very kinds of attacks that were supposed to be prevented by cryptography.

### 1.1 Usability Barriers

Usability barriers such as difficult-to-use software and confusing terminology [3] are widely perceived as the primary reason why organizations and individuals have not adopted secure messaging technology.

It is easy to understand why usability barriers have affected the exchange of cryptographically sealed mail: two people cannot exchange such messages unless they have compatible software and possess each others’ public keys. And even if keys have been exchanged and have been certified, there is always a risk that the recipient will be unable to unseal the message after it is received—perhaps because the key is lost after the message was sent. For messages that do not obviously require secrecy, many correspondents think that the risk of unauthorized interception is not worth the effort of encryption.

Widespread deployment of digitally-signed mail has been blocked by many barriers. An initial barrier was the deployment of four different and mutually-incompatible standards for signed email: Privacy Enhanced Mail [4–6], PGP clear-signed signatures [7], OpenPGP MIME [8, 9], and S/MIME [10, 11]. The obvious problem caused by competing standards is that there is no guarantee that a signed message, once sent, will be verifiable by the recipient. A deeper problem is that signatures, and sometimes the original email message itself, appear as indecipherable attachments when they are received by email clients that implement the other MIME-based standard.

The wide-scale deployment of mail clients implementing the S/MIME standard has largely solved the standardization problem. Support for S/MIME is built-in to Microsoft Outlook, Outlook Express, Mozilla and Netscape. What’s more, keys for several popular certification authorities (CAs), such as VeriSign, are distributed both with these programs and with many popular operating systems. Thus, while *sending* digitally-signed mail is still relatively cumbersome (requiring that the user obtain a key and procure a digital certificate signed by an established CA), there is a high likelihood that properly-signed mail, once sent, can be readily verified. Nevertheless, few individuals or organizations appear to be sending digitally-signed mail.

### 1.2 Genesis of the Survey

EU Directive 99/93/EU calls for the use of advanced digital signatures for certain kinds of electronic messages. “Advanced digital signatures” are generally taken to mean digital signatures, signed with a private key, that permits the recipient to determine whether or not the contents of the document were modified after the document was sent.

Amazon Services Europe S.à r.l. started sending signed electronic Value Added Tax (VAT) invoices to each of its Amazon Marketplace, Auctions, and zShops sellers in June 2003. Amazon's signatures were S/MIME digital signatures certified by a VeriSign Class 1 Digital ID.

Amazon does not send digitally-signed messages to its sellers operating in America, Asia, and other geographic regions. Because some sellers were receiving signed messages and some were not, we decided to survey Amazon's sellers to discover their reaction to these messages in particular and digitally-signed messages in general.

Digital signatures assure the integrity of email, but did the recipients of the signed email think that such messages were more trustworthy or more likely to be truthful than messages that were not digitally-signed? Did the sellers even know what a digital-signature was, or know that they were receiving them? How did receiving these signatures change the seller's opinion of Amazon? And to what other purposes did the sellers think digital certification should be applied?

### 1.3 Prior Work

We have found very few published studies of popular attitudes regarding encryption and other security technologies. As previously noted, Gutmann suggests that digitally-signed messages comprise a tiny percentage of the non-spam messages that traverse the Internet each day. [1] The 10th GVU WWW User Survey [12] found that a majority of respondents described themselves very (52.8%) or somewhat (26.7%) concerned about security. Nevertheless, "the most important issue facing the Internet" most frequently selected by GVU's respondents was privacy (19.1%); "security of e-commerce" ranked $8^{\text{th}}$ garnering just 5% of the votes.

Whitten and Tygar's study of PGP 5.0 [3] confirmed popularly-held beliefs that even software with attractive graphical user interfaces can have stunning usability problems. But Whitten and Tygar only measured the difficulty of *sending* encrypted mail and key management; they didn't measure their subjects' ability to *receive* and understand the significance of digitally-signed mail.

## 2 Methodology

Our survey consisted of 40 questions on 5 web pages. Respondents were recruited through a set of notices placed by Amazon employees in a variety of Amazon Seller's Forums. Participation was voluntary and all respondents were anonymous. Respondents from Europe and The United States were distinguished through the use of different URLs. A cookie deposited on the respondent's web browser prevented the same respondent from easily filling out the survey multiple times.

A total of 1083 respondents clicked on the link that was posted in the Amazon forums in August 2004. Of these, 470 submitted the first web page, and 417 completed all five pages. We attribute this high follow-through rate to the brevity of the survey: each page took on average 2 minutes to complete.

### 2.1 Characterizing the Respondents

The average age of our respondents was 41.5. Of the 411 responding, 53.5% identified themselves as female, 42.6% as male, and 3.9% chose "Declined to answer." The sample was highly-educated, with more than half claiming to have an advanced degree (26.1%) or a college degree (34.9%), and another 30.0% claiming some college education. More than three quarters described themselves as "very sophisticated" (18.0%) or "comfortable" (63.7%) at using computers and the Internet. Roughly half of the respondents had obtained their first email account in the 1990s, with one quarter getting their accounts before 1990 and one quarter getting their accounts after 1999.

### 2.2 Segmenting the Respondents

The survey contained four tests for segmenting the respondents:

- We can divide our sample according to whether they accessed the survey from the URL that was posted to the Amazon forums frequented by European sellers or those accessed by American sellers. We call these groups *Europe* and *US*. As noted, Amazon has been sending sellers in the *Europe* group digitally-signed email since June 2003, while those in the *US* group have never been sent digitally-signed email from Amazon. A few recepients of digitally-signed messages sent messages back to Amazon such as "what is this `smime.p7s` attachment? I can't read it!" Nevertheless, the vast majority of them did not comment before the study either favorably or negatively on the digitally-signed messages. There were 93 respondents in the *Europe* group and 376 in the *US* group.
- An alternative partitioning is between respondents who have some experience or stated knowledge with encryption technology and those that do not. We selected respondents who met any of the following criteria:
  - Respondents who had indicated that their "understanding of encryption and digital signatures" was 1 ("very good") or who indicated that their understanding was a 2 on 5-point scale (with 5 listed as "none")—23 and 53 respondents, respectively;[1]
  - Respondents who indicated that they had received a digitally-signed message (104 respondents);
  - Respondents who indicated that they had received a message that was sealed with encryption (39 respondents);
  - Respondents who said they "always," or "sometimes," send digitally-signed messages (29 respondents);

  A total of 148 respondents met one or more of these criteria. We called this the *Savvy* group—they were savvy because they had some experience with encryption

[1] We asked our segmenting questions before defining terms such as *encryption* and *digital signature*. Although this decision resulted in some criticism from respondents, we wanted to select those in the *Savvy* based on their familiarity with the terminology of public key cryptography (e.g. "digitally-sign," "encrypt"), rather than the underlying concepts, since user interfaces generally present the terminology without explanation.

**Table 1.** "When were your born?"

| Group | Year | $N$ | $\sigma$ |
|---|---|---|---|
| ALL | 41.5 | 407 | 12.36 |
| Europe | **36.2** | 74 | 10.81 |
| US | **42.7** | 333 | 12.38 |
| Savvy | **38.0** | 135 | 11.74 |
| Green | **43.2** | 272 | 12.28 |

or had self-identified themselves as knowing more about encryption than the average person. Those individuals not in the *Savvy* group were put in a second group called *Green*.

Thus, the *Europe/US* division measures the impact on attitudes given the actual experience in receiving digitally-signed mail from Amazon, while the *Savvy/Green* division measures the impact of people's stated knowledge of or experience with both digital signatures and message sealing.

Results of partitioning the respondents into two groups are deemed to be statistically significant if a logistic regression based on a Chi-Square test yielded a confidence level of $p = 0.05$ for the particular response in question. Such responses are printed **in bold** and necessarily appear in pairs. Where the confidence level is considerably better than $p = 0.05$ the corresponding confidence level is indicated in a table footnote. The lack of bold type does not indicate that findings are not statistically significant; it merely indicates that there is no statistically-significant difference between the two groups.

We performed analysis in terms of education for our segments. Overall, both the *Europe* and *Savvy* groups were younger (Table 1) and less educated (Table 2) than their *US* and *Green* counterparts—differences that were statistically significant.

**Table 2.** "What's your highest level of education:"

| | ALL | Europe | US | Savvy | Green |
|---|---|---|---|---|---|
| Some high school | 2% | 4% | 1% | **4%** $^{*}$ | **1%** $^{*}$ |
| Completed high school | 7% | **16%** $^{**}$ | **5%** $^{**}$ | 8% | 7% |
| Some college | 30% | 27% | 31% | 31% | 29% |
| College degree | 35% | 30% | 36% | **27%** $^{*}$ | **39%** $^{*}$ |
| Advanced degree | 26% | 23% | 27% | 29% | 25% |
| Total Respondents | 410 | 74 | 336 | 137 | 273 |
| No Response | (7) | (1) | (6) | (1) | (6) |

$^{*}p < .05$; $^{**}p < .01$;

As Table 3 shows, many people who had received digitally-signed mail from Amazon were not aware of the fact. The fact that roughly half of these individuals indicated that they had not received such a message suggests that differences in opinion regarding digitally-signed mail between *Europe* and *US* may be attributable to passively experiencing the little certificates in the user interface that are displayed when programs such as Outlook Express receive digitally-signed messages—and not to any specific instruction or indoctrination about the technology.

**Table 3.** "What kinds of email have you received? Please check all that apply"

| | ALL | Europe | US |
|---|---|---|---|
| Email that was digitally-signed | 22% | **33%** $^{**}$ | **20%** $^{**}$ |
| Email that was sealed with encryption so that only I could read it. | 9% | **16%** $^{*}$ | **7%** $^{*}$ |
| Email that was both signed and sealed. | 7% | 10% | 6% |
| I do not think that I have received messages that were signed or sealed. | 37% | 30% | 39% |
| I have not received messages that were signed or sealed. | 21% | 23% | 20% |
| I'm sorry, I don't understand what you mean by "signed," "sealed" and "encrypted". | 26% | **17%** $^{*}$ | **28%** $^{*}$ |
| Total Respondents | 455 | 88 | 367 |
| No Response | (15) | (5) | (9) |

$^{*}p < .05$; $^{**}p < .01$;

### 2.3 Evaluating the Segments

To evaluate our segments, we compared their responses to two test questions. One question asked users, "Practically speaking, do you think that there is a difference between mail that is digitally-signed and mail that is sealed with encryption?" The correct answer was "yes:" sealing renders the message unintelligible to all but the intended recipients, while signatures provide integrity and some assurance of authorship. As shown in Table 4, both *Europe* and *Savvy* demonstrated a significantly higher understanding of digital signatures than *US* or *Green*. (Although we also received a higher percentage of "no" answers, the increase was not statistically significant at $p = 0.05$.)

We also asked respondents if they thought there was a difference between messages that were sealed with encryption and messages that were both signed and sealed. Once again, the answer to this question is "Yes," with both *Europe* and *Savvy* understanding this distinction more than their counterparts, as shown in Table 5.

## 3 Results

Respondents were asked a variety of questions as to when they thought that it was appropriate to use digital signatures for signing or encryption for sealing electronic mail. They were also asked questions on a 5-point scale regarding their opinion of organizations that send signed mail.

**Table 4.** "Practically speaking, do you think that there is a difference between mail that is digitally-signed and mail that is sealed with encryption?" [The correct answer is "yes"]

| | ALL | Europe | US | Savvy | Green |
|---|---|---|---|---|---|
| Yes | 54% | **67%** $^{**}$ | **51%** $^{**}$ | **78%** $^{***}$ | **42%** $^{***}$ |
| No | 7% | 7% | 7% | 10% | 5% |
| Don't know | 39% | **26%** $^{**}$ | **43%** $^{**}$ | **12%** $^{***}$ | **52%** $^{***}$ |
| Total Respondents | 452 | 86 | 366 | 146 | 306 |
| No Response | (18) | (7) | (10) | (2) | (16) |

$^{**}p < .01$; $^{***}p < .001$;

**Table 5.** "Practically speaking, do you think that there is a difference between mail that is sealed with encryption so that only you can read it, and mail that is both sealed for you and signed by the sender so that you can verify the sender's identity" [The correct answer is "yes"]

| | ALL | Europe | US | Savvy | Green |
|---|---|---|---|---|---|
| Yes | 51% | **62%** $^{*}$ | **48%** $^{*}$ | **71%** $^{***}$ | **41%** $^{***}$ |
| No | 8% | 9% | 8% | 11% | 7% |
| Don't know | 41% | **28%** $^{**}$ | **44%** $^{**}$ | **18%** $^{***}$ | **53%** $^{***}$ |
| Total Respondents | 452 | 85 | 367 | 146 | 306 |
| No Response | (18) | (8) | (9) | (2) | (16) |

$^{*}p < .05$; $^{**}p < .01$; $^{***}p < .001$;

### 3.1 Ability to Validate Digitally-Signed Mail

The first matter of business was to determine whether or not respondents could in fact validate digitally-signed mail. For the majority, the answer was an unqualified "yes:" The vast majority of our respondents used Microsoft Outlook Express (41.8%), Outlook (30.6%), or Netscape (10.1%) to read their mail—all of which can validate email signed with S/MIME signatures. Adding in other S/MIME compatible mail readers such as Apple Mail and Lotus Notes, we found that 81.1% could validate digitally-signed messages.

Many of our users didn't know that they could handle such mail, however. We asked users if their email client handles encryption, giving them allowable answers of "Yes," "No," "I don't know" and "what's encryption?" and found that only 26.9% responded in the affirmative.

### 3.2 Appropriate Uses of Signing and Sealing

It has long been argued by encryption advocates that encryption should be easy-to-use and ubiquitous — that virtually all digital messages should be signed, at least with anonymous or self-signed keys, and many should be sealed, as well.

Our respondents feel otherwise. When asked what kind of protection is appropriate for email, respondents answered that different kinds of email require different kinds of protection. In many cases the results of these answers were significantly different for the group that had been receiving digitally-signed messages versus the group that had not been.

**Commercially-Oriented Email (Tables 6, 7 and 8).** Typical email exchanged between merchants and consumers includes *advertisements* from the merchant to the consumer, *questions* that the consumer may pose the merchant, and *receipts* that the merchant may send the consumer after the transaction takes place. The consumer may send the merchant additional follow-up questions. Given that these are typical kinds of messages our respondents exchange with their customers, we sought to discover what level of security our respondents thought appropriate.

Roughly 29% of all respondents agreed with the statement that advertisements should never be sent by email. (This question did not distinguish between email that should not be sent because it might be considered "spam" and messages that should not be sent by email because their content is too sensitive, but comments from respondents indicated that many took this question to be a question about unsolicited commercial email.)

Very few respondents (14%) thought advertisements should be digitally-signed—a surprising number, considering that forged advertisements would definitely present many merchants with a significant problem. Instead, a majority of respondents (54%) thought that advertisements require no special protection at all.

Likewise, few respondents thought that questions to online merchants required any sort of special protection. Remember, *all respondents in the survey are online merchants* — so these merchants are basically writing about what kind of messages they wish to receive. Interestingly, our two groups with either actual or acknowledged experience thought that questions to merchants required *less protection* than their counterpart groups.

This result is surprising because Europeans are generally thought to be more concerned in the privacy practices of businesses than are Americans. One possible explanation for these results is that experience with digital signatures led the Europeans to conclude that a signed receipt was sufficient protection; another explanation is that a significant number of Americans misunderstood the question.

On the other hand, a majority of all respondents (58.8%) thought that receipts from online merchants should be digitally signed, while a roughly a third (46.8%) thought that receipts should be sealed with encryption. Of course, this is not the case with the vast majority of receipts being sent today.

**Personal Email - At Home and Work (Tables 9 and 10).** For years advocates of cryptography have argued that one of the primary purposes of the technology is to protect personal email sent or received at home and at work. The respondents to our survey found no strong desire for technical measures to ensure either integrity or privacy. Even more noteworthy, respondents in the *Europe* and *Savvy* groups saw fewer needs for protection than those in the *US* and *Green* group. One explanation for this result is that increased exposure to security technology increases one's confidence in the computer infrastructure — *even when that technology is not being employed.* Another explanation is that generally more stringent privacy legislation in Europe has removed eavesdropping as a concern from many people's minds.

**Table 6.** "Advertisements"

| | ALL | Europe | US | Savvy | Green |
|---|---|---|---|---|---|
| Does not need special protection | 54% | 58% | 53% | 52% | 54% |
| Should be *digitally-signed* | 14% | 14% | 14% | 18% | 12% |
| Should be *sealed* with encryption | 1% | 1% | 1% | 2% | 0% |
| Should be *both* signed and sealed | 3% | 1% | 3% | 2% | 3% |
| Should never be sent by email | 29% | 26% | 30% | 26% | 30% |
| *sealed* or *both* | 3% | 3% | 4% | 4% | 3% |
| *digitally-signed* or *both* | 17% | 15% | 17% | 20% | 15% |
| Total Respondents | 429 | 78 | 351 | 142 | 287 |
| No Response | (4) | (2) | (2) | (0) | (4) |

**Table 7.** "Questions to online merchants"

| | ALL | Europe | US | Savvy | Green |
|---|---|---|---|---|---|
| Does not need special protection | 61% | 69% | 59% | 67% | 58% |
| Should be *digitally-signed* | 20% | 15% | 21% | 18% | 20% |
| Should be *sealed* with encryption | 5% | 6% | 5% | 6% | 5% |
| Should be *both* signed and sealed | 13% | 9% | 14% | **8%** $^{*}$ | **15%** $^{*}$ |
| Should never be sent by email | 1% | 0% | 1% | 0% | 1% |
| *sealed* or *both* | 18% | 15% | 19% | 14% | 20% |
| *digitally-signed* or *both* | 33% | 24% | 34% | **26%** $^{*}$ | **36%** $^{*}$ |
| Total Respondents | 426 | 78 | 348 | 141 | 285 |
| No Response | (7) | (2) | (5) | (1) | (6) |

$^{*}p < .05$;

**Table 8.** "Receipts from online merchants"

| | ALL | Europe | US | Savvy | Green |
|---|---|---|---|---|---|
| Does not need special protection | 25% | 29% | 25% | 26% | 25% |
| Should be *digitally-signed* | 25% | **39%** $^{**}$ | **22%** $^{**}$ | **33%** $^{*}$ | **21%** $^{*}$ |
| Should be *sealed* with encryption | 13% | **6%** $^{*}$ | **15%** $^{*}$ | 12% | 14% |
| Should be *both* signed and sealed | 34% | **23%** $^{*}$ | **36%** $^{*}$ | **27%** $^{*}$ | **37%** $^{*}$ |
| Should never be sent by email | 3% | 3% | 3% | 2% | 3% |
| *sealed* or *both* | 47% | **30%** $^{***}$ | **51%** $^{***}$ | **39%** $^{*}$ | **51%** $^{*}$ |
| *digitally-signed* or *both* | 59% | 62% | 58% | 60% | 58% |
| Total Respondents | 425 | 77 | 348 | 141 | 284 |
| No Response | (8) | (3) | (5) | (1) | (7) |

$^{*}p < .05$; $^{**}p < .01$; $^{***}p < .001$;

**Financial Communications (Table 11).** Not surprisingly, a majority (62.7%) of our respondents thought that financial statements should be both signed and sealed. There was no significant difference in response rates to this question between any of our groups. Similar response rates were seen for official mail sent to government agencies.

**Table 9.** "Personal email sent or received at work"

| | ALL | Europe | US | Savvy | Green |
|---|---|---|---|---|---|
| Does not need special protection | 35% | **47%** $^*$ | **33%** $^*$ | 40% | 33% |
| Should be *digitally-signed* | 17% | 18% | 17% | 21% | 15% |
| Should be *sealed* with encryption | 15% | 17% | 14% | **9%** $^{**}$ | **18%** $^{**}$ |
| Should be *both* signed and sealed | 23% | **14%** $^*$ | **25%** $^*$ | 18% | 26% |
| Should never be sent by email | 10% | **4%** $^*$ | **11%** $^*$ | 13% | 8% |
| *sealed* or *both* | 38% | 31% | 39% | **26%** $^{***}$ | **44%** $^{***}$ |
| *digitally-signed* or *both* | 40% | 32% | 42% | 38% | 41% |
| Total Respondents | 425 | 77 | 348 | 141 | 284 |
| No Response | (8) | (3) | (5) | (1) | (7) |

$^*p < .05$; $^{**}p < .01$; $^{***}p < .001$;

**Table 10.** "Personal email sent or received at home:"

| | ALL | Europe | US | Savvy | Green |
|---|---|---|---|---|---|
| Does not need special protection | 51% | 58% | 49% | 53% | 49% |
| Should be *digitally-signed* | 18% | 16% | 18% | 22% | 16% |
| Should be *sealed* with encryption | 9% | 9% | 9% | 9% | 9% |
| Should be *both* signed and sealed | 23% | 17% | 24% | **17%** $^*$ | **25%** $^*$ |
| Should never be sent by email | 0% | 0% | 0% | 0% | 0% |
| *sealed* or *both* | 31% | 26% | 33% | **25%** $^*$ | **34%** $^*$ |
| *digitally-signed* or *both* | 40% | 32% | 42% | 38% | 41% |
| Total Respondents | 426 | 77 | 349 | 139 | 287 |
| No Response | (7) | (3) | (4) | (3) | (4) |

$^*p < .05$;

**Communication with Politicians (Table 12).** Unlike mail on official business, respondents felt that neither newsletters from politicians nor mail to political leaders required any kind of special protection. Once again this is somewhat surprising, given that such communications are easily spoofed either to discredit a politician or to mislead leaders about the depth of public support on a particular issue.

There was no statistically-significant difference between the way that any of our groups answered this question, so individual breakdowns by group are not provided.

### 3.3 Opinions of Companies That Send Digitally-Signed Mail (Table 13)

When queried on a scale of 1 to 5, where 1 was "Strongly Agree" and 5 was "Strongly Disagree," respondents on average slightly agreed with the statement that companies sending digitally-signed mail "Are more likely to have good return policies." Respondents also slightly agreed with the statement that such companies "Are more likely to be law-abiding." No significant difference was seen between any of our groups for these two questions.

We were curious as to whether or not interest in cryptography was seen as an American technology, so we asked respondents whether or not they thought that companies

**Table 11.** Financial Communications: What Kind of Protection is Necessary?

| | "A bank or credit-card statement:" | "Mail to government agencies on official business, such as filing your tax return or filing complaints with regulators:" |
|---|---:|---:|
| Does not need special protection | 1.2% | 4.2% |
| Should be *digitally-signed* | 2.1% | 9.2% |
| Should be *sealed* with encryption | 16.2% | 9.9% |
| Should be *both* signed and sealed | 62.7% | 64.6% |
| Should never be sent by email | 17.8% | 12.2% |
| *sealed* or *both* | 78.9% | 74.4% |
| *digitally-signed* or *both* | 64.8% | 73.7% |
| Total Respondents | 426 | 426 |
| No Response | (7) | (7) |

**Table 12.** Communication to and from Political Leaders: What Kind of Protection is Necessary?

| | "Newsletters from politicians:" | "Mail to political leaders voicing your opinion on a matter:" |
|---|---:|---:|
| Does not need special protection | 54.9% | 52.5% |
| Should be *digitally-signed* | 19.7% | 27.2% |
| Should be *sealed* with encryption | 0.5% | 4.2% |
| Should be *both* signed and sealed | 2.1% | 10.3% |
| Should never be sent by email | 22.8% | 5.9% |
| *sealed* or *both* | 2.6% | 14.5% |
| *digitally-signed* or *both* | 21.8% | 37.5% |
| Total Respondents | 426 | 427 |
| No Response | (7) | (6) |

sending digitally-signed mail "Are more likely to be based in the United States." Interestingly enough, this *did* have statistically-significant variation between our various groups. The *Europe* and *Savvy* groups disagreed with this statement somewhat, while the *US* and *Green* groups agreed with the statement somewhat.

When asked whether or not a digitally-signed message "is more likely to contain information that is truthful," respondents neither agreed nor disagreed, with no significant difference between our four groups.

All groups disagreed somewhat with the statement that digitally-signed mail "is less likely to be read by others," although respondents in the *Europe* group disagreed with the statement significantly more than the *US* group.

**Table 13.** Do you *strongly agree (1)* or *strongly disagree (5)* with the following statements?"

Companies that send digitally-signed mail

| Question | Group | $\overline{x}$ | $n$ | $\sigma$ |
|---|---|---|---|---|
| "Are more likely to have good return policies" | ALL | 3.0 | 412 | 1.07 |
| "Are more likely to be law-abiding" | ALL | 2.8 | 412 | 1.17 |
| "Are more likely to be based in the United States" | **Europe** | **3.5** | 77 | 1.28 |
| | **US** | **3.0** | 334 | 1.13 |
| | **Savvy** | **3.3** | 135 | 1.26 |
| | **Green** | **3.0** | 276 | 1.12 |

Digitally-signed mail:

| Question | Group | $\overline{x}$ | $n$ | $\sigma$ |
|---|---|---|---|---|
| "Is more likely to contain information that is truthful," | ALL | 3.0 | 411 | 1.20 |
| "Is less likely to be read by others," | **Europe** | **3.7** | 77 | 1.25 |
| | **US** | **3.2** | 335 | 1.22 |

### 3.4 Free-Format Responses

Our survey contained many places where respondents could give free-format responses. Many wrote that they wished they knew more about email security. For example:

> *I wish I knew more about digitally-signed and sealed encrypted e-mail, and I wish information were more generally available and presented in a manner that is clear to those who aren't computer scientists or engineers.*
>
> *This is an interesting topic... I had not thought about the need to send/receive signed or sealed e-mail for other than tax info.*

Others do not understand cryptography and do not want to learn:

> *Most sellers do not care about digital signatures when selling on on-line marketplaces unless they are dealing in big sums of money in the transaction, even then I still do not care.*
>
> *I think it's a good idea, but I'm lazy and it's too much trouble to bother with.*

These comments, and many others, reinforce our belief that the usability standards for a successfully-deployed email security system must be extraordinarily high. It is not enough for systems to be easily learned or used, as Whitten argues. [13] Instead, we believe that normal use of security systems must require zero training and zero keystrokes. Security information should be conveyed passively, providing more information on demand, but should not otherwise impact on standard operations.

Many respondents used the free-format response sections to complain about spam, viruses, and phishing — sometimes to the point of chastising us for not working on these problems:

> *I hope this [survey] will help to stop the viruses, spam, spyware and hijackers all too prevalent on the web.*

> *[I] feel the topic is somehow "phony" because of the way viruses are transmitted by email. I'm more concerned with attacks by future NIMDAs*[2] *than I am with sending or receiving signed email.*

Several respondents noted that there is little need to send sealed email, since such messages can be sent securely using feedback forms on SSL-encrypted websites.

## 4 Conclusions and Policy Implications

We surveyed hundreds of people actively involved in the business of e-commerce as to their views on and experience with digitally-signed email. Although they had not received prior notification of the fact, some of these individuals had been receiving digitally-signed email for more than a year. To the best of our knowledge this is the first survey of its kind.

It is widely believed that people will not use cryptographic techniques to protect email unless it is extraordinarily easy to use. We showed that even relatively unsophisticated computer users who do not send digitally-signed mail nevertheless believe that it should be used to protect the email that they themselves are sending (and to a lesser extent, receiving as well).

We believe that digitally-signed mail could provide some measure of defense against phishing attacks. Because attackers may try to obtain certificates for typo or copycat names, we suggest that email clients should indicate the difference between a certificate that had been received many times and one that is being received for the first time—much in the way that programs implementing the popular SSH protocol [15] alert users when a host key has changed.

We found that the majority (58.5%) of respondents did not know whether or not the program that they used to read their mail handled encryption, even though the vast majority (81.1%) use such mail clients. Given this case, companies that survey their customers as to whether or not the customers have encryption-capable mail readers are likely to yield erroneous results.

We learned that digitally-signed mail tends to increase the recipient's trust in the email infrastructure. We learned that despite more than a decade of confusion over multiple standards for secure email, there are now few if any usability barriers to receiving mail that's digitally-signed with S/MIME signatures using established CAs.

Finally, we found that people with no obvious interest in selling or otherwise promoting cryptographic technology believe that many email messages sent today without protection should be either digitally-signed, sealed with encryption, or both.

The complete survey text with simple tabulations of every question and all respondent comments for which permission was given to quote is at `http://www.simson.net/smime-survey.html`.

[2] W32/Nimda was an email worm that was released in September 2001 and affected large parts of the Internet. [14]

### 4.1 Recommendations

We believe that financial organizations, retailers, and other entities doing business on the Internet should immediately adopt the practice of digitally-signing their mail to customers with S/MIME signatures using a certificate signed by a widely-published CA such as VeriSign. Software for processing such messages is widely deployed. As one of our respondents who identified himself as "a very sophisticated computer user" wrote:

> *I use PGP, but in the several years since I have installed it I have never used it for encrypting email, or sending signed email. I have received and verified signed email from my ISP. I have never received signed email from any other source (including banks, paypal, etc, which are the organisations I would have thought would have gained most from its use).*

Given that support for S/MIME signatures is now widely deployed, we also believe that existing mail clients and webmail systems that do not recognize S/MIME-signed mail should be modified to do so. Our research shows that there is significant value for users in being able to verify signatures on signed email, even without the ability to respond to these messages with mail that is signed or sealed.

We also believe that existing systems should be more lenient with mail that is digitally-signed but which fails some sort of security check. For example, Microsoft Outlook and Outlook Express give a warning if a message is signed with a certificate that has expired, or if a certificate is signed by a CA that is not trusted. We believe that such warnings only confuse most users; more useful would be a warning that indicates when there is a change in the distinguished name of a correspondent—or even when the sender's signing key changes—indicating a possible phishing attack.

### 4.2 Future Work

Given the importance of email security, a survey such as this one should be repeated with a larger sample and a refined set of questions.[3] It would also be useful to show respondents screen shots of email that was digitally-signed but which failed to verify (for example, because the message contents had been altered or because the CA was created by hackers for a phishing website) and ask what they would do upon receiving such a message. Organizations interested in sending digitally-signed mail may wish to consider before-and-after surveys to gauge the impact of the mail signing on those receiving the messages.

## Acknowledgements

The idea for survey was originally suggested by Jean Camp when she was at Harvard's Kennedy School. Sean Smith at Dartmouth College, John Linn at RSA Security provided useful comments. We are thankful to David C. Clark, Karen Solins and Min Wu for their initial comments on our survey questions, and to Steven Bauer, David C. Clark,

[3] In particular, no questions were asked on the subject of medical privacy.

David Krikorian and Sian Gramates for their comments on the final paper. John-Paul Ferguson at MIT and Elaine Newton at CMU provided useful comments on our statistical methods. Amazon.com's computer security group and nearly 200 other Amazon employees graciously pretested an earlier version of this survey. Apart from allowing its employees to participate in the study, Amazon.com did not contribute monetarily to this study and does not necessarily endorse the conclusions and recommendations herein.

## References

1. Gutmann, P.: Why isn't the internet secure yet, dammit. In: AusCERT Asia Pacific Information Technology Security Conference 2004; Computer Security: Are we there yet? (2004) http://conference.auscert.org.au/conf2004/.
2. Federal Trade Comission: Identity thief goes phishing for consumers' credit information (2003) http://www.ftc.gov/opa/2003/07/phishing.htm.
3. Whitten, A., Tygar, J.D.: Why Johnny can't encrypt: A usability evaluation of PGP 5.0. In: 8th USENIX Security Symposium. (1999) 169 184
4. Linn, J.: RFC 989: Privacy enhancement for Internet electronic mail: Part I: Message encipherment and authentication procedures (1987) Obsoleted by RFC1040, RFC1113 [5, 6]. Status: UNKNOWN.
5. Linn, J.: RFC 1040: Privacy enhancement for Internet electronic mail: Part I: Message encipherment and authentication procedures (1988) Obsoleted by RFC1113 [6]. Obsoletes RFC0989 [4]. Status: UNKNOWN.
6. Linn, J.: RFC 1113: Privacy enhancement for Internet electronic mail: Part I message encipherment and authentication procedures (1989) Obsoleted by RFC1421 [16]. Obsoletes RFC0989, RFC1040 [4, 5]. Status: HISTORIC.
7. Zimmermann, P.R.: The Official PGP User's Guide. MIT Press (1995)
8. Atkins, D., Stallings, W., Zimmermann, P.: RFC 1991: PGP message exchange formats (1996) Status: INFORMATIONAL.
9. Elkins, M.: RFC 2015: MIME security with pretty good privacy (PGP) (1996) Status: PROPOSED STANDARD.
10. Dusse, S., Hoffman, P., Ramsdell, B., Lundblade, L., Repka, L.: RFC 2311: S/MIME version 2 message specification (1998) Status: INFORMATIONAL.
11. Ramsdell, B.: Secure/multipurpose internet mail extensions (s/mime) version 3.1 message specification (2004)
12. GVU: GVU's tenth WWW user survey results (1999) http://www.cc.gatech.edu/gvu/user surveys/survey-1998-10/.
13. Whitten, A.: Making Security Usable. PhD thesis, School of Computer Science, Carnegie Mellon University (2004)
14. CERT Coordination Center: CERT advisory ca-2001-26 nimda worm. Technical report, CERT Coordination Center, Pittsburgh, PA (2001)
15. T. Ylonen, e.a.: SSH protocol architecture (1998) Work in Progress.
16. Linn, J.: RFC 1421: Privacy enhancement for Internet electronic mail: Part I: Message encryption and authentication procedures (1993) Obsoletes RFC1113 [6]. Status: PROPOSED STANDARD.

# Securing Sensitive Data with the Ingrian DataSecure Platform

Andrew Koyfman

Ingrian Networks, Redwood City, CA 94122, USA
akoyfman@ingrian.com
http://www.ingrian.com/

**Abstract.** Recent high profile data thefts have shown that perimeter defenses are not sufficient to secure important customer data. The damage caused by these thefts can be disastrous, and today an enterprise with poor data security may also find itself violating privacy legislation and be liable to civil lawsuits. The Ingrian DataSecure Platform presents an approach for protecting data inside the enterprise – and so to help eliminate many of the threats of data theft.

This paper demonstrates how an enterprise can prevent unauthorized data exposure by implementing column encryption in commercially available databases. Adding security at the database layer allows an enterprise to protect sensitive data without rewriting associated applications. Furthermore, large enterprises require scalable and easily administrable solutions. In order to satisfy this demand this paper introduces the concept of a Network-Attached Encryption Server, a central device with secure storage and extensive user access permissions for protecting persistent security credentials.

## 1 Introduction

Consumers and on-line shoppers have been educated to look for the little lock icon at the bottom of their browser to verify that they are connected securely to the website of their choice. The icon tells the consumer that he is using SSL and that the data in transit will be protected from eavesdroppers. While some potential weaknesses exist in the SSL protocol [1] the amount of time that is needed to capture the data and break the cryptography surrounding the communication far outweighs the sensitivity of the data that is being protected [2]. Attackers that want to acquire personal information have a far easier target – the database where these records are stored.

Getting access to the database allows attackers to gather millions of records of identifying information in one security breach. While enterprises worldwide spend $42 billion per year on IT security, [3] expensive breaches continue to occur. Attackers use a variety of ingenious methods to penetrate firewalls and other perimeter security defenses. Many attackers exploit unpatched systems or tunnel through existing applications and are thereby undetected by perimeter defense systems set up by the IT department. Further, most estimates cite that over 50% of the breaches are perpetrated by those inside the organization [4].

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 203 – 210, 2005.
© Springer-Verlag Berlin Heidelberg 2005

In all these security breaches, after penetrating the perimeter defenses the attacker can access many corporate machines virtually unchallenged. If the database server is poorly protected, then the attacker will obtain information stored in customer records. Even access to the media containing raw database files is often sufficient for an attacker to gain the information he is seeking.

### 1.1 Damage Potential from Data Theft

An important question to ask is what damage can attackers cause by getting access to customer information, and how much effort should be spent on defending a system against these information gathering attacks? The answer, of course, depends on the nature of the company's business and the type of information stored. For a financial institution the consequences may be severe.

First, poor security may subject the company to a number of civil lawsuits. Consider the recent case of mass identity theft at BJ's Warehouse. After the theft, financial institutions had to reissue thousands of customers' credit cards as a precaution. For example, Sovereign Bank had to spend $1mil to reissue credit cards [5]. The affected companies are considering a civil lawsuit against BJ's to recoup their costs. Second, the company's brand name may suffer as a result of disclosing the security breaches, especially if the company is a financial institution, or if the breaches are repetitive. Often disclosure is mandated by laws such as The California Privacy Law, SB-1386. Of course, the process of notification is expensive in itself.

Finally, the customer data itself is very valuable. It contains email addresses, demographics, and spending habits; information that is valuable to advertisers and competitors. Attackers can use this information for sending spam, as was the case in a data theft at Acxiom Corp [5], or they could disclose it to the company's competitors. The precise estimate of damage may be difficult to calculate, but the resulting loss of competitive advantage is obvious.

Each of these examples serves as a legitimate argument for protecting sensitive information such as customer records. Since perimeter security by itself is not enough to prevent data theft, the question arises of how best to limit the damage from possible security breaches.

## 2 Securing Customer Data in Databases

There are a number of approaches are available to organizations that can help them to protect customer data [6]. Perhaps the best and most secure solution is to write security-aware applications that encrypt and decrypt data prior to storing it and after retrieving it from back-end storage. However, it is expensive, if not impossible, to change existing, legacy applications to follow this model. Our system implements an alternative solution to perform encryption at the database level. The algorithm for enabling database encryption is well understood. We built tools that automate this process. Here, we will briefly mention the operations performed.

Consider a sample table containing the customer name, credit card number, order number, and the transaction date. (Fig. 1) In this table only the credit card number needs to be encrypted to prevent unauthorized access, prevent theft, and comply with privacy regulations.

| NAME | CREDIT_CARD | ORDER_NUM | DATE |
|---|---|---|---|
| Jorge Chang | 1234-5678-9012-2345 | 12345 | 8/25/04 |
| Bruce Sandell | 2234-5678-9012-2312 | 67890 | 2/29/04 |
| ... … | … | … | … |

**Fig. 1.** A sample table CUSTOMER

Changing the database tables to encrypt data can be an error-prone and time consuming task if done by hand. The DataSecure Platform provides tools for DBAs to migrate existing data automatically. The tools follow a two step process, which is briefly discussed below. In Section 3, we will consider additional options for further restricting access to the encrypted data.

## 2.1 Creation of Encrypted Tables

During column encryption our tools first create a new table, CUSTOMER_ENC, and populate it with the data from the CUSTOMER table. Once the new table is created, it is desirable to transfer all data out of the CREDIT_CARD column in the CUSTOMER_ENC table to avoid any data type problems or other conversion issues. This is accomplished by creating a temporary table and populating it with the existing credit card column.

Next we adjust the data type in the CREDIT_CARD column to allow for storage of encrypted (binary) data, and to increase the size of the field if necessary. Finally, the CREDIT_CARD data is encrypted and re-imported into the table. (Fig. 2) The original table CUSTOMER is no longer needed and is deleted.

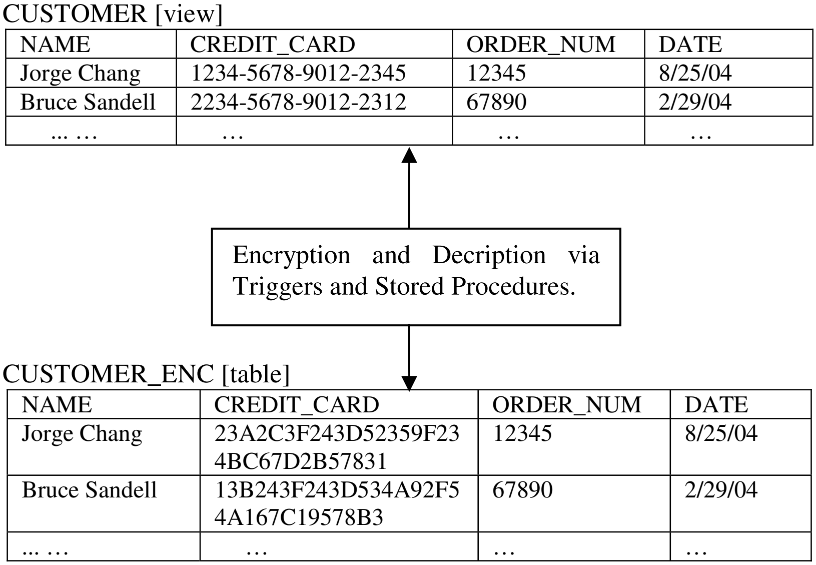
| NAME | CREDIT_CARD | ORDER_NUM | DATE |
|---|---|---|---|
| Jorge Chang | 23A2C3F243D52359F234BC67D2B57831 | 12345 | 8/25/04 |
| Bruce Sandell | 13B243F243D534A92F54A167C19578B3 | 67890 | 2/29/04 |
| ... … | … | … | … |

**Fig. 2.** A sample table CUSTOMER_ENC

## 2.2 Creation of Triggers and Views

Of course, once the CUSTOMER table is deleted, all existing applications that were using this table will break. In order to allow the use of the current table, we create a new view that has the same name as the original table, CUSTOMER. The new view has triggers associated with SELECT, INSERT, and UPDATE operations. These triggers execute stored procedures that encrypt and decrypt data as necessary. In this way, the existing application can still reference the same table name, and perform the same operations as before, while the data itself is now encrypted. (Fig. 3)

As one can see, the basic procedure for encrypting tables is fairly simple, even if some applications may require preparatory work to untangle complex inter-table relationships. A greater challenge lies in the proper storage and management of the secret keys used to encrypt the data.



CUSTOMER [view]

| NAME | CREDIT_CARD | ORDER_NUM | DATE |
|---|---|---|---|
| Jorge Chang | 1234-5678-9012-2345 | 12345 | 8/25/04 |
| Bruce Sandell | 2234-5678-9012-2312 | 67890 | 2/29/04 |
| ... … | … | … | … |

Encryption and Decription via Triggers and Stored Procedures.

CUSTOMER_ENC [table]

| NAME | CREDIT_CARD | ORDER_NUM | DATE |
|---|---|---|---|
| Jorge Chang | 23A2C3F243D52359F23 4BC67D2B57831 | 12345 | 8/25/04 |
| Bruce Sandell | 13B243F243D534A92F5 4A167C19578B3 | 67890 | 2/29/04 |
| ... … | … | … | … |

**Fig. 3.** Relationship between CUSTOMER view and CUSTOMER_ENC table

# 3 Key Management

The question of how to best store and distribute keys used for cryptography is a difficult one. This problem is particularly challenging in the case of database security, especially when the database runs on a cluster of machines. Usually, customers require that each machine in the cluster be able to perform the encryption operations on behalf of a user. Furthermore, only authorized users should have access to the encryption key. Finally, each machine should be able to perform cryptographic operations after an unattended restart, when the system administrator cannot type in the password.

## 3.1 Network-Attached Encryption (NAE) Device

The DataSecure Platform relies on a Network-Attached Encryption (NAE) server to handle all encryption operations associated with long-term data storage. (Fig. 4) The

NAE server can store all cryptographic keys used throughout the entire enterprise. Databases and other applications make calls to the NAE server and request it to perform an encryption or a decryption on their behalf. The NAE server verifies that the client has the permissions necessary to perform the operation and sends the result back to the client. The client only knows the key name, but not the actual key used for the operation. Thus, the key never leaves the NAE server and cannot be stored on insecure machines. This isolates all the keys' security to a single location which can be protected.

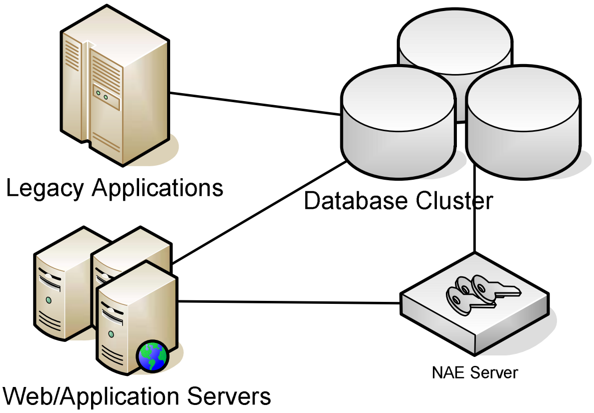


**Fig. 4.** NAE device in an enterprise setting. Legacy applications communicate with the database which in turn requests cryptographic operations from the NAE device. Newer web applications can access the database, or they can access the NAE device directly

### 3.2. User Access Permissions

In order to perform cryptographic operations, a client needs to authenticate itself to the NAE server. Using a username and password is still the most common way to perform this authentication. It is possible that some of these passwords are stored insecurely. For instance, developers may store them in plain-text in order to perform an unattended restart of the web server.

The NAE server can help to address this problem by restricting key use to authorized clients only. To accomplish this, the server maintains some meta-data along with the actual key material. The meta-data contains an extensive access permissions policy for potential key users. Some users may have full access to the key, while others may only be limited to either encryption or decryption operations. Additionally, the rate at which users can perform operations and the time of day when these operations may occur can be restricted in order to mitigate potential risks.

For example, consider an enterprise that has four different classes of users that need to access the CUSTOMER table from Figure 3. First a sample web application that needs to support unattended restarts (for example a checkout program), will populate the table with a new credit card number received from the customer. This application would have the permissions to perform encryption operations, but would not be able to decrypt the CREDIT_CARD column. Therefore even if the password used by the checkout program was stolen, the thief would not be able to decrypt any of the data in the database.

A separate application, for instance one that performs batch credit card billing, will need to be able to decrypt the CREDIT_CARD column, but does not need to add new rows to the table. In order to reduce resource consumption during the day, batching operations are performed at night. Using the principle of least privilege, this application is given decrypt permissions only and is restricted to using its key between 2am and 4am.

On occasion users may call in with questions about their order. In order to help them, a customer service representative will need to access the CUSTOMER table. However, service representatives do not need to access the customer's credit card data. Therefore, they do not have permissions to perform encryption or decryption operations.

However, sometimes a customer is not satisfied with the order and demands a refund. In our example, it is the company's policy to have managers approve all refunds so, the customer service manager needs to have access to the customer's credit card information. The manager is human; hence it would be suspicious if he was issuing hundreds of refunds per second or working outside the regular business hours. Therefore, a service manager is restricted to making only one encryption or decryption per minute, and only during the day.

This separation of privileges limits the damage that may be caused by an attacker. The only way for an attacker to gain access to a large quantity of credit card numbers is to compromise the batch processing application. An enterprise would need to take sufficient precautions to make this infeasible.

**Table 1.** Key Policy Permissions

| User | Can Encrypt | Can Decrypt | Rate | Time of Day |
|---|---|---|---|---|
| Checkout | y | n | Unl. | Unl. |
| Billing | n | y | Unl. | 2AM – 4 AM |
| Service Rep. | n | n | NA | NA |
| Service Manager. | y | y | 1/min | 9AM - 5AM Mon. - Fri. |

### 3.3 Disaster Recovery

Although not central to the security of the system, there is an additional benefit to centralizing key storage. Backup and disaster recovery is much easier when all of cryptographic information is stored in one place. An administrator no longer needs to worry about backing up different sets of keys that are scattered on machines throughout the enterprise. Instead, he only needs to back up the NAE server. Of course, these backup sets require additional protection to prevent them from falling into the wrong hands. The Ingrian NAE server encrypts the backups using a password provided by the administrator. Standard physical security practices should also be used to protect the back up media.

## 4 Performance

Improvements in security and manageability come at the expense of performance. Usually, turning on encryption dramatically increases the CPU usage on a machine. In some environments the NAE server model can ameliorate the CPU burden placed on the application or database server by offloading encryption operations. In these cases, the factors that limit performance are the network throughput and latency.

The initial design of the NAE server protocol followed the *init, update, final* model of encryption. The model requires the client to wait for a response from the server after each of the operations. While this works well for encryption of large chunks of data where network latency for the *init* and *final* calls is amortized over the entire operation, we have found that applications using the NAE server typically request encryption of short data chunks. In this case, the network latency has a severe impact on the number of operations a single thread can perform. We found that we can improve performance significantly by making a single *crypt* call to the server.

**Table 2.** Encryption and decryption operations performed per second against a single Ingrian NAE server using TCP to communicate. The server is a dual Penium 3, 1.26GHz cpu with 512Kb cache, 2Gb RAM and a Cavium crypto card (CN 1220-350). The client is a Pentium4HT, 3.0GHz cpu with 1Gb RAM. The AES algorithm uses a 256 bit key. The triple-Des algorithm uses a 168 bit key. The data size is 15 bytes, padded to 16 bytes, per operation

| Encryption Method | Algorithm | 1 thread<br>(ops/sec) | 10 threads<br>(ops/sec) |
|---|---|---:|---:|
| *init update final* | AES/CBC/PKCS5 | 714 | 3,703 |
| | 3DES/CBC/PKCS5 | 666 | 3,703 |
| single *crypt* | AES/CBC/PKCS5 | 1,250 | 9,090 |
| | 3DES/CBC/PKCS5 | 1,333 | 9,302 |
| batch encryption | AES/CBC/PKCS5 | 31,648 | NA |
| | 3DES/CBC/PKCS5. | 31,625 | NA |

The single crypt call performance is sufficient for most applications we have encountered. However, there is one class of applications, batch processing, which requires significantly higher throughput. For these applications we use a batch mode to communicate with the NAE server. The batch mode is optimized for a single thread to send multiple encryption requests without waiting to receive the answer from the server. This increases performance significantly.

The performance numbers in Table 2 were obtained by running a client test program against a single NAE server. NAE servers can be clustered to support a higher number of clients and operations.

## 5 Conclusion

Perimeter security and network security is not sufficient by itself to prevent attackers from stealing customer data records. In order to prevent expensive thefts, enterprises must focus on protecting data in storage as well as data in transit. The Ingrian Data-Secure platform combined with the NAE server provides a set of utilities that can aide administrators in securing enterprise data. The DBA is able to encrypt columns containing sensitive information and a Security Administrator can restrict access to the keys needed to retrieve the data. The NAE server also helps protects data throughout the enterprise by centralizing key storage, and by facilitating backup and recovery. While other methods for securing customer information exist, the benefit of using our approach is that it allows the enterprise to support existing, legacy applications without making expensive modifications.

## References

1. Boneh, D., Brumley, D.: Remote Timing Attacks Are Practical, Proceedings of the 12th Usenix Security Symposium
2. Schneier, B.: CryptoGram Newsletter, March 15, 2003
3. Leyden, J.: Global IT Security Spending Hits $42bn, The Register, http://www.theregister.co.uk/2004/04/30/idc_security_booming/
4. Mogul, R.: Danger Within – Protecting your Company from Internal Security Attacks, Gartner, 2002
5. CBS News: “Big-Time ID Theft”, Aug 13, 2004 http://www.cbsnews.com/stories/2004/08/13/tech/main635888.shtml
6. Mattsson, U.: A Database Encryption Solution That Is Protecting Against External And Internal Threats, And Meeting Regulatory Requirements, HelpNetSecurity, July 2004, http://www.net-security.org/article.php?id=715

# Ciphire Mail
# Email Encryption and Authentication

Lars Eilebrecht

Ciphire Labs
le@ciphirelabs.com

**Abstract.** Ciphire Mail is cryptographic software that provides email encryption and digital signatures. The Ciphire Mail client resides on the user's computer between the email client and the email server, intercepting, encrypting, decrypting, signing, and authenticating email communication. During normal operation, all operations are performed in the background, making it very easy to use even for non-technical users. Ciphire Mail provides automated secure public-key exchange using an automated fingerprinting system. It uses cryptographic hash values to identify and validate certificates, thus enabling clients to detect malicious modification of certificates. This data is automatically circulated among clients, making it impossible to execute fraud without alerting users. The Ciphire system is a novel concept for making public-key cryptography and key exchange usable for email communication. It is the first transparent email encryption system that allows everyone to secure their communications without a steep learning curve.

**Keywords:** Ciphire, secure email, email encryption, email authentication, digital signatures, certificates, fingerprints, fingerprint system, PKI.

## 1 Introduction

Ciphire Mail is cryptographic software providing email encryption and digital signatures [24]. The Ciphire Mail client resides on the user's computer between the email client (mail user agent, MUA) and the email server (mail transfer agent, MTA), intercepting, encrypting, decrypting, signing, and authenticating email communication. During normal operation, all operations are performed in the background. This makes Ciphire Mail very similar to a transparent proxy. Apart from per-user installations, Ciphire Mail may also be deployed on mail servers as a gateway solution. A combination of client and gateway installations is possible, as well.

Public-key exchange and key agreement [1] are automated and handled via certificates available through a central certificate directory. These services are operated by Ciphire Labs and do not require any local server installations, additional hardware, or additional software.

The Ciphire system provides an automated fingerprint verification system to solve trust issues existing with central certification and directory services. The Ciphire

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 211 – 224, 2005.
© Springer-Verlag Berlin Heidelberg 2005

Fingerprint System allows the users of the Ciphire system to verify the authenticity of certificates and prevents them from being compromised by the provider of the central services.

Ciphire Mail uses only well-known standard cryptographic algorithms including RSA [2], DSA [3], ElGamal [4], Twofish [5], AES [6], or SHA [7] for its cryptographic operations. It uses 2048-bit keys for asymmetric algorithms and 256-bit keys for symmetric algorithms.

## 2 Installation and Integration

### 2.1 Ciphire Mail Client

The Ciphire Mail client consists of three parts: the core client, a graphical configuration interface, and mail connector modules (redirector). Supported email protocols include SMTP [8], POP3 [9], and IMAP4 [10]. The STARTTLS, and direct SSL [11] and TLS [12] variants of these protocols are supported as well.

For the proprietary email systems Microsoft Exchange and Lotus Notes separate connector modules are available that directly integrate with the Outlook and Notes client as a plug-in and automatically handle communication between Ciphire Mail and the email application.

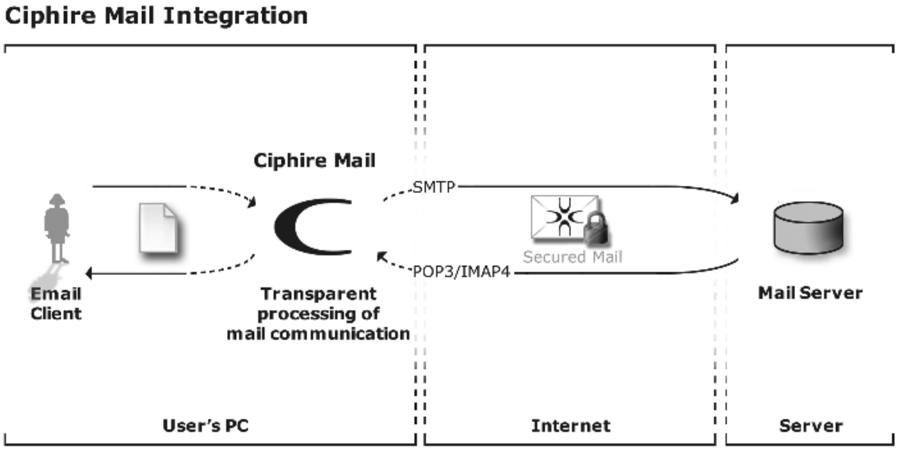


**Fig. 1.** Integration of Ciphire Mail

### 2.2 Ciphire Mail Gateway

The Ciphire Mail client can be run in “server mode” providing a gateway solution. When used in this mode, Ciphire Mail allows creation of single user certificates as well as creation of server certificates. By default, lookups are performed to find the certificate corresponding to the exact email address of the recipient. If the Ciphire

Mail client or gateway finds no certificate for an email address, the lookup will automatically fall back to the domain name level.

# 3 Ciphire Certificates

Ciphire certificates use ASN.1 format [13]. This makes them similar to X.509 certificates [14], with the following exceptions and improvements.

## 3.1 Multiple Public Keys

Each certificate can contain an arbitrary number of public keys. Currently, Ciphire Mail uses three different keys: RSA, DSA, and ElGamal. Each certificate is signed using RSA and DSA and a Ciphire Mail client requires both signatures to be valid in order to deem the certificate as valid. Further, each message is encrypted using RSA and ElGamal (multi-layer encryption). Using always two or more different cryptographic algorithms ensures that a message or certificate will still stay secure, even if a weakness in one of the algorithms is found in the future.

## 3.2 Identity

A Ciphire certificate binds public keys to an email address, host or domain name. No other information about the requestor is included or required. This allows for an automated certification process.

## 3.3 User Controls Certification

To ensure that the user controls creation, renewal and revocation of certificate, each certificate contains self-signatures. This prevents the CA from changing or revoking a certificate without the users consent.

## 3.4 Revocation and Renewal of Certificates

If a certificate is revoked, a dedicated revocation certificate is created. It replaces the old certificate using the same values, e.g., public keys. The renewal of a certificate involves the creation of a new set of public keys and is a combination of revocation of the old and creation of a new certificate.

## 3.5 Certificate Chaining

The renewal of certificate creates a cryptographic link from the old certificate to the new certificate. The revocation certificate includes the certificate ID of the new certificate and the new certificate includes the certificate ID of the revocation certificate. In addition, the revocation certificate contains a »successor signatures« created with the new keys. After a certificate has been renewed multiple times, a certificate chain is created that can be checked by the Ciphire Mail client.

# 4 Certification

Certification is an automated process invoked by a Ciphire Mail client when the user creates a certificate for a specific email address (or fully-qualified domain name). To verify the existence of the given address and to verify that the owner of the address owns the private keys corresponding to the public key, the Ciphire CA uses a mail-based challenge/response mechanism.

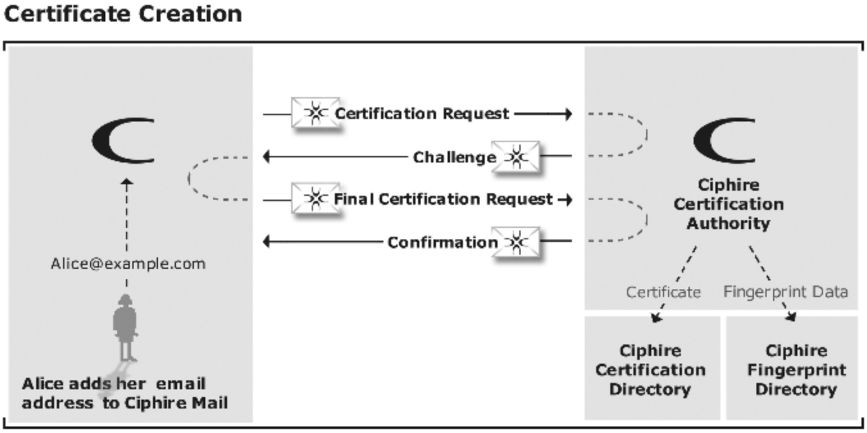

Certificate Creation
Certification Request
Challenge
Final Certification Request
Confirmation
Alice@example.com
Alice adds her email address to Ciphire Mail
Ciphire Certification Authority
Certificate
Fingerprint Data
Ciphire Certification Directory
Ciphire Fingerprint Directory

**Fig. 2.** Automatic processing of certification requests

If all criteria for a particular certification request have been met, the Ciphire CA issues the certificate (or revocation certificate) and publishes it in the Ciphire Certificate Directory (CCD). The CA ensures that only one active certificate is available for a specific address at any given time.

# 5 Ciphire Certificate Directory

The CCD contains all certificates issued by the Ciphire CA, including active and revoked certificates. All private keys are of course created by the client and kept on the user's computer. CCD servers are part of a central infrastructure operated by Ciphire Labs. The infrastructure provides redundant services and is distributed over multiple data centers in different locations.

Every client can download certificates from the CCD by looking them up by their email address or their unique certificate ID. Lookups by email address always retrieve the current active certificate, provided one is available for the given address. A lookup by certificate ID return either the current active certificate or a revocation certificate.

All certificate lookups are fully automated and performed by the Ciphire Mail client whenever a certificate and its associated public keys are required to process a certain email message.

All lookups are cached by the client, including negative lookups that do not return a certificate. The default cache time is 36 hours, but users may configure their clients to cache lookup responses from only a few hours up to several weeks. If a Ciphire Mail client receives an email that is signed or authenticated with a new certificate, the cached copy of the certificate is automatically updated with the new certificate. Further, a user may force a remote lookup on a per-message base.

## 5.1 Secure Communication

The CCD is not accessed directly by Ciphire Mail clients. Instead, multiple front-end proxies are available that provide access to the CCD and other services, such as the software update, fingerprint, and time service. The core proxies are provided by Ciphire Labs, but third-party organizations are also running public Ciphire proxies. Further, organizations and Internet service provider can run a local Ciphire proxy to optimize bandwidth consumption if a large number of Ciphire users have to be served.

Communication with a proxy server is encrypted and all responses from the Ciphire services (e.g., CCD) are signed. Further, the signed response also includes the original lookup argument from the client. This ensures, that the client is able to authenticate the response and verify, that the response corresponds to his original lookup. Therefore, the proxy, or proxies, cannot change the content of the response, e.g., to return a wrong certificate.

## 5.2 Traffic Analysis?

A valid question regarding the CCD is: Can the provider of the CCD do traffic analysis, i.e., see who is communicating with whom?

In order to access a proxy the Ciphire Mail client has to log-on to the proxy that requires authentication with a valid Ciphire certificate. Therefore communication with a Ciphire proxy is not anonymous. This potentially allows traffic analysis on the CCD or Ciphire proxy. This kind of traffic analysis is always possible for the user's email or Internet Service Provider (ISP). However, the Ciphire system tries to minimize this risk using the following mechanisms:

- Encrypted Communication: First of all, to prevent that an external observer is able to do traffic analysis based on certificate lookups, all communication with a proxy is encrypted.
- Lookup Cache: As describe above, every Ciphire client uses a lookup cache. If a cached copy is available, the Ciphire client will not send a lookup to the proxy, until the cached response expires.
- Hashed Lookup Arguments: Lookup arguments, such as email addresses, are not included as plaintext values in the lookup. Instead a hash is calculated and used as argument. Only if the lookup yields a certificate will the proxy be able to determine the email address or certificate information of interest. Otherwise, the proxy is not able to derive any useful information from the lookup arguments.
- Daily Logons: A client does not authenticate itself for every lookup. This is only done once a day and each client is assigned a random session token during authentication. Only the session token is used to encrypt communication with a proxy.

This makes it very cumbersome for the proxy to correlate lookups to the email address of a requestor.
- Primary Certificate: The certificate (i.e., corresponding private key) used for the proxy logon is not necessarily the certificate for the account that is being used as the sender of an email message. If a user has added multiple email address to Ciphire Mail, the user can choose the account that will be used to log-on to a proxy.
- Web Proxy: If a user is concerned about his IP address being known to a Ciphire proxy, the user may use a normal web proxy to communicate with a Ciphire proxy.

To prevent that the provider of the core proxies or CCD can do any kind of traffic analysis, a user may use one or more third-party proxies, i.e., proxies operated by a different organization. A lookup from a client is only forwarded by the client, but apart from the lookup argument, it does not contain any information about the client. The proxy itself logs on to an upstream proxy or one of the core proxies.

## 6 Trusted Certification and Directory Services

In many public-key cryptography solutions the user is required to blindly trust a third-party, like a classical certification authority (CA), that the issued certificate is still valid and has not been tampered with. Other systems, like OpenPGP-based systems [15], require the user to perform manual verifications of an owner's identity and integrity of a public key to find out if it is valid or not.

In the Ciphire system a user is not required to perform manual verifications and most importantly he is not required to blindly trust the Ciphire CA [25].

### 6.1 Concept

To achieve this, the Ciphire system uses, in addition to the usual CA certification, an automated fingerprinting system that provides the following:

- Verification, if a certificate for a particular user (email address) has been issued by the CA (non-repudiation of certificate issuance)
- Verification, that a certificate has not been modified after it has been issued by the CA (proof of certificate integrity)

This is achieved by the Ciphire Fingerprint System using hash-chaining techniques [16] to create a trusted log of all certification actions the Ciphire CA has performed. It makes sure, that old entries in the log cannot be changed at a later time without invalidating newer entries.

These fingerprint data is made available to all Ciphire Mail clients and used by the clients to automatically authenticate certificates. To ensure that every client has the same fingerprint data as any other client, the most current log entry (summary hash) is exchanged with other clients. When the user sends a secure email message to another Ciphire user, the client automatically includes the summary hash in the encrypted email message. The receiving client extracts the hash and compares it with the corresponding hash in its local copy of the fingerprint data. If the hash values do not match, either the sending client or the receiving client has wrong fingerprint data. The Ciphire Mail client handles all this processing automatically.

### 6.2 Fingerprint Creation

Fingerprints are created in the following cases:

- Certificate creation: A single fingerprint is created when a new certificate is created and issued.
- Certificate renewal: Two fingerprints are created when a certificate is renewal (one fingerprint for the new certificate and one fingerprint for the revocation certificate).
- Certificate revocation: A single fingerprint is created when a certificate is revoked (including emergency revocation), i.e., when the revocation certificate is created and issued.
- Software Update Package creation: A single fingerprint is created when a new Software Update Package is issued.

Together with information about the creation time of a fingerprint, all generated fingerprints are collected in a fingerprint list.

### 6.3 Fingerprint Format

A fingerprint consists of 3 cryptographic hash values (H) and a meta data field:

- H(AID$_n$): The hash of the certificate's address ID (i.e., the user's identity in the form of an email address or hostname)
- H(CID$_n$): The hash of the certificate's serial ID (SID) and issuer data (this hash is also called the certificate ID or CID)
- H(C$_n$): The hash of the certificate's complete data (C$_n$)
- Meta data: A 2-byte binary field that defines the type of the corresponding certificate (e.g., normal certificate or revocation certificate) and shows if it has been created during certificate creation, renewal, or revocation.

With H being a 256-bit hash function (e.g., SHA$_d$-256), a fingerprint has a total size of 768 bit (98 byte).

### 6.4 Fingerprint Lists

Fingerprints are published in fingerprint lists (FPLs) with information on the creation time of the fingerprints. A fingerprint list is signed by the Ciphire Fingerprint Authority (FPA). But directly downloading all fingerprints in a single list is not feasible for a client, as the amount of data and the bandwidth consumption would be too high. Therefore, the FPA does not create a single list containing all fingerprints, but multiple lists containing a certain part of all fingerprints. Such a list is called “Branch FPL” and belongs to a certain “branch”. All branch FPL are assigned - based on their branch number - to a section and a hash over the branch FPL's contents is added to a so-called »Section FPL«. Finally a hash over each section FPL is added to a so-called “Master FPL”.

Each branch FPL contains fingerprints for a certain time interval, the FPL creation interval. An interval is usually one hour, but it may be defined according to the number of certificates issued by the CA. Finally, there are three levels of FPLs: branch FPLs, section FPLs, and the master FPL.

Fingerprints are collected for a specific interval and for every interval a set of branch, section and master FPLs are created. These are sometimes referred to as "Interval FPLs". The interval time is not fixed, but may be changed from time to time. Common values for the interval time are in the range of 15 minutes up to 120 minutes. For example, with an interval of 60 minutes the interval start time may be 18:00:00 and the interval end time may be 18:59:59.

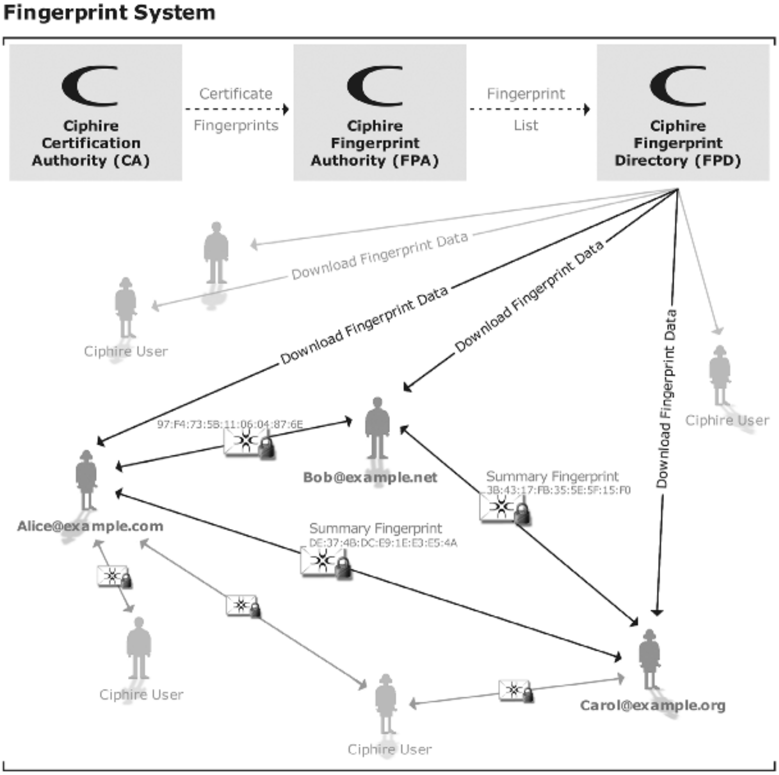

Fingerprint System
Ciphire Certification Authority (CA)
Certificate
Fingerprints
Ciphire Fingerprint Authority (FPA)
Fingerprint
List
Ciphire Fingerprint Directory (FPD)
Download Fingerprint Data
Download Fingerprint Data
Download Fingerprint Data
Download Fingerprint Data
Ciphire User
Ciphire User
97:F4:73:5B:11:06:04:87:6E
Bob@example.net
Summary Fingerprint
3B:43:17:FB:35:5E:5F:15:F0
Alice@example.com
Summary Fingerprint
DE:37:4B:DC:E9:1E:E3:E5:4A
Ciphire User
Carol@example.org
Ciphire User

**Fig. 3.** Flow of fingerprint data in the Ciphire system

All interval FPLs are cryptographically linked by a carry-over hash that provides for a continuous chain of all FPLs. In addition, all interval FPLs are connected by a "Cross FPL".

The cross FPL is a single FPL containing hashes calculated over all master FPL hashes. With each FPL creation interval an additional entry is added to the cross FPL. The cross FPL is a chronological list of hash entries. It keeps track of all certificates ever issued. Each entry corresponds to a time interval, hence to a set of interval FPLs. The main purpose of the cross FPL is to have a global hash that is known to all clients and can be verified by all clients.

This cross FPL hash and the corresponding time stamp are included in each message sent by a Ciphire client to other Ciphire clients. This functionality is called

“Cross-Client Verification”. With this functionality the system ensures that every client has the same FPL data. If not, i.e., if fingerprint verification fails, the user is prominently informed about the mismatch of the fingerprint entry. For example, if a user has fake or wrong FPL data, every Ciphire-secured email the user receives is going to trigger a pop-up informing the user about the security issue.

## 7 Secure Email Communication

When an email client submits a message, the redirector (mail connector module) intercepts the communication and looks up certificates for all recipient email addresses. If no certificate exists for a recipient, the client either sends the email unencrypted, rejects the email, or asks the user what to do, depending on the user's configuration.

If a lookup for an email address in the CCD yields a certificate, the client automatically downloads and validates it by verifying the certificates built-in security properties (e.g., self-signature and issuer signature). In addition, the certificate is verified with the fingerprint system described above. When the certificate is validated, the email is encrypted and sent. All this happens on the fly while the message is being delivered to the email server.

Similar steps are followed when performing decryption, and verification of digital signatures.

### 7.1 Tunneling Email Through Email

Ciphire Mail uses a different message format for encrypted and signed emails. When encrypting an email, the whole email, including its header, is wrapped into a new email. The new email contains only minimal headers required to deliver the message. Information from the original email Subject or CC headers is only part of the encrypted contents that are put in base64-encoded form into the body of the email. The original email is tunneled through email and restored by the recipient's Ciphire Mail client. Email headers that have been added to the email while it was in transit, such as Received headers, are merged into the original email. To ensure the security of the original email, headers may be added, but a header from the original email is never overwritten with a value from the unsecure header.

Some email clients, especially when using IMAP4, download only headers of new email messages, before downloading the complete message. Therefore, Ciphire Mail includes certain email headers, e.g., the Subject, in encrypted form in the header of the encrypted email message. The encrypted data is automatically decrypted to allow these email clients to display this information.

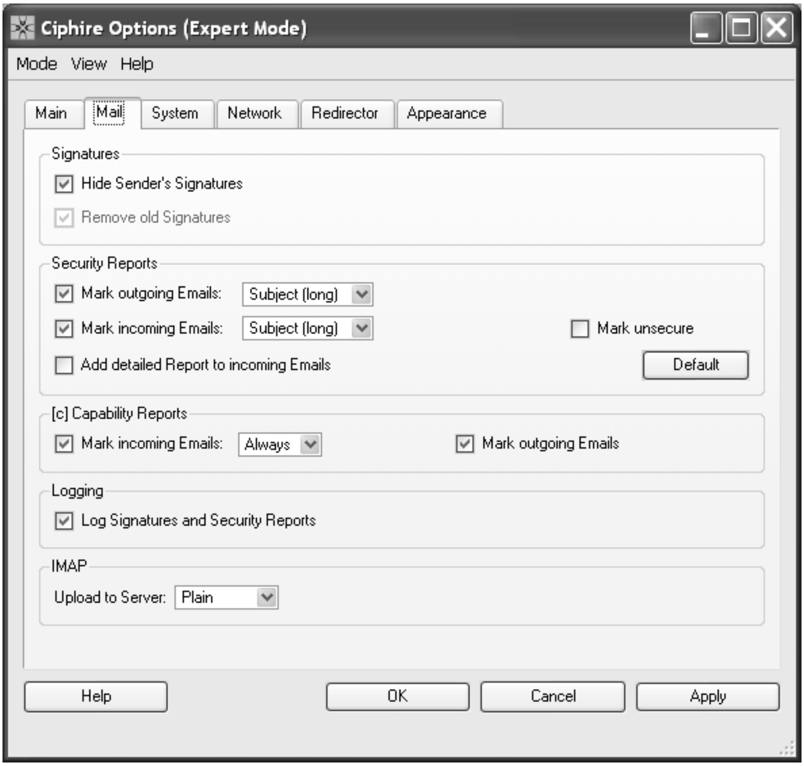
### 7.2 Signing Emails

There are cases where it is desirable to send cleartext-signed email message. The problem with that is, that some mail server and especially content and virus scanner tend to modify or remove certain parts of email messages, e.g., removing the HTML part of an email message. This would break a signature if the signature has been calculated over all parts of the email message. Ciphire Mail signs every MIME [17] part (i.e., attachment) of an email message individually. This ensures that the recipient of the email message can still verify the parts it receives even if a mail server, content or virus scanner removed a certain part from the email message.

Further, a Ciphire signature always includes the email address of the sender and the email addresses of all recipients to protect against surreptitious forwarding attacks [26].

### 7.3 Authentic Emails

Signing emails (non-repudiable authentication) may not always be desirable. To ensure that the recipient of an email is still able to identify the sender of the email message, authentication information about the sender is includes in every Ciphire-encrypted message. When a Ciphire Mail client encrypts a message, the symmetric encryption key used for this, is signed with the sender's private key. In addition to the encryption key, further data like the sender and recipient email address, a timestamp, and protocol-specific data is included in the signature. This provides for repudiable authentication, i.e., deniability of such email messages, but the recipient of the message can be sure that the email of the sender address he is seeing in his email client is the authentic email address of the sender.



Signatures

☑ Hide Sender's Signatures

☑ Remove old Signatures

Security Reports

☑ Mark outgoing Emails: Subject (long)

☑ Mark incoming Emails: Subject (long)

□ Mark unsecure

□ Add detailed Report to incoming Emails

[c] Capability Reports

☑ Mark incoming Emails: Always

☑ Mark outgoing Emails

Logging

☑ Log Signatures and Security Reports

IMAP

Upload to Server: Plain

**Fig. 4.** Mail tab of Ciphire Mail options window (expert mode)

## 7.4 Status of Incoming Emails

Ciphire Mail works almost transparently, but of course it has to show the user the status of incoming email messages, i.e., if they have been received plain text or if the have been encrypted, signed, or both. This is done by putting security reports into the Subject or, optionally, From header. The user can choose between short and long reports.

- `[ciphired]` or `[es]` indicates, that the message was received encrypted and signed.
- `[encrypted]` or `[e]` indicates, that the message was received encrypted, but not signed.
- `[signed]` or `[s]` indicates, that the message was received signed, but unencrypted.
- `[u]` indicates, that the message was unencrypted and unsigned.

In addition to these reports, the user can configure Ciphire Mail to add detailed inline reports to each message.

## 7.5 Controlling Outgoing Emails

Outgoing email is processed based on the user's configuration. By default all emails are encrypted if an active certificate could be found for the recipient and is automatically signed. The user can configure these settings, e.g., configure Ciphire Mail to warn the user if a message cannot be encrypted, or to not sign all outgoing emails by default. These default security strategy settings can be defined for individual recipient, e.g., for an email address, host or domain name.

However, in some cases it may be desirable to define these setting on a per-message base. This is done by putting short tags at into the Subject of outgoing email messages. Ciphire Mail checks outgoing emails for these tags and performs the appropriate action, and removes the tag from the Subject.

- `s!` - sign message
- `n!` - do not sign message
- `e!` - encrypt message (reject message, if encryption is not possible)
- `u!` - do not encrypt message
- `f!` - override local lookup cache

These tags can be combined, e.g., using `un!` would result in an unencrypted and unsigned message being sent.

## 7.6 Single-Point-of-Failure?

The CCD, CA, and related services are provided as highly-available services hosted at multiple locations. Should the CCD still not be available while a Ciphire Mail client is trying to download a certificate, the user is informed about the issue and is asked if he would like to send the message in unencrypted form.

### 7.7 Syncronized Date and Time

The Ciphire Mail client synchronizes its internal time with the Ciphire server, i.e., the Ciphire Time-Stamping Authority (TSA). The Ciphire TSA uses the UTC time zone.

A correct time setting is important to ensure that replay attacks are not possible (e.g., when communicating with a proxy) and that signatures and certification requests from clients contain proper date and time values.

## 8 Application Requirements

Supported operating systems are Windows XP and 2000 (Service Pack 3 or higher), Mac OS X 10.3 (Panther), Linux (Kernel 2.4.0 or higher).

Ciphire Mail supports all email applications using standard SMTP for sending and POP3 or IMAP4 for receiving email (including SSL variants and STARTLS support). Microsoft Exchange and Lotus Notes will be supported in future versions of Ciphire Mail.

## 9 Cryptographic Specifications

Algorithms used in Ciphire-specific cryptographic functions:

- Asymmetric algorithms: RSA, ElGamal, and DSA-2k (DSA-2k is a variation of the normal DSA/DSS algorithm supporting 2048-bit keys [23])
- Key agreement algorithms: (not required)
- Symmetric algorithms: AES, Twofish, and Serpent [21]
- Operation modes and authentication algorithms: CBC-HMAC [19], CCM [20], and CTR
- Hash algorithms: $\text{SHA}_d$-256 and $\text{Whirlpool}_d$-512 [22]
- Pseudo-random number generation algorithm: Fortuna [18] using Twofish in CTR mode
- Supported signing modes: $\text{SHA}_d$-256 with DSA-2k, $\text{SHA}_d$-256 with RSA, and $\text{Whirlpool}_d$-512 with RSA

In addition to this, Ciphire Mail supports SSL and TLS and its associated algorithms. SSL/TLS is not used for Ciphire-specific cryptographic functions, but for supporting mail clients that use SSL/TLS for mail server connections. In such cases, Ciphire Mail proxies SSL/TLS connection between the email client and email server.

## 10 Availability

The Ciphire Mail tool and further information is available on the web site `www.ciphire.com`. Ciphire Mail is free of charge to home-users, non-profit organizations, and the press.

## 11 About Lars Eilebrecht

Lars Eilebrecht is Senior Security Officer at Ciphire Labs, and is involved in the design and development of the Ciphire system. He was awarded a Master of Science degree in computer engineering from the University of Siegen and has 10 years of experience in secure computing. In addition, Lars is co-founder and member of the Apache Software Foundation (ASF). He has authored and co-authored multiple books about web server technologies and is a frequent speaker about open-source, web, and security technologies at IT conferences.

## 12 About Ciphire Labs

Ciphire Labs is a cryptographic research and development facility with offices in Munich, Germany, and Zurich, Switzerland. The company is privately held and produces user-friendly solutions, including Ciphire Mail that enables secure communication over the Internet. Ciphire Labs technologies are peer-reviewed by recognized experts to maximize quality and improve security.

## References

1. W. Diffie, M. E. Hellmann: “New Directions in Cryptography”, IEEE Transactions on Information Theory, 1976.
2. B. Schneier, et al: “Twofish: A 128-bit Block Cipher”, http://www.schneier.com/paper-twofish-paper.pdf, June 1998.
3. B. Kaliski; “PKCS #1: RSA Cryptography Specifications Version 2.0”, RFC 2437, March 1998.
4. NIST: “Digital Signature Standard (DSS)”, FIPS 186-2, January 2000.
5. T. ElGamal: “A public-key cryptosystem and a signature scheme based on discrete logarithms.”, IEEE Transactions on Information Theory, IT-31: 469-472, 1985.
6. NIST: “Advanced Encryption Standard (AES)”, FIPS-192, November 2001.
7. NIST: “Specifications for the Secure Hash Standard”, FIPS 180-2, August 2002.
8. J. Klensin, et al: “Simple Mail Transfer Protocol”, RFC 2821, April 2001.
9. J. Myers, M. Rose: “Post Office Protocol - Version 3”, RFC 1939, May 1996.
10. M. Crispin: “Internet Message Access Protocol - Version 4rev1”, RFC 2060, December 1996.
11. A. Frier, P. Karlton, P. Kocher, “The SSL 3.0 Protocol”, Netscape Communications Corp., November1996.
12. T. Dierks, C. Allen: “The TLS Protocol Version 1.0”, RFC 2246, January 1999.
13. ITU: “Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation”, ITU-T Recommendation X.680 (ISO/IEC 8824-1:2002), 2002.
14. R. Housley, et al: “Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL)”, RFC 3280, April 2002.
15. J. Callas, et al: “OpenPGP Message Format”, RFC 2440, November 1998.
16. S. Haber, W. S. Stornetta: “How to Time-Stamp a Digital Document”, Journal of Cryptography, 1991.

17. N. Freed: “Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies”, RFC 2045, November 1996.
18. N. Ferguson, B. Schneier: “Practical Cryptography”, Wiley, 2003.
19. H. Krawczyk, et al: “HMAC: Keyed-Hashing for Message Authentication”, RFC 2104, February 1997.
20. D. Whitting, R. Housley, N. Ferguson: “Counter with CBC-MAC (CCM)”, http://www.macfergus.com/pub/ccm.html, 2002.
21. R. Anderson, E. Biham, L. Knudson: “The Case for Serpent”, March 2000. http://www.cl.cam.ac.uk/~rja14/serpent.html
22. V. Rijmen, P. S. L. M. Barreto: “The Whirlpool Hash Function”, http://planeta.terra.com.br/informatica/paulobarreto/WhirlpoolPage.html, 2001.
23. L. Eilebrecht: “Ciphire - Technical Product Description”, Ciphire Labs, unpublished.
24. R. Housley, N. Ferguson: “Security Design Review of the Ciphire System”, July 2004
25. B. Schneier: “Analysis of the Ciphire System's Resistance to Insider Attacks”, January 2005.
26. D. Davis: “Defective Sign & Encrypt in S/MIME, PKCS#7, MOSS, PEM, PGP, and XML”, Proceedings USENIX Technical Conference, 2001.

# A User-Friendly Approach to Human Authentication of Messages

Jeff King and Andre dos Santos

College of Computing, Georgia Institute of Technology*,
Atlanta GA 30332, USA
{peff, andre}@cc.gatech.edu

**Abstract.** Users are often forced to trust potentially malicious terminals when trying to interact with a remote secure system. This paper presents an approach for ensuring the integrity and authenticity of messages sent through an untrusted terminal by a user to a remote trusted computing base and vice versa. The approach is both secure and easy to use. It leverages the difficulty computers have in addressing some artificial intelligence problems and therefore requires no complex computation on the part of the user. This paper describes the general form of the approach, analyzes its security and user-friendliness, and describes an example implementation based on rendering a 3-D scene.

**Keywords:** Authentication, Human Cryptography.

## 1 Introduction

Security protocols often require their participants to perform complex computations. While computers are built for such tasks, human users must trust computers to faithfully perform the operations on their behalf. Unfortunately, this trust is often misplaced or misunderstood. The ubiquity of public computers is convenient, but those computers act not in the interests of their users, but of programs running inside them. Even if the owner of a computer is trusted, there is no guarantee that the system is not actually under the control of a virus, worm, or other attacker.

Consider a user who wishes to purchase something from a store. In most modern stores, the user swipes a credit card through a machine provided by the store. The machine shows the amount of the transaction on a visual display, and the user confirms the transaction by pressing a button on the machine. There are two threats to the user in this scenario. First, the store's machine may have misrepresented the amount of the transaction, showing one value but charging another to the account. Second, the user intends for only one transaction to occur. However, the machine has been provided with the credit card number, which is the only information necessary to make an unbounded number of transactions.

* This work was supported by a grant from Microsoft Corporation.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 225–239, 2005.
© Springer-Verlag Berlin Heidelberg 2005

To combat the latter threat, many systems have proposed the use of a trusted computing platform which performs sensitive operations in a secure way. For example, if the user in the previous example had been carrying a smart card, it would have been possible for the trusted smart card to produce a digital signature for a single transaction of a certain amount without revealing any additional information to the store's machine. Such a trusted platform must always be available to the user (either as a mobile device, or on a network) and must be resistant to external attacks.

Because of security, cost, and physical constraints, trusted platforms are often unable to interact directly with the user. For example, most smart cards rely on an expensive, non-portable reading device to display output and receive input from the user. In a point-of-sale setting such as the one described above, the reading device would likely be provided by the store. In the case of a trusted platform on a computer network, the user is clearly dependent on a local computing system to transmit and receive messages across the network.

The result of this lack of interactivity is that systems utilizing a trusted platform may be vulnerable to attack by an untrusted system which sits between the user and the trusted platform. Consider again the example of the shopping user, with one change: he now connects his smart card to the store's machine. The store's machine sends the amount of the transaction to the smart card. The card shows the amount to the user by asking the store's machine to display it. The user confirms the transaction by pressing a button on the store's machine, and the card finishes the transaction. Because all interaction has gone through the store's machine, which has the ability to alter messages in transit, the user cannot be sure that the amount displayed was the same as the amount sent to the smart card.

One way to solve this problem is to devise a protocol which allows a user to exchange messages with a trusted platform in a way that ensures the integrity and authenticity of the exchanged data. Furthermore, it must be possible for the actions required for one of the protocol's participants to be performed by a human without the aid of a computing device.

This paper introduces a novel approach for exchanging messages between a trusted computing platform and a human over an untrusted medium that is both secure and easy to use. The security of the approach is based on the difficulty that computers have in solving certain hard artificial intelligence (AI) problems. The user-friendliness is based on the natural ability of humans to solve the same problems. Section 2 introduces the concept of a keyed AI transformation, a general primitive for providing integrity and authenticity checks for messages with a human recipient. It also describes and analyzes a keyed AI transformation based on the problem of recognizing three dimensional (3-D) objects in a rendered scene. Section 3 describes a protocol for sending secure messages from a human to the trusted platform based on the keyed AI transformation primitive. Section 4 discusses the performance characteristics of the 3-D example, while sections 5 and 6 discuss related and future work.

## 2 Keyed AI Transformation

The notion of using hard artificial intelligence problems as security primitives has recently received a great deal of attention[1]. In short, a hard AI problem is a problem that can be solved by a majority of humans but not by a computer using state of the art AI programs (in the consensus of the AI research community). Security protocols can leverage the hardness of these problems just as cryptography has made use of hard problems in number theory, such as factoring.

A CAPTCHA is an automated test for telling humans and computers apart[1]. Many CAPTCHAs are based on a simple formula: randomly select a message, transform it in such a way that only a human can understand it, and ask the user to decipher all or part of the original message. If the response matches the original message, then the user is deemed human. For example, Yahoo wants to prevent automated programs from signing up for free email accounts. When creating an account, a user is presented with an image containing distorted text. To pass the test, the user must type one or more words that are represented in the image.

A trusted platform that wishes to protect a message being sent to a human receiver through a hostile medium can use a CAPTCHA transformation to provide confidentiality and integrity against attacking computers. Instead of randomly generating a message as above, the computer transforms the specific message that it wishes to send and transmits the result through the hostile medium. An attacking computer is unable to extract the original message from the transmitted message due to the properties of the transformation. Since any human can extract the message, the confidentiality provided is very weak. However, if a human attacker is not available to see the message in real-time, then the message can be considered temporarily confidential. This primitive is discussed further in section 3. An attacking computer will also have difficulty making a meaningful but undetectable modification to the message. Without the ability to determine which parts of the message are meaningful and which parts are obfuscation, modifying an existing image is difficult.

An attacking computer can, however, simply replace the entire message. Because all parameters of the transformation procedure are publicly known, there is nothing to uniquely identify the sending computer. A given message could have been created by any computer, not only by the trusted platform. To solve this problem, we introduce the notion of a keyed AI transformation, which is a function taking as input a message $m$ and a secret key $k$, and returning a new message $t$. Both the sender and the receiver of the message have knowledge of the key, but nobody else does. The sender's transformation incorporates the key, and the receiver verifies that the resulting message was made by a program with knowledge of the key. Because the receivers are humans, it must be easy for them to remember the key and to verify its usage in the message.

We define the security and usability properties of a transformation with the following parameters: A transformation $t = T(m, k)$ is an $(\alpha, \beta, \gamma, \delta, \epsilon, \tau)$-keyed transformation if and only if:

- the probability that a human can extract $m$ from $t$ is at least $\alpha$
- the probability that a human with knowledge of $k$ can correctly verify whether $k$ was used to create $t$ is at least $\beta$
- there does not exist a computer program that runs in time $\tau$ such that the probability of the program extracting $m$ from $t$ is greater than $\gamma$
- there does not exist a computer program that runs in time $\tau$ such that the probability of the program extracting $k$ from $t$ is greater than $\delta$
- let $A$ be a computer program that modifies $t$ such that a human will extract $m'$ from $t$ (with $m' \neq m$); there does not exist an $A$ that runs in time $\tau$ such that the probability of a human failing to detect the modification is greater than $\epsilon$

To be useful, a keyed transformation must be both user-friendly and secure. The parameters $\alpha$ and $\beta$ represent the ability of humans to use the system and would ideally be maximized. The values of $\alpha$ and $\beta$ for a specific transformation can be determined empirically.

The security of the system depends on the values of $\gamma$, $\delta$, and $\epsilon$. For maximum security, a transformation would ideally minimize all three values. A high value of $\gamma$ indicates that the message data can be intercepted by a computer, destroying even the temporary confidentiality provided by the system. Although this does not directly affect the integrity or authenticity of a message, there may be some security impacts. These are discussed in section 3. A high value of $\delta$ indicates that a computer attacker will have good success at learning the key. Since the key is the only feature distinguishing forged messages from authentic ones, if it is compromised, then arbitrary messages can be forged. Finally, $\epsilon$ represents the integrity of the transformation; if it is high, then a legitimate encoding can be undetectably changed to carry a different message. The values of $\gamma$, $\delta$, and $\epsilon$ can be determined by using computer programs to perform the stated tasks. The programs should represent state of the art techniques in the area of the transformation.

### 2.1 3-D Keyed Transformation

One example of a keyed transformation is based on the rendering of a three dimensional scene. The message is a 3-D object model to be sent from a trusted computer to a human, and the key is a set of 3-D object models. The message to be sent may be a model of a physical object, or of 3-D extruded text. The user and the trusted computer both know the key. The trusted computer knows the model data of the objects, while the user simply knows the appearance of the objects. The security of the transformation stems from the hardness of extracting components from the rendered scene, and from the difficulty in seamlessly modifying the image.

To encode a message, the trusted computer first creates an empty three dimensional scene. It then inserts a set of randomly colored planes to act as a background. The message model is placed in the scene at a random location. The coloration, rotation, and size of the model are determined randomly. Similarly, one or more instances of each key object are placed in the scene, each

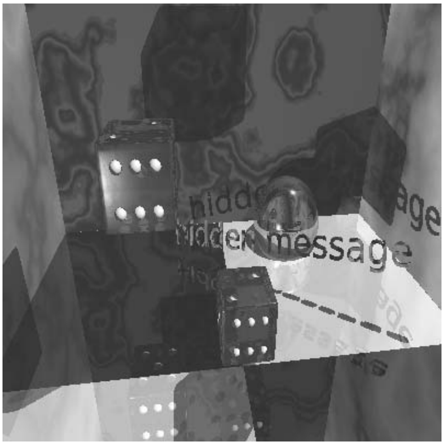


**Fig. 1.** 3-D Transformation with message “hidden message” and key “dice”

with a randomly selected rotation and size. Next, several objects with reflective surfaces are placed randomly throughout the scene. A camera location and light source are selected randomly, and the result is raytraced to create a two dimensional image, which is the final transformed output. An example is shown in figure 1. A human receiver of such an image would look for the presence of his key (“dice”) and read the message contents (“hidden message”).

Note that the random selection of the scene parameters presents a tradeoff between security and usability. Clearly some combinations of values will result in images in which the message or key objects are difficult for the user to see, or which are easy for a computer program to manipulate. For example, the message model must not be occluded by the key objects and vice versa. The colors and brightness used must provide sufficient contrast to make the models visible to a human viewer.

Unfortunately, there is no simple way to determine the best method for selecting these parameters. An implementation must use manually-coded heuristics to constrain the parameters; these heuristics are based on empirical measurements of human reaction to the scenes. The ranges for our proof-of-concept system were reached through a trial-and-error development process in which the parameter constraints were manually changed, a large number of scenes were randomly rendered, and the resulting images were evaluated for their usability and security properties. This process has led to an implementation that can input arbitrary text (within a certain length constraint) and key object models to produce scenes which have a high probability of being readable by humans, but from which current computer programs will have difficulty extracting the text and keys.

### 2.2 Attacks on 3-D Transformation

The security of the transformation is based on an assumption of the hardness of certain tasks. In this section, we analyze some possible attacks and show that their probabilities of success are bounded by the parameters of the transformation, including $\gamma$, $\delta$, and $\epsilon$.

Suppose that Carol is a computer that wishes to send a short text message $m$ to a human, Harry. Harry is sitting at a workstation named Wanda, which is able to connect to Carol across the Internet. Harry doesn't trust Wanda, but he wants to be sure that the message he receives was sent by Carol and that it was not modified in transit. Carol and Harry have previously established the secret key $(apple, orange)$ and Carol is storing a 3-D model of an apple and one of an orange. She renders $m$ along with the apple and orange models; the result is $t$, which she sends to Harry. Wanda is capable of discarding or modifying messages in transit, as well as creating new messages and displaying them to Harry. Wanda's goal is to convince Harry that the message is actually $m'$.

*Key Guessing.* Wanda can simply discard the message sent by Carol and instead send her own message. Suppose that Wanda has stored beforehand a set of object models. Harry will be looking for an image containing a message model (either text or a meaningful object) and all of his key objects (apples and oranges). Harry will tolerate the presence of other objects in the scene, since they may have been included to confuse an attacker. Wanda randomly selects $n$ objects from her set, renders them in a scene along with $m'$, and sends the result to Harry. If Wanda's image includes both apples and oranges (or objects that look similar enough to fool Harry), then Harry will accept the forged message as authentic.

Assume that Harry has selected $k$ object models from a finite list of $N$ models. Wanda includes $n$ models in her attempted forgery. If her set of $n$ models is a superset of Harry's set of $k$ objects, her forgery is successful. Assume that $n < N$ and $k < N$ (neither Wanda nor Harry can choose more objects than exist in the list). The probability $P$ of Wanda's success is

$$\begin{array}{l}\text{if } n < k \text{ then } P = 0 \\ \text{if } n \geq k \text{ then } P = \prod_{i=1}^{k} \frac{n-i+1}{N-i+1}\end{array}$$

It is therefore in Wanda's best interest to make $n$ as large as possible. However, $n$ is bounded by the amount of space in the image. In practice, she probably can't fit more than five or six objects into the scene before it gets too cluttered for Harry to identify anything. In order to lower Wanda's probability of success, Harry can increase either $k$ or $N$. Increasing $k$ has the side effect that Harry must remember and verify a larger number of objects; it is probably inconvenient for Harry to have more than three key objects. Increasing $N$ to be several orders of magnitude larger than $n$ gives Wanda a very low probability of successfully guessing the key objects. Ideally, $N$ would be unbounded or impossible for Wanda to enumerate. If each individual user can have an arbitrary model, then Wanda will not be able to store a pre-determined set of possible objects.

In the previous analysis, it was assumed that Harry picked a model from the $N$ choices using a uniform distribution. In reality, Harry's selection will be influenced by the appearance of the models and his particular interests. For example, a model of a very cute puppy will likely be chosen more frequently than one of asparagus. Wanda can weight her selections to achieve a higher success rate. This is similar to using a dictionary attack against easy-to-guess text passwords. People are more likely to choose a password based on their login name, pet's name, or words from a dictionary than they are to choose a random string. Such easy to guess passwords should be avoided. One solution is to randomly generate a password. With text passwords, this frequently leads to passwords that are difficult to remember. However, randomly assigning an object model is much more user-friendly; the user only needs to remember the appearance of a few objects.

*Convert 2-D to 3-D.* Wanda can intercept $t$ (Carol's 2-D image) and attempt to "reverse render" it back to a 3-D scene description. Once she has the original scene description, she can replace $m$ with $m'$, re-render the image, and send it to Harry. The probability of Wanda successfully performing the reverse render has an upper bound of $\min(\gamma, \delta)$. This follows intuitively from the fact that recreating the original scene would allow extracting both the message and the key from Carol's image. The probabilities of performing those tasks are bounded by $\gamma$ and $\delta$ respectively. The values of $\gamma$ and $\delta$ are discussed in the next two attacks.

*Extract Key Objects.* Wanda can try to extract the key objects, render a new scene with the extracted objects and $m'$, and send the result to Harry. Her probability of success in extracting the keys is bounded by $\delta$. It is also likely that key objects will be partially occluded. Wanda will have to guess at the missing portion or attempt to cover it up with text or other objects in her new scene.

Moreover, even if she is able to locate the objects within the image, Wanda will have only a single perspective on a three dimensional object. When placing the object in her newly rendered scene, she may attempt either to render the object from the same perspective or to extrapolate other perspectives on the object. Rendering from the same perspective requires that the camera angle, camera direction, and lighting be identical; it is difficult to calculate these parameters from only the 2-D rendered image. If Wanda chooses instead to extrapolate other perspectives on the object, she will have to guess at the appearance of the hidden sides of the object. Her guesses must be accurate enough to fool Harry, which should be difficult for non-trivial objects.

*Modify 2-D Image.* Rather than attempting to extract the 3-D information from the image, Wanda can simply attempt to insert, delete, or modify the text of the 2-D image. The probability of her changes not being detected by Harry has an upper bound of $\epsilon$.

The parameter $\epsilon$ is kept low by the "reality" of the 3-D scene. Harry has a probability of detecting odd reflections, out-of-place colors, or letters with unmatched angles. His ability to do so is expressed by $\epsilon$. For example, consider the case of deleting text. Wanda must not only identify the text and locate its exact boundaries, but she must extrapolate the objects and background that were not visible behind the text. Furthermore, deleting one image of the text isn't enough. The same text may appear as a reflection in other places, using different colors or shapes.

Insertion of text is somewhat easier, since it is covering up sections of the image rather than revealing them. However, to make a believable result, Wanda will have to account for added shadow and reflection in the scene. Without knowing the coordinates of the original light source or camera, accurately calculating those features is very difficult.

### 2.3 Human Adversary

The security parameters of the keyed transformation are based on foiling a non-human attacker. In some situations, this may be a reasonable assumption. An untrusted environment may be physically secured from other humans and blocked from accessing a network. However, in many cases human cooperation with a malicious computer is a possibility. It is assumed that a human attacker can easily extract the message text and key from a transformation. We will call such an attacker Robert, since he can "Read" the data in the transformation.

Consider the 3-D keyed transformation and the example given. Once a message is transmitted, Wanda can send it to Robert. Robert extracts the key and sends it back to Wanda, who renders a new scene with an alternate message and sends it to Harry. One problem for Wanda is that Robert may have trouble communicating the key objects to her. Robert may know the objects are apples and oranges, but he might not have a 3-D model of either fruit on hand. If there is a finite list of models, he can probably choose the correct one. However, if the model is not taken from a public list, he will have to find or construct models of apples and oranges. This significantly increases Robert's work, and makes it difficult for him to cooperate in real-time.

It is, in fact, feasible for a keyed transformation to have a key whose features are practically impossible for humans to articulate. An important point is that, due to the AI domain, Harry doesn't need to be able to describe the key. He only needs to be able to recognize the key's presence in a transformation. When Harry and Carol initially agree on a key, Carol can randomly generate a key and show it to Harry. Harry learns the key well enough during this procedure to verify messages later.

### 2.4 Other Keyed Transformations

Keyed transformations are intended as a general security primitive. The example given is meant to be illustrative, and is by no means the only keyed transformation available. Other transformations can be created using different hard AI problems.

For example, there has been some exploration of speech as a security primitive [2]. It may be possible to construct a keyed transformation by synthesizing text to speech using a particular voice (and obscuring the voice with audible noise to make automated analysis difficult). The shared key would be the parameters used to generate the voice. Instead of memorizing the vocal parameters, the human verifier would simply recognize the voice. This system is very user-friendly; recognizing voices is already a well-established security mechanism used between humans. Furthermore, the speech transformation would be more resistant to human attempts at disclosing the key or modifying the audio stream, potentially making it a better choice than the 3-D transformation.

## 3 Protocol Description

The previous section has shown that a keyed transformation can provide for the authenticity of messages sent from a trusted computer to a human. However, it may be the case that other operations are of equal interest. For example, a human may want to send a message to a computer, ensuring that it arrived intact, or a computer may want to send a message to a human and ensure that the human received it. This section describes a protocol based on keyed transformations for sending reliable, authenticated messages between a human and a computer through an untrusted intermediary.

The participants in the protocol are again Harry (the human) and Carol (the computer). Wanda (Harry's workstation) is able to modify or delete messages between Harry and Carol, and may introduce arbitrary messages. Harry and Carol have previously agreed upon a shared key $k$ and a keyed transformation $T$.

Imagine Harry wants to send a message $m$ to Carol. He wants to know whether she received the message without modification, and he wants her to know whether the message was genuine. The protocol works as follows:

1. Harry transmits the message $m$ to Carol without any security features
2. Carol computes $t = T(m, k)$ and transmits it to Harry
3. Harry verifies the authenticity of $t$ based on the shared key $k$
4. Harry extracts the message text from $t$ and confirms that it is equivalent to the original $m$
5. Harry transmits a single secure bit to Carol indicating whether the message was correctly received

Similarly, imagine that Carol has a message $m$ that she wants to send to Harry. She uses the following protocol:

1. Carol computes $t = T(m, k)$ and transmits it to Harry
2. Harry verifies the authenticity of $t$ based on the shared key $k$
3. Harry extracts the message from $t$
4. Harry transmits a single secure bit to Carol indicating that he has received the message

Note that Wanda can send an arbitrary message $m$ to Carol, who will return $T(m, k)$. If Harry and Carol are using the same key to send messages in both directions, then the resulting transformation can be used by Wanda to spoof arbitrary messages from Carol to Harry. It is therefore critical in such a bidirectional system that Carol and Harry agree upon separate keys for original messages and confirmation messages.

The protocol is built on the concept that given trusted data flow in one direction (provided by the keyed transformation) and a single bit of trusted flow in the other direction, arbitrary trusted input can be constructed[3]. It is not immediately obvious how the final step in each procedure, the sending of the single secure bit, is performed. The decision will vary from system to system, depending on the capabilities of Harry and Carol, the desired level of security, and the desired ease of use of the system. This section describes two methods with very different characteristics and analyzes their properties.

*Physical Interaction.* If Carol is a mobile device with which Harry can physically interact, then he may be able to send a signal to her directly. Such mobile devices may not have buttons or other direct input features. In this case, some procedure must be devised, such as interrupting the interaction between the device and the host system. For example, if Carol is a USB device, Harry can unplug her. If she is a smart card, he can remove her from the reader.

If disconnecting Carol sends a "0" bit (indicating that the message was tampered with), then what sends a "1" bit (confirmation that the message was valid)? Since there are no other signals, Carol can assume that the failure to be disconnected within a certain time period constitutes a confirmation. Therefore Carol waits $N$ seconds after sending a response before believing the authenticity of a received message. If she is disconnected before her timer has elapsed, then the message is discarded.

The intuitiveness of this scheme is appealing. If Harry detects that Wanda is cheating, he simply disconnects his device and walks away. There is an obvious problem with this scheme, however: the confirmation is implicit. That is, without any interaction from Harry, Carol assumes that a message is valid. To exploit this, Wanda could wait for a time when Harry is not interacting with Carol. She sends Carol a message, who responds with a transformation. Wanda discards the transformation. Harry, unaware that any of this is happening, does not disconnect Carol. After Carol's timer expires, she accepts the message as valid. One easy solution to this problem is to keep Carol disconnected when she is not in use. The practicality of connecting Carol only when needed depends on Harry's situation. If his only operation with the malicious terminal is performing a single interaction (such as making a financial transaction in the checkout line of a store), then he can simply connect his device when he's ready to use it. On the other hand, if Harry is using a workstation for several hours, he doesn't want to repeatedly connect and disconnect his device every time it is used.

Another way to combat this problem is to remove the implicit confirmation by reversing the meaning of the messages. That is, Carol requires an explicit disconnection within a certain time period to confirm the message. Because some tamper resistant mobile devices are powered by the host computer, Harry may have to reconnect Carol for her to do useful work on the message. There are two problems with this. One is that Carol may not be able to keep sufficient state across a disconnect; once reconnected, the context and data of the original message may have been lost. The other problem is that Harry's actions are not intuitive for humans. In order to confirm a message, he must disconnect and reconnect his mobile device. In order to reject it, he must leave his device connected and wait for Carol's timer to expire.

*One-time Secret.* A much more general approach is to assume that Harry can send only insecure messages to Carol through Wanda. In this case, Harry and Carol pre-arrange a particular confirmation word. After receiving a message, Carol sends the response and waits for input from Harry. When she receives the expected word, the message is confirmed as genuine. A lockdown mechanism prevents Wanda from using a brute-force attack to guess the word (e.g., Carol might accept only a few incorrect attempts before considering the message a forgery). Harry is responsible for typing the word to confirm the message. He and Carol may have agreed beforehand upon the confirmation word, or Carol may randomly generate a word and embed it in her transformed response.

Clearly the confirmation word must be kept secret until it is used. Furthermore, in using it, the word is revealed to Wanda. Thus, each word can be used only once. This may create a usability problem for Harry if the words are established with Carol beforehand. Rather than simply remembering that his key is "apples and oranges" he now must remember a unique word for each transaction. With a long list, he will almost certainly need to carry a memory aid such as a paper list. With a short list, there is a limit to the number of transactions he can perform.

Including the confirmation word in the transformed response from Carol increases the user-friendliness of the system but may decrease the security. In this case, Harry is not required to remember anything; he simply must type a word that he sees embedded in the transformation. However, the secrecy of the word relies on the temporary confidentiality provided by the transformation. If Wanda can extract the confirmation word, either herself or with the help of Robert, then she can send and confirm arbitrary messages.

# 4 Performance

Besides the security and usability of the proposed system, performance is also of interest. Keyed transformations rely on hard problems which are often resource intensive (e.g., the 3-D transformation). Furthermore, the system is targeted towards areas of secure computation, including mobile devices which are often severely constrained in terms of processing and storage resources.

This section examines the resource requirements of the 3-D transformation and looks at the implications of offloading processing and storage requirements to a third-party platform.

## 4.1 Computational Resources

Our current unoptimized implementation of the 3-D keyed transformation requires between two and ten seconds per message on a modern workstation, depending on the key object models used. This makes the system usable on a small scale, but may be a problem for high-volume servers. Such servers may be able to increase performance through the use of specialized graphics hardware.

This processing requirement is completely unacceptable for low-powered mobile devices. For example, current smart cards typically operate at speeds of eight megahertz or less. The only option in this case is to use a separate computation server to do the rendering. The computation server is assumed to be accessible over a potentially hostile network; it never directly interacts with Harry. Communication between the trusted platform and the server is secured using traditional cryptographic methods. Carol's requests contain a scene to be rendered and are signed and encrypted so that only the server can read them. The response, containing the transformed image, is similarly authenticated.

From a security perspective, the ideal situation is that the computation server can be trusted not to disclose the key. If the server cannot be trusted, then forgeries may be possible. Because the server never interacts directly with Harry, it cannot forge or modify messages between Carol and Harry in the same way that Wanda can. However, if the server and Wanda cooperate, then arbitrary forgeries can be created. When Harry sends a message $m$, Wanda can send Carol an alternate message $m'$ but ask the server to do the transformation using $m$. Carol thus returns to Harry a rendering with the message he expects. However, Carol has received Wanda's alternate message. When Harry sends the confirmation to Carol, he is confirming the wrong message. Alternatively, Wanda can ask the server to disclose the key to her, and she can do the rendering herself. Carol cannot display the returned image directly to Harry; she must ask Wanda to do it for her. Wanda can discard Carol's image and show her own rendering instead.

This attack relies on Wanda and the server being able to match messages to rendering requests. If Wanda knows how to contact the server directly, then she can simply send $m$ and $m'$ to the server. The server finds a rendering request that includes $m'$ and replaces the message with $m$. If Carol is relying on Wanda to send messages through the network (as is the case with many mobile devices), then Wanda knows the address of the computation server and can communicate directly with it.

To prevent direct communication, Carol can send the rendering request through a mix-net[4] or other anonymizing system. In this case, the server doesn't know Wanda's address, so it cannot contact her directly. The anonymizing system can also deliver the rendering request to a random computation server, preventing Wanda from knowing the server's address. The two malicious parties (the server and Wanda) may still be able to simply guess each other's addresses. To reduce the probability of successful guessing, the list of possible servers should be large, as should the list of possible "Wandas." Furthermore, it may be useful for legitimate computation servers to keep track of requests from malicious terminals looking for malicious computation servers. By tracking these terminals, a blacklist of malicious terminals can be created to warn users.

Anonymizing requests to the computation servers prevents direct communication, but there may still be a covert channel. The transformed image is created by the server, sent to Carol, and then given to Wanda to display. The server can use steganography to encode information about the key in the returned image[5]. When Carol sends the image to Wanda, Wanda extracts the key, renders an alternate scene, and displays her new scene in place of the one Carol sent. To prevent the use of this covert channel, either Carol or the mix-net should attempt to strip steganographic information from the returned image[6].

### 4.2 Storage Resources

The 3-D models in our current implementation are described by the construction of geometric solids, and are thus on the order of a few kilobytes (stored in a human-readable form). Using polygonal mesh models would make the size much larger. It should be trivial for modern servers to store models of either type.

For resource constrained systems, however, even a few kilobytes of data may be pushing the limit. For many models, compression is a good solution (our human-readable models showed 75% compression using standard Lempel-Ziv encoding).

Another solution is to simply store a pointer to a model, rather than the model itself. Many constrained devices will have to offload the processing anyway; in this case, there is no need to store the model directly. If there are a finite number of models, then the trusted platform can simply store an index into the list of models (e.g., storing the word “dice” rather than the actual model; the computation server has a model that matches “dice”). Providing such a master list of models may make the key guessing attack described in section 2.2 easier. Wanda can create her list of guesses based on the list of models.

One final solution would be to have a separate storage location that is not trusted, but used only to store encrypted data. The constrained system would store only the key; the encrypted data and the key (encrypted so that only the computation server can read them) would be sent to the computation server. If each trusted platform provides its own model, then an arbitrary number of models can be used, making the key guessing attack more difficult. However, the user now has an additional responsibility to provide the storage (perhaps by carrying a business-card CD, USB storage device, or other medium).

## 5 Related Work

Previous work in the area of human-computer security protocols has frequently focused on authenticating a human user to a computer[7, 8]. Hopper and Blum provide a discussion of such human authentication protocols in [8]. However, the systems described provide only authentication of the involved parties, not the authenticity and integrity of individual messages. Furthermore, the practicality of such systems is in question.

Naor and Pinkas propose the use of visual cryptography for message authentication [9]. The user carries a printed transparency as a key; the cipher text is an image of the same size. When a user places the transparency over the image, the plaintext is revealed (with some amount of random noise). This system both requires the user to carry a physical transparency and can only be used a limited number of times.

Gobioff et al describe protocols for providing trusted input and output with a smart card[3]. However, their suggestions rely on installing low-bandwidth (e.g., single-bit) trusted input and output paths to the card. To date, this has not been done.

Stabell-Kulø et al define a protocol for authenticating messages to be signed by a smart card[10]. Their proposal relies on the user computing a one-time pad substitution cipher over the message. This not only requires the user to carry a memory aid of the substitution, but it is a computational burden to the user. They do, however, introduce the concept of a confirmation word as used in section 3 of this paper.

## 6 Conclusions and Future Work

The use of hard AI problems as security primitives is a relatively new field. This paper explains the concept of using AI problems to check the integrity and authenticity of messages, as well as introducing metrics for evaluating individual problems. The construction of the 3-D keyed transformation highlights some of the difficulties of this approach, both in terms of security and usability.

The next step in this line of research is to develop and evaluate individual keyed AI transformations. The 3-D transformation described has been subjected to very limited usability testing. Similarly, the difficulty in breaking the transformation has been evaluated only on a small scale. The true values for $\gamma$, $\delta$, and $\epsilon$ can only be determined by the consensus of the AI and security research communities. Even if the 3-D transformation proves useful, other transformations may have desirable properties. For example, an audio-based transformation may be useful alongside a visual transformation to provide greater accessibility.

Furthermore, complete systems that use keyed transformations need to be developed and deployed. Many of the details of the protocols are specific to individual situations (e.g., the capabilities of the trusted computing device).

Using hard AI problems as security primitives is a little-explored but worthwhile pursuit. In particular, many of the techniques map directly to ways in which humans intuitively ensure security in the real world. The 3-D transformation attempts to make a coherent scene so that cut-and-paste forgeries are impossible. Similarly, nobody would trust a paper contract in which a piece of paper with text had been glued on top the original document. People authenticate each other over the telephone using the unique sound of their voices. The speech transformation attempts to do exactly this by giving the trusted computer a unique, audible voice.

This paper takes a clear step in the direction of human-computer authentication. It is hoped that researchers, both in the security community and without, will help to advance this field both by developing schemes based on specific problems, and by trying to break existing schemes by solving the underlying problems.

## References

1. von Ahn, L., Blum, M., Hopper, N.J., Langford, J.: CAPTCHA: Using hard AI problems for security. In: Advances in Cryptology – Eurocrypt 2003. Volume 2656 of Lecture Notes in Computer Science., Springer-Verlag (2003)
2. Kochanski, G., Lopresti, D., Shih, C.: A reverse turing test using speech. In: Proceedings of the International Conferences on Spoken Language Processing, Denver, Colorado (2002) 1357–1360
3. Gobioff, H., Smith, S., Tygar, J.D., Yee, B.: Smart cards in hostile environments. In: Proceedings of the Second USENIX Workshop on Electronic Commerce. (1996)
4. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM **24** (1981) 84–88
5. Smith, J.R., Comiskey, B.O.: Modulation and information hiding in images. In: Workshop on Information Hiding. Volume 1174., Isaac Newton Institute, University of Cambridge, UK (1996) 207–226
6. Johnson, N.F., Jajodia, S.: Steganalysis of images created using current steganography software. In: Second Workshop on Information Hiding. Volume 1525 of Lecture Notes in Computer Science., Portland, Oregon, USA (1998) 273–289
7. Matsumoto, T.: Human-computer cryptography: an attempt. In: Proceedings of the 3rd ACM conference on Computer and communications security, ACM Press (1996) 68–75
8. Hopper, N.J., Blum, M.: Secure human identification protocols. In: Advances in Cryptology – Asiacrypt 2001. Volume 2248 of Lecture Notes in Computer Science., Springer-Verlag (2001) 52–66
9. Naor, M., Pinkas, B.: Visual authentication and identification. In: Advances in Cryptology – Crypto '97. Volume 1294 of Lecture Notes in Computer Science., Springer-Verlag (1997) 322–336
10. Stabell-Kulø, T., Arild, R., Myrvang, P.H.: Providing authentication to messages signed with a smart card in hostile environments. In: USENIX Workshop on Smartcard Technology. (1999)

# Approximate Message Authentication and Biometric Entity Authentication*

G. Di Crescenzo[1], R. Graveman[2], R. Ge[3], and G. Arce[3]

[1] Telcordia Technologies, Piscataway, NJ
giovanni@research.telcordia.com
[2] Work done while at Telcordia Technologies
rfg@acm.org
[3] University of Delaware, Newark, DE
{ge, arce}@ece.udel.edu

**Abstract.** Approximate Message Authentication Code (AMAC) is a recently introduced cryptographic primitive with several applications in the areas of cryptography and coding theory. Briefly speaking, AMACs represent a way to provide data authentication that is tolerant to acceptable modifications of the original message. Although constructs had been proposed for this primitive, no security analysis or even modeling had been done.

In this paper we propose a rigorous model for the design and security analysis of AMACs. We then present two AMAC constructions with desirable efficiency and security properties.

AMAC is a useful primitive with several applications of different nature. A major one, that we study in this paper, is that of entity authentication via biometric techniques or passwords over noisy channels. We present a formal model for the design and analysis of biometric entity authentication schemes and show simple and natural constructions of such schemes starting from any AMAC.

## 1 Introduction

The rise of financial crimes such as identity theft (recent surveys show there are currently 7-10 million victims per year) and check fraud (more than 500 million checks are forged annually with losses totaling more than 10 Billion dollars in the United States alone) is challenging financial institutions to meeting high security levels of entity authentication and data integrity. Passwords are a good start to secure access to their systems but, when used alone, don't seem

* Copyright Telcordia Technologies. Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 240–254, 2005.
© Springer-Verlag Berlin Heidelberg 2005

enough to provide the security and convenience level for identification needed by financial organizations. (Passwords can be compromised, stolen, shared, or just forgotten.) Biometrics, on the other hand, are based on a user's unique biological characteristics, and can be an effective *additional* solution to the entity authentication problem for financial systems. One challenge in implementing biometric authentication is, however, the reliability of the system with respect to errors in repeated measurements of the same biometric data, such as fingerprints, voice messages, or iris scans.

In this paper we put forward a formal model for the study of approximate data authentication schemes, that are tolerant with respect to errors in the data, and therefore are suitable for the verification of biometric data in entity authentication schemes. We then present efficient constructions of approximate data authentication, and use them to obtain efficient constructions for two types of biometric entity authentication schemes.

Data Authentication. A fundamental cryptographic primitive is that of Message Authentication Codes (MAC), namely, methods for convincing a recipient of a message that the received data is the same that originated from the sender. MACs are extremely important in today's design of secure systems since they reveal to be useful both as atomic components of more complex cryptographic systems and as themselves alone, to guarantee integrity of stored and transmitted data. Traditional message authentication schemes create a hard authenticator, where modifying a single message bit would result in a modification of about half the authentication tag. These MACs fit those applications where the security requirement asks to reject any message that has been altered to the minimal extent. In many other applications, such as those concerning biometric data, there may be certain modifications to the message that may be acceptable to sender and receiver, such as errors in reading biometric data or in communicating passwords through very noisy channels. This new scenario, not captured by the traditional notion of MACs, motivated the introduction and study in [6] of a new cryptographic primitive, a variant of MACs, which was called Approximate Message Authentication Code (AMAC); namely, methods that propagate "acceptable" modifications to the message to "recognizable" modifications in the authentication tag, and still retain their security against other, "unacceptable" modifications. Examples of the applicability of AMACs include: message authentication in highly-noisy or highly-adversarial communication channels, as in mobile ad hoc networks; simultaneous authentication of sets of semantically equivalent messages; and, of specific interest in this paper, entity authentication through inherently noisy data, such as biometrics or passwords over noisy channels.

Our Contributions. If, on one hand, after investigations in [6, 17], the intended notion of AMAC was precisely formulated, on the other hand, a rigorous model for the security study of AMACs was not. Therefore, a problem implicitly left open by [6, 17] was that of establishing such a model. In this paper we propose a rigorous model for analyzing approximation in message authentication. It turns out that the issue of approximation has to be considered in both the correctness property (if Alice and Bob share a key and follow the protocol, then Bob accepts the message) and the security property (no efficient adversary not knowing the shared key and mounting a chosen message attack can make Bob accept a new message). Our notions of approximate correctness and approximate security use as a starting point the previously proposed notions for conventional MACs and address one difficulty encountered in both allowing acceptable modifications to the message and achieving a meaningful security notion. In addition, we formulate two preimage-resistance requirements that make these AMACs especially applicable to two variants of biometric entity authentication problems.

We then present two AMAC constructions: the first scheme uses systematic error correcting codes, is stateless and satisfies our weaker notion of preimage resistance; the second scheme solves the technical problem of constructing a probabilistic universal one-way hash function with distance-preserving properties, is counter-based and satisfies our stronger notion of preimage resistance. Both constructions can be implemented quite easily and only use symmetric primitives.

We then show how to apply these constructions (and, in fact, any AMAC scheme) to obtain simple and efficient biometric entity authentication schemes in both a closed-network and an open-network setting, for which we also present a formal model. Our scheme are non-interactive and can be seen as an extension, using biometrics, of well-known password-based entity authentication schemes.

Formal proofs and some definitions are only briefly sketched due to lack of space.

Related Work. References in conventional Message Authentication Codes are discussed in Section 2. Universal one-way hash function were introduced in [14] and are being often applied in cryptographic constructions. Related work to AMACs includes work from a few different research literatures.

There is a large literature that investigates biometric techniques without addressing security properties (see, e.g. [8] and references therein). Security and privacy issues in biometrics have been independently recognized and advocated by many researchers (see, e.g., [3, 15, 16]).

A second literature (related to information and coding theory) investigates techniques for authentication of noisy multimedia messages (see, e.g., [12, 13] and references therein). All these constructs either ignore security issues or treat them according to information theoretic models. Typically, constructions of the latter type have a natural adaptation to the symmetric MAC setting but all constructions we found, after this adaptation, fail to satisfy the MAC requirement of security under chosen message attack (and therefore the analogue AMAC requirement). Some works use digital signatures as atomic components but they result in constructions that are not preimage-resistant, according to our Definition 2, and therefore cannot be applied to give a satisfactory solution to our biometric authentication problem.

A third literature investigates coding and combinatorial techniques for error tolerance in biometrics (see, e.g., [10, 9]), as well as privacy amplification from reconciliation. Recently, [5, 2] considered the problem of generating strongly random keys from biometric data. Although these constructions might be useful towards solving the problem of biometric entity authentication, current proposals fall short of achieving this. In particular, the proposal in [5] was broken by [2] in the setting of identification to multiple servers; and the (interactive) proposal of [2] is still based on some (somewhat questionable) assumption referring to biometrics as entropy sources. Yet, these papers address interesting primitives and notions (fuzzy commitments, fuzzy extractors, etc.) unaddressed by ours and viceversa. Our non-interactive proposal is based on a very intuitive and perhaps minimal assumption on biometrics.

We stress that all this previous work did not even imply a formal definition of AMACs.

## 2 Definitions and Preliminaries

In this section we present our novel definition of Approximate MACs. In the rest of the paper we will assume familiarity with definitions of cryptographic primitives used in the paper, such as universal one-way hash functions, (conventional) MACs, symmetric encryption schemes and finite pseudo-random functions.

**Approximation in MACs.** We introduce formal definitions for approximate MACs, using as a starting point the well-known definition for conventional MACs. Informally, one would like an approximate MAC to be tolerant to "acceptable" modifications to the original message. Less informally, we will define approximate versions of the same properties as an ordinary MAC, where the approximation is measured according to some polynomial-time computable distance function on the message space. For the correctness property, the notion of a modification being acceptable is formalized by requiring an authentication tag computed for some message $m$, to be verified as correct even for messages having up to a given distance from $m$. We note that this property might not be compatible with the property of security against chosen message attack, for the following reason. The latter property makes an adversary unable to produce a valid pair of message and authentication tag, for a new message, for which he hasn't seen an authentication tag so far; the former property, instead, requires the receiver himself to be able to do so for some messages, that is, for messages having a certain distance from the original message obtained from the sender. In order to avoid this apparent definitional contradiction, we define a chosen message attack to be successful if the valid pair of message and authentication tag produced by the adversary contains a message which has a larger distance from all messages for which he has seen an authentication tag during his chosen message attack. Therefore, we even define the security property for MACs in some approximate sense. We now proceed more formally.

**Definition 1.** Let $M$ denote the message space and let $d$ be a polynomial-time computable distance function over $M$. An *approximately correct and approximately secure message authentication code for distance function $d$* (briefly, $d$-ac-as-MAC) is a triple (Kg,Tag,Verify), where the polynomial-time algorithms

Kg, Tag, Verify satisfy the following syntax. The key-generation algorithm Kg takes as input a security parameter $1^l$, and distance function $d$, and returns an $l$-bit secret key $k$. The authenticating algorithm Tag takes as input a message $m$, a secret key $k$, and distance function $d$, and returns a string *tag*. The verifying algorithm Verify takes as input a message $m$, a secret key $k$, a string *tag*, and distance function $d$, and returns a value $\in$ {yes,no}. Moreover, the triple (Kg,Tag,Verify) satisfies the following two requirements.

1. $(d, p, \delta)$-*Approximate Correctness*: after $k$ is generated using Kg, if *tag* is generated using algorithm Tag on input message $m$ and key $k$, then, with probability at least $p$, algorithm Verify, on input $k, m', tag$, outputs: *yes*, if $d(m, m') \leq \delta$.
2. $(d, \gamma, t, q, \epsilon)$-*Approximate Security*: Let $k$ be generated using Kg; for any algorithm *Adv* running in time at most $t$, if *Adv* queries algorithm Tag$(k, \cdot)$ with adaptively chosen messages, thus obtaining pairs $(m_1, t_1), \ldots, (m_q, t_q)$, and then returns a pair $(m, t)$, the probability that Verify$(k, m, t) =$ *yes* and $d(m, m_i) \geq \gamma$ for $i = 1, \ldots, q$, is at most $\epsilon$.

Note that $(t, q, \epsilon)$-secure MAC schemes are $(d, p, \delta)$-approximately correct and $(d, \gamma, t, q, \epsilon)$-approximately secure MAC schemes for $p = 1$, $\delta = 0$, $\gamma = 1$, and $d$ equal to the Hamming distance. In the sequel, we will omit $d$ in the term $d$-ac-as-MAC when clear from the context, or directly abbreviate the term $d$-ac-as-MAC as AMAC. Although not included in the above definition, as for conventional MACs, an important *efficiency requirement* for AMACs is that the size of the tag is desired to be significantly smaller than the length of the input message.

**Two Additional Properties of AMACs.** In certain applications of AMACs as those considered in this paper, it may be desirable that the AMAC tag does not help in recovering any message for which that tag is valid. We formally define two variants of a 'preimage-resistance' property. In the first variant, called 'weak preimage-resistance', we require that the tagging algorithm, if viewed as a function on the message space, is hard to invert, no matter what is the distribution on the message space. (Later, while showing the applications of AMACs to biometric entity authentication, this property will be useful in proving that the entity authentication scheme obtained is secure against adversaries that can gain access to the AMAC output from the biometric storage file.) In the second variant, called 'strong preimage-resistance', we require that this property holds even if the adversary is given access to the receiver's private key. We now formally define both properties.

**Definition 2.** The $d$-ac-as-MAC (Kg,Tag,Verify) is $(d, t, q, \epsilon)$-*weakly-preimage-resistant* if the following holds. Let $k$ be generated using Kg; and assume that an efficient algorithm *Adv* obtains from an oracle O$(d, k)$ valid tags $t_1, \ldots, t_q$; that is, tags for which there exist messages $m_1, \ldots, m_q$, independently drawn from some efficiently samplable distribution $D_m$, such that $t_i =$Tag$(d, k, m_i)$, for $i = 1, \ldots, q$. For any such *Adv* running in time at most $t$, the probability that *Adv*$(d, M, t_1, \ldots, t_q)$ returns $m'$ such that Verify$(d, k, m', t_i) = 1$ for some $i \in \{1, \ldots, q\}$, is at most $\epsilon$. Furthermore, we say that the $d$-ac-as-MAC (Kg,Tag,Verify) is $(t, \epsilon)$-*strongly-preimage-resistant* if the above holds even with respect to algorithms *Adv* who takes $k$ as an additional input.

We note that essentially all conventional MAC constructions in the literature would satisfy an analogue preimage-resistance requirement. However it is easy to transform a MAC into one that is not weakly preimage-resistant and for some applications like biometric identification, it may be very desirable to require that the AMAC used is weakly or strongly preimage-resistant (or otherwise an accidental loss of the AMAC output or the server's private key could reveal a password or some biometric data to an adversary).

**Previous Work on AMACs.** Previously to this work, variations of a single approximate MAC contruction had been proposed and investigated in [6, 17]. Informally, the tagging algorithm in these constructions uses operations such as xoring the message with a pseudo-random string of the same length, computing a pseudo-random permutation of the message, and returning majority values of subsets of message bits. As already observed in [4], it can be seen that these constructions are secure against an adversary that cannot mount a chosen message attack; while they are not intended to be secure under a sufficiently long chosen message attack, since they only use a polynomial amount of pseudorandomness.

**Simple Attempts Towards AMAC Constructions.** First of all, we remark that several simple constructions using arbitrary error correcting codes and ordinary MACs fail in satisfying even the approximate correctness and security requirements of AMACs. These include techniques such as interpreting the input message as a codeword, and using a conventional MAC to authenticate its decoding (here, the property of approximate correctness fails). Other techniques that also fail are similar uses of fuzzy commitments from [10], fuzzy sketches from [5] and reusable fuzzy extractors from [2]. We note however that there are a few simple constructions that meet the approximate correctness and security requirements of AMACs but don't meet the preimage-resistance or the efficiency requirements. The simplest we found goes as follows. Let us denote as (K,T,V) a conventional MAC scheme. The tagging algorithm, on input key $k$ and message $m$, returns $tag = m \,|\, \mathrm{T}(k, m)$. The verifying algorithm, on input $k, m', tag$, sets $tag = t1 \,|\, t2$ and returns 1 if and only if $d(t1, m') \leq \delta$ and $V(k, t1, t2) = 1$, where $d$ is the distance function. The scheme satisfies the approximate correctness and security; however, note that the tag of this scheme contains the message itself and therefore the scheme is neither preimage-resistant nor efficient.

## 3 Our AMAC Constructions

In this section we present two constructions of approximately-correct and approximately secure MACs with respect to the Hamming distance. The first construction is stateless and weakly preimage-resistant under the existence of secure symmetric encryption schemes and weakly preimage-resistant conventional

MACs. The second construction, the main one in the paper, is counter-based and strongly preimage-resistant under the existence of collision-intractable hash functions.

### 3.1 A Weakly Preimage-Resistant AMAC Construction

A construction of an AMAC for the Hamming distance function can be obtained by using any conventional MAC scheme, any symmetric encryption scheme, and any appropriate systematic error correcting code. The construction satisfies approximate correctness with optimal parameter $p = 1$ and approximate security with optimal parameter $\gamma = \delta + 1$.

**Formal Description.** Let us denote by (K$_a$,T,V) a conventional MAC scheme, and by (K$_e$,E,D) a symmetric encryption scheme. Also, by (SEnc,SDec) we denote a systematic error-correcting code (that is, on input $m$, SEnc$(m) = c$, where $c = m|pc$, and $pc$ are parity check bits), such that the decoding algorithm perfectly recovers the message if at most $\delta$ errors happened or returns failure symbol $\perp$ otherwise (this latter condition is without loss of generality as any error correcting code can be simply transformed into one that satisfies it).

**Instructions for Kg:** generate a uniformly distributed $k$-bit key $K$

**Input to Tag:** two $k$-bit keys $K_a, K_e$, an $n$-bit message $M$, parameters $p, \delta, \gamma$.

**Instructions for Tag:**

```
1. Set $c = Enc(M)$ and write $c$ as $c = M|pc$
2. Set $subtag = T_{K_a}(M)$ and $epc = E(K_e, pc)$
3. **Return:** $tag = epc|subtag$ and halt.
```

**Input to Verify:** parameters $p, \delta, \gamma$, two $k$-bit keys $K_a, K_e$, an $n$-bit message $M'$ and a string $tag$

**Instructions for Verify:**

```
1. Write $tag$ as $tag = epc|subtag$
2. Let $pc = D(K_e, epc)$ and $m' = Dec(M'|pc)$
3. If $m' = \perp$ then **Return:** 0
4. If $V(K_a, m', subtag) = 1$ then **Return:** 1 else **Return:** 0.
```

We can prove the following

**Theorem 1.** Let $d$ denote the Hamming distance, let $n$ be the length of the input message for (Kg,Tag,Verify) and let (SEnc,SDec) a systematic error-correcting code that corrects up to $\delta$ errors and returns $\perp$ if more than $\delta$ errors happened, for some parameter $\delta$. Then (Kg,Tag,Verify) is an AMAC that satisfies the following properties:

1. $(d, p, \delta)$-approximate correctness for $p = 1$
2. $(d, \gamma, t', q', \epsilon')$-approximate security under the assumption that (KG$_a$,T,V) is a $(t, q, \epsilon)$-secure MAC, where $\gamma = \delta+1$, $t' = t - O(q \cdot time(\mathrm{D}) + time(\mathrm{SDec}))$, $q' = q$, $\epsilon' = \epsilon$, and $time(F)$ denotes the running time of function $F$.

3. $(d, t', q', \epsilon')$-weak preimage-resistance under the assumption that $(\mathrm{KG}_a, \mathrm{T}, \mathrm{V})$ is $(t_a, q_a, \epsilon_a)$-weakly preimage-resistant and $(\mathrm{KG}_e, \mathrm{E}, \mathrm{D})$ is a $(t_e, q_e, \epsilon_e)$-secure symmetric encryption scheme (in the real-or-random sense), where $q_e = 1$, $q' = q_a$, $\epsilon' \leq \epsilon_a + q_e\epsilon_e$, and $t' = \min(t_1, t_2)$, for $t_1 = t_a - O(q' \cdot (time(\mathrm{Enc}) + time(\mathrm{E}) + time(D_m)) + time(\mathrm{KG}_e))$, and $t_2 = t_e - O(q' \cdot (time(\mathrm{Enc}) + time(\mathrm{T}) + time(D_m)) + time(\mathrm{KG}_a))$.

The above theorem already provides AMACs with some useful properties, such as approximate correctness, approximate security and weak preimage-resistance. However, we note two facts that make this scheme not a definitely satisfactory solution: first, its tag length depends on the performance of the systematic code used, and can thus be significantly longer than regular MACs even for moderately large values of the parameter $\delta$; second, this scheme does not satisfy the stronger preimage resistance property. As we will see in Section 4, the latter is very desirable in order to construct a network biometric entity authentication scheme, a main application of AMACs in this paper. The scheme in Section 3.2 satisfies both efficiency of tag length (for any value of $\delta$) and the strong preimage-resistance property.

### 3.2 Our Main AMAC Construction

**Informal Description.** We explain the ideas behind this scheme in two steps. First, we explain how to construct a probabilistic universal one-way hash function and use it to guarantee that outputs from this hash function will have some additional distance-preserving properties. Second, we construct an approximately correct and secure MAC based on such a probabilistic universal one-way hash function.

We achieve a combination of distance-preserving properties and target collision resistance by making a universal one-way hash function probabilistic, and using the following technique. First, the message bits are xored with a pseudorandom string and pseudo-randomly permuted and then the resulting message is written as the concatenation of several equal-size blocks. Here, the size of each block could be the fixed constant size (e.g., 512 bits) of the input to compression functions (e.g., SHA) that are used as atomic components of practical constructions of universal one-way hash functions. Now multiple hashes are computed, each being obtained using the universal one-way hash function, using as input the concatenation of a different and small enough subset of the input blocks. Here, the choice of each subset is done using pseudo-random bits. Furthermore, each subset has the same size, depending on the length of the input and on the desired distance-preserving properties. The basic idea so far is that by changing the content of some blocks of the message, we only change a small fraction of the inputs of the atomic hashes and therefore only a small fraction of the outputs of those hashes will change.

Given this ‘probabilistic universal one-way hash function’, the tagging and verifying algorithm can be described as follows.

The tagging algorithm, on input a random key and a message, uses another value, which can be implemented as a counter incremented after each application (or a random value chosen independently at each application). Then the algorithm computes the output of the finite pseudo-random function on input such value and divides this output in two parts: the first part is a random key for the universal one-way hash function and the second part is a sequence of pseudo-random bits that can be used as randomness for the above described probabilistic universal one-way hash function. Now, the tagging algorithm can run the latter function to compute multiple hashes of the message. The tag returned is then the input to the finite pseudo-random function and the hashes.

The construction of the verifying algorithm is necessarily differently from the usual approach for exactly correct and secure MACs (where the verifying algorithm runs the tagging algorithm on input the received message and checks that its output is equal to the received tag), as this algorithm needs to accept the same tag for multiple messages. Specifically, on input the tag returned by the tagging algorithm, the verifying algorithm generates a key and pseudo-random bits for the probabilistic universal one-way hash function and computes the hashes of the received message exactly as the tagging algorithm does. Finally, the verifying algorithm checks that the received and the computed sequences of hashes only differ in a small enough number of positions.

**Formal Description.** Let $k$ be a security parameter, $t$ be an approximation parameter, and $c$ be a block size constant. We denote by $H = \{tcrh_K : K \in \{0,1\}^k\}$ a finite universal one-way hash function (also called 'target collision resistance function' in the literature), such that for each $K \in \{0,1\}^k$, $tcrh_K$ is a collision-intractable hash function. We denote by $F = \{f_K : K \in \{0,1\}^k\}$ a finite pseudo-random function. We now present our construction of an approximately-secure and approximately-correct MAC, which we denote as (Kg,Tag,Verify).

**Instructions for Kg:** generate a uniformly distributed $k$-bit key $K$

**Input to Tag:** a $k$-bit key $K$, an $n$-bit message $M$, parameters $p, \delta, \gamma$, a block size $1^c$ and a counter $ct$.

**Instructions for Tag:**

```
– Set $x_1 = \lceil n/2c\delta \rceil$ and $x_2 = \lceil 10 \log(1/(1-p)) \rceil$
– Set $(u|\pi|\rho|L) = f_K(ct)$, where $u \in \{0,1\}^k$, $L \in \{0,1\}^n$, and $\pi$ is a permutation of $\{0,1\}^n$
– Write $\pi(L \oplus M)$ as $M_1|\cdots|M_{\lceil n/c \rceil}$, where $|M_i| = c$ for $i = 1, \ldots, \lceil n/c \rceil$
– Use $\rho$ as randomness to randomly choose $x_1$-size subsets $S_1, \ldots, S_{x_2}$ of $\{1, \ldots, \lceil n/c \rceil\}$
– For $i = 1, \ldots, x_2$,
      let $N_i = M_{i_1}|\cdots|M_{i_{x_1}}$, where $S_i = \{i_1, \ldots, i_{x_1}\}$
      let $sh_i = tcrh_u(N_i)$
– Let $subtag = sh_1|\cdots|sh_{x_2}$
– Return: $tag = ct|subtag$.
– Set $ct = ct + 1$ and halt.
```

**Input to Verify:** parameters $\delta, \gamma$, a block size $1^c$, a $k$-bit key $K$, an $n$-bit message $M'$ and a string $tag$

**Instructions for Verify:**

- Write *tag* as $ct|sh_1|\cdots|sh_{x_2}$
- Set $x_1 = \lceil n/2c\delta \rceil$ and $x_2 = \lceil 10 \log(1/(1-p)) \rceil$
- Set $(u|\pi|\rho|L) = f_K(ct)$, where $u \in \{0,1\}^k$, $L \in \{0,1\}^n$, and $\pi$ is a permutation of $\{0,1\}^n$
- Write $\pi(L \oplus M')$ as $M'_1|\cdots|M'_{\lceil n/c \rceil}$, where $|M'_i| = c$ for $i = 1, \ldots, \lceil n/c \rceil$
- Use $\rho$ to randomly select $x_1$-size subsets $S'_1, \ldots, S'_{x_2}$ of $\{1, \ldots, \lceil n/c \rceil\}$
- For $i = 1, \ldots, x_2$,
  let $N'_i = M'_{i_1}|\cdots|M'_{i_{x_1}}$, where $S'_i = \{i_1, \ldots, i_{x_1}\}$
  let $sh'_i = tcrh_u(N'_i)$
- Check that $sh'_i = sh_i$, for at least $\alpha x_2$ of the values of $i \in \{1, \ldots, x_2\}$, for $\alpha = 1 - 1/2\sqrt{e} - 1/2e$.
- **Return:** 1 if all verifications were successful and 0 otherwise.

The above construction satisfies the following

**Theorem 2.** Let $d$ denote the Hamming distance, let $\delta, c, p$ be parameters. Then the above construction (Kg,Tag,Verify) is an AMAC satisfying the following properties.

1. $(d, p, \delta)$-approximate correctness
2. $(d, \gamma, t_A, q_A, \epsilon_A)$-approximate security under the assumption that F is a $(t_F, q_F, \epsilon_F)$-secure pseudo-random function and H is a $(t_H, q_H, \epsilon_H)$-target-collision-resistant hash function, where $\gamma = 2\delta$, $\epsilon_A \leq p_1 \leq \epsilon_F + 2\epsilon_H \cdot q_A + 2(1-p)$, $q_A = q_F \geq 1$, $q_H = \lceil 10 \log(1/(1-p)) \rceil$, and $t_A = \min(t_{A,1}, t_{A,2})$, where $n$ is the length of the message, $c$ is a block size constant, $ct$ is the counter input to algorithm Tag, $time(g; x)$ denotes the time required to compute function $g$ on inputs of size $x$, and
   - $t_{A,1} = t_F - O(q_A(n(\log n + \log(1/(1-p))) + \log(1/(1-p)) + time(h_u; n/2c\delta))$
   - $t_{A,2} = t_H - O(n(\log n + \log(1/(1-p))) + time(f_K; |ct|))$.
3. $(d, t', q', \epsilon')$-strong preimage resistance under the assumption that for each $K \in \{0,1\}^k$, function $h_K$ is $(t, \epsilon)$-collision resistant, where $\epsilon' \leq \epsilon$, and $t' = t - O(time(\text{Tag}; n))$.

**Remarks.** Our scheme is quite simple to implement and our implementation experience required very small effort. We note that in practice the families H and F can be implemented using well-known keyed cryptographic hash functions (e.g., UMAC [1] or other constructions cited in there) and well-known block ciphers (e.g., AES).

The length of the tag returned by algorithm Tag is $x_2 \cdot c$, where $x_2 = 10 \log(1/(1-p))$, and $c$ is the length of the output of the universal one-way hash function. (In practice, this value could be smaller, but it would require a more involved security analysis.) We note that $c$ is constant with respect to $n$, and acceptable settings of parameter $p$ can lie anywhere in the range $[1 - 1/2^{(\log n)^{1+\epsilon}}, 1]$, for any constant $\epsilon > 0$. Therefore the length of the tag returned by the scheme can be as small as $10c(\log n)^{1+\epsilon}$; most importantly, this holds for *any* value of parameter $\delta$. The tag length remains much shorter than the message even for much larger settings of $p$; for instance, if $p = 1 - 2^{-\sqrt{n}}$, the tag length becomes $O(\sqrt{n})$.

### 3.3 Properties of Our Main Construction

We now discuss the properties mentioned in Theorem 2. As the strong preimage resistance property immediately follows from the collision resistance of functions from $H$, we now focus on proving the approximate correctness and approximate security properties.

Approximate Correctness. Assume $d(M, M') \leq \delta$. First, we assume for simplicity that $f_K$ is a random function. Then, for $i = 1, \ldots, x_2$, define random variable $X_i$ as equal to 1 if $sh_i \neq sh_i'$ or 0 otherwise. Furthermore, we denote by $N_i$ and $M_{i_1}, \ldots, M_{i_{x_1}}$ (resp., $N_i'$ and $M_{i_1}', \ldots, M_{i_{x_1}}'$) the values used in the 5th step of algorithm Tag on input $M$ (resp., $M'$). Then it holds that

$$\begin{aligned}
a &= \text{Prob}\,[\,X_i = 1\,] \\
&\leq 1 - \text{Prob}\,[\,N_i = N_i'\,] \\
&= 1 - (\text{Prob}\,\left[\,M_{i_1} = M_{i_1}'\,\right])^{n/2\delta} \\
&\leq 1 - ((n/c - \delta)/(n/c))^{n/2c\delta} = 1 - (1 - c\delta/n)^{n/2c\delta} \;\leq\; 1 - 1/\sqrt{e},
\end{aligned}$$

where the first inequality follows from the definition of $X_i$ and from how $sh_i, sh_i'$ are computed; the second equality follows from the definition of $N_i, N_i'$; and the second inequality follows by observing that $M$ and $M'$ differ in at most $\delta$ blocks, and that blocks $M_i, M_i'$ are uniformly and independently chosen among all blocks in $\pi(M), \pi(M')$, respectively, as so are subsets $S_i, S_i'$. We obtain that $a - \alpha = (\sqrt{e} - 1)/2e$. Since $X_1, \ldots, X_{x_2}$ are independent and identically distributed, we can apply a Chernoff bound and obtain that

$$\text{Prob}\left[\sum_{i=1}^{x_2} X_i \;<\; \alpha x_2\right] \;\leq\; e^{-2(a-\alpha)^2 x_2} \;\leq\; 1 - p,$$

which implies that algorithm Verify returns 1 with probability at least $p$. Note that the assumption that $f_K$ is a random function can be removed by only subtracting a negligible factor to $p$, as otherwise the test used by algorithm Verify can be used to contradict the pseudorandomness of $F$.

Approximate Security. The proof for this (only sketched here) requires the definition of four probability experiments that slightly differ from each other. We assume that the requirement of $(d, \gamma, t, q, \epsilon)$-approximate security is not satisfied and reach some contradiction.

Experiment 1 is precisely the experiment in the definition of approximate security. We denote by $p_1$ the probability that experiment 1 is successful; our original assumption implies that $p_1 > \epsilon$.

Experiment 2 differs from experiment 1 only in that *Adv* queries a finite random function $r$ rather than a finite pseudo-random function Tag. Denoting as

$p_2$ the probability that experiment 2 is successful, we can prove that $p_2 - p_1 \leq \epsilon_F$, or otherwise *Adv* can be used to violate the assumption that $F$ is a $(t_F, q_F, \epsilon_F)$-secure pseudo-random function.

Experiment 3 is a particular case of experiment 2; specifically, it is successful when experiment 2 is and the adversary returns a tag with the same counter as in a tag previously returned by the oracle. We distinguish two cases, according to whether the following condition is true or not: all $i \in \{1, \ldots, x_2\}$ such that $sh_i = sh'_i$ are associated with values $N_i, N'_i$ such that $N_i = N'_i$. If the condition does not hold, then this means that *Adv* found two distinct preimages $N_i, N'_i$ of the same output under $tcrh_u$ and therefore *Adv* can be used to violate the assumption that $H$ is a $(t_H, q_H, \epsilon_H)$-target collision resistant hash function. If the condition holds, then this means that a large number of subsets $S_i$ 'missed' all $\gamma = 2\delta$ bits where $M$ and $M'$ differ. By using a Chernoff bound argument dual to that used in the proof of the approximate correctness property, we derive that this happens with probability at most $1 - p$. We denote as $p_3$ the probability that experiment 3 is successful, and, from the above two cases, obtain that $p_3 \leq \epsilon_H \cdot q_A + 1 - p$.

Experiment 4 is a particular case of experiment 2; but it considers the case complementary to the case in experiment 3. Specifically, it is successful when experiment 2 is and the adversary returns a tag with a counter different from those in all tags previously returned by the oracle. The analysis of this case goes on very similarly as for experiment 3, with the only difference that in the step similar to the proof of the approximate correctness property, we use the fact that the messages $M, M'$ are xored with pseudo-random strings. We obtain that $p_4 < p_3$, where by $p_4$ we denote the probability that experiment 4 is successful.

We conclude the analysis by using the obtained inequalities: $p_1 - p_2 \leq \epsilon_F$, $p_2 \leq p_3 + p_4$, $p_3 \leq \epsilon_H \cdot q_A + 1 - p$, and $p_4 < p_3$; and therefore obtaining that $\epsilon_A \leq p_1 \leq \epsilon_F + 2\epsilon_H \cdot q_A + 2(1 - p)$.

## 4 Biometric Entity Authentication

We present a model for the design and analysis of biometric entity authentication (BEA) schemes, and show that two simple constructions based on AMACs can be proved secure in our model under standard assumptions on cryptographic tools and biometric distribution.

**Our Model.** There is a server $S$ and several users $U_1, \ldots, U_m$, where the server has a biometric storage file $bsf$ and each user $U_i$ is associated with a biometric $b_i$, a reader $R_i$ and a computing unit $CU_i$, for $i = 1, \ldots, m$. We define a (non-interactive) BEA scheme between user $U_i$ and $S$ as the following two-phase protocol. The first phase is an *initialization phase* during which user $U_i$ and $S$ agree on various parameters and shared keys and $S$ stores some information on $bsf$. The second phase is the *authentication phase*, including the following steps. First, user $U_i$ inputs her biometric $b_i$ to the reader $R_i$, which extracts some feature information $fb_{i,t}$ (this may be a sketched version of the original biometric

$b_i$) and returns a measurement $mb_{i,t}$, where $t$ here represents the time when $R_i$ is executed. (Specifically, the reader may return a different value $mb_{i,t}$ for each different time $t$, on input the same $b_i$.) Then the computing unit $CU_i$, on input $mb_{i,t}$ sends an authenticating value $ab_{i,t}$ to the server, that, using information stored during the initialization phase, decides whether to accept $ab_{i,t}$ as a valid value for user $U_i$ or not.

The *correctness* requirement for a BEA scheme states that the following happens with high probability: after the initialization phase is executed between $U_i(b_i)$ and $S$, if, for some $t$, $mb_{i,t} = R_i(b_i)$, and $ab_{i,t} = CU_i(mb_{i,t})$ then $S$ accepts pair $(U_i, ab_{i,t})$.

An *adversary Adv* tries to attack a BEA scheme by entering a biometric $b_j$ into a reader $R_i$, and, before doing that, can have access to several and different resources, according to which parties it can corrupt (i.e., noone; users $U_j$, for $j \neq i$; server $S$; etc.), and which communication lines or storage data he has access to (i.e., none; the communication lines containing any among $mb_{i,t}, ab_{i,t}$; the biometric storage file $bsf$; the server's secret keys; user $U_i$'s secret keys, etc.). The *security* requirement for a BEA scheme states that after the initialization phase is executed between $U_i(b_i)$ and $S$, for $i = 1, \ldots, m$, the probability that an efficient adversary *Adv* can input his biometric $b_j$ into a reader $R_i$, for $i \neq j$, and make $S$ accept the resulting pair $(U_i, ab^j_{i,t})$, is negligible.

We are now ready to show two simple BEA constructions given any AMAC scheme with certain properties (in fact, not necessarily as strong as those required by Definition 1). The first construction is for *local* BEA; that is, the adversary has no access to the measurements $mb_{i,t}$ and the user can send them in the clear to the server. Local BEA is comparable, in terms of both functionality and security, to well-known password-based authentication schemes in non-open networks. The second construction is for *network* BEA; that is, the message sent from a user to a server during the authentication phase can travel through an open network. Network BEA should be contrasted, in terms of both functionality and security, to password-based authentication schemes in open networks; in particular, we will show that our scheme does not require a user to send over an open network (not even in encrypted form) a reading of her biometric. Both constructions necessarily make an assumption on the distribution of biometric that we now describe.

**A Basic Assumptions on Biometrics.** We assume that there exist a distance function $d$, appropriate parameters $\delta < \gamma$, and an efficiently computable measurement M of biometrics such that: (1) for each individual with a biometric $b$ with feature information $fb(t)$ at time $t$, and for any times $t_1, t_2$, it holds that $d(\mathrm{M}(fb(t_1)),\mathrm{M}(fb(t_2))) \leq \delta$; (2) for any two individuals with biometrics $b_1, b_2$, with feature information $fb_1(t), fb_2(t)$ at time $t$, respectively, and for any times $t_1, t_2$, it holds that $d(\mathrm{M}(fb(t_1)),\mathrm{M}(fb(t_2))) \geq \gamma$. We refer to this as the *Biometric Distribution Assumption* (BD Assumption). We note that biometric entity authentication (in any model) inherently relies on similar assumptions.

**A Construction for Local BEA.** Informally, the first construction consists of the user sending the reading of her biometric to the server, that checks it against the previously stored AMAC tag of a reading done at initialization phase. More formally, let (Kg,Tag,Verify) denote an AMAC scheme. Then the BEA scheme lAmacBEA goes as follows. During the initialization phase, user $U_i$ sends $ab_{i,t_0}$ to the server $S$, that stores $tag_0$ =Tag($k, ab_{i,t_0}$) in the $bsf$ file. During the authentication phase, at time $t_1$, user $U_i$ inputs $b_i$ into the reader $R_i$, that returns $mb_{i,t_1}$; the latter is input to $CU_i$ that returns $ab_{i,t_1} = mb_{i,t_1}$; finally, pair $(U_i, ab_{i,t_1})$ is sent to $S$. On input pair $(U_i, ab_{i,t_1})$, server $S$ computes Verify($k, ab_{i,t_1}, tag_0$) and accepts $U_i$ if and only if it is equal to 1.

We can prove the following

**Theorem 3.** Under the BD assumption, if (Kg,Tag,Verify) is an AMAC scheme then the construction lAmacBEA is a BEA scheme satisfying the above correctness and security requirement against efficient adversaries that can corrupt up to all users $U_j$ but one. Furthermore, if scheme (Kg,Tag,Verify) is weakly preimage-resistant then the construction lAmacBEA satisfies security against efficient adversaries that have also access to the biometric storage file $bsf$.

**A Construction for Network BEA.** Informally, the second construction modifies the first construction by having the user compute the AMAC tag over the reading of her biometric; the AMAC tag is then sent to the server that can check it against the previously stored AMAC tag of a reading done at initialization phase. Also, we assume for simplicity that the channel between each user and the server is properly secured (using standard encryption, authentication and time-stamping techniques). More formally, let (Kg,Tag,Verify) denote an AMAC scheme with strong preimage resistance. Then the BEA scheme nAmacBEA goes as follows. During the initialization phase, user $U_i$ inputs her biometric $b_i$ into reader $R_i$, that returns $mb_{i,t_0}$; the latter is input to $CU_i$ that returns and sends $ab_{i,t_0}$ =AMAC($k, mb_{i,t_0}$) to $S$; finally, $S$ stores $ab_{i,t_0}$ into $bsf$. The authentication phase is very similar to the identification phase; specifically, user $U_i$ computes $ab_{i,t_1}$ in the same way, and pair $(U_i, ab_{i,t_1})$ is sent to $S$, that computes Verify($k, ab_{i,t_1}, ab_{i,t_0}$) and accepts $U_i$ if and only if it is equal to 1.

We can prove the following

**Theorem 4.** Under the BD assumption, if (Kg,Tag,Verify) is an AMAC scheme, then the construction nAmacBEA is a BEA scheme satisfying the above correctness and security requirement against efficient adversaries that can corrupt up to all users $U_j$ but one and have access to the communication lines containing $mb_{i,t}, ab_{i,t}$. Furthermore, if scheme (Kg,Tag,Verify) is strongly preimage-resistant then the construction nAmacBEA satisfies security against efficient adversaries that additionally have access to the biometric storage file $bsf$, and to the server's secret keys.

We note that the first AMAC construction in Section 3 is weakly preimage-resistant and therefore suffices for the AMAC scheme required by Theorem 3. Furthermore, the second AMAC construction in Section 3 is strongly preimage-resistant and can therefore be used to construct the AMAC scheme required by Theorem 4.

**Disclaimer.** The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government.

## References

1. J. Black, S. Halevi, H. Krawczyk, T. Krovetz, and P. Rogaway, *UMAC: Fast and Secure Message Authentication*, Proc. of CRYPTO '99, Springer.
2. X. Boyen, *Reusable Cryptographic Fuzzy Extractors,* Proc. of 11th ACM Conference on Computer and Communication Security, 2004
3. G. Davida, Y. Frankel, and B. Matt, *On Enabling Secure Application through Off-Line Biometric Identification,* Proc. of 1998 IEEE Symposium on Research in Security and Privacy
4. G. Di Crescenzo, R. F. Graveman, G. Arce and R. Ge, *A Formal Security Analysis of Approximate Message Authentication Codes,* Proc. of the 2003 CTA Annual Symposium, a US Dept. of Defense publication.
5. Y. Dodis, L. Reyzin, and A. Smith, *Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data,* Proc. of Eurocrypt 2004, Springer.
6. R. F. Graveman and K. Fu, *Approximate Message Authentication Codes,* Proc. of 3rd Annual Symposium on Advanced Telecommunications & Information Distribution Research Program (ATIRP), 1999
7. P. Indyk, R. Motwani, P. Raghavan, and S. Vempala, *Locality-Preserving Hashing in Multidimensional Spaces,* Proc. of ACM STOC 97
8. A. Jain, R. Bolle, and S. Pankanti, eds. Biometrics: Personal Identification in a Networked Society, Kluwer Academic Publishers, 1999.
9. A. Juels and M. Sudan, *A Fuzzy Vault Scheme,* Proc. of IEEE International Symposium on Information Theory, 2002
10. A. Juels and M. Wattenberg, *A Fuzzy Commitment Scheme,* Proc. of 6th ACM Conference on Computer and Communication Security, 1999
11. N. Linial and O. Sasson, *Non-Expansive Hashing,* Proc. of ACM STOC 96
12. E. Martinian, B. Chen and G. Wornell, *Information Theoretic Approach to the Authentication of Multimedia,* Proc. of SPIE Conference on Electronic Imaging, 2001
13. E. Martinian, B. Chen and G. Wornell, *On Authentication With Distortion Constraints,* Proc. of IEEE International Symposium on Information Theory, 2001
14. M. Naor and M. Yung, *Universal one-way hash functions and their cryptographic applications,* Proc. of ACM STOC 89.
15. S. Prabhakar, S. Pankanti, and A. Jain, *Biometric Recognition: Security and Privacy Concerns,* IEEE Security and Privacy Magazine, vol. 1, n. 2, March 2003.
16. B. Schneier, *Inside Risks: The Uses and Abuses of Biometrics,* Communications of the ACM, vol. 42, no. 8, pp. 136, Aug. 1999.
17. L. Xie, G. R. Arce, and R. F. Graveman, *Approximate Image Message Authentication Codes,* IEEE Transactions on Multimedia, vol. 3, June 2001.

# Analysis of a Multi-party Fair Exchange Protocol and Formal Proof of Correctness in the Strand Space Model

Aybek Mukhamedov[1], Steve Kremer[2], and Eike Ritter[1]

[1] School of Computer Science,
University of Birmingham, UK
{A.Mukhamedov, E.Ritter}@cs.bham.ac.uk
[2] Laboratoire Spécification et Vérification,
CNRS UMR 8643 & INRIA Futurs projet SECSI & ENS Cachan, France
kremer@lsv.ens-cachan.fr

**Abstract.** A multi-party fair exchange protocol is a cryptographic protocol allowing several parties to exchange commodities in such a way that everyone gives an item away if and only if it receives an item in return. In this paper we discuss a multi-party fair exchange protocol originally proposed by Franklin and Tsudik, and subsequently shown to have flaws and fixed by González and Markowitch. We identify flaws in the fixed version of the protocol, propose a corrected version, and give a formal proof of correctness in the strand space model.

## 1 Introduction

The problem of fairly exchanging electronic goods over a network has gained increasing importance. In a fair exchange several entities want to exchange their goods in a way that none of them will be fooled, i.e. no entity will give away his own item without also getting the other expected item. The problem arises because of an inherent asymmetry: no entity wants to be the first one to send out his item because another entity could refuse to do so after having obtained the first entity's item. In 1980, Even and Yacobi [11] showed that no deterministic contract-signing protocol—contract signing is a special case of fair exchange where digital signatures on a contract are exchanged—exists, without the participation of a trusted third party. A simple solution consists in using a trusted party ($T$) as an intermediary. Signers send their respective contracts to $T$, who first collects the contracts and then distributes them among the signers. Other solutions include randomized protocols as well as protocols based on gradual information exchange. More recently, the so-called *optimistic* approach was introduced in [3, 7]. The idea is that $T$ intervenes only when a problem arises, i.e. some entity is trying to cheat or a network failure occurs at a crucial moment during the protocol. Such a trusted party is called *offline*. However, these protocols are less attractive when the group of entities involved in the exchange is large, because the risk of $T$ intervening is increased.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 255–269, 2005.

© Springer-Verlag Berlin Heidelberg 2005

Most protocols that have been proposed in literature are *two-party* protocols. More recently, different kinds of *multi-party fair exchange* have been considered. In [12], Franklin and Tsudik propose a classification. One can distinguish between single-unit and multi-unit exchanges. Moreover different exchange topologies are possible. While two-party non-repudiation, certified e-mail, contract signing, or more generally fair exchange protocols are very similar, in the case of a group protocol, the different topologies corresponding to particular kinds of fair exchange increase the diversity of the protocols.

Bao et al. [4] and Franklin and Tsudik [12] concentrated on a ring topology. Each entity $e_i$ $(0 \leq i \leq n-1)$ desires an item (or a set of items) from entity $e_{i \boxminus 1}$ and offers an item (or a set of items) to entity $e_{i \boxplus 1}$, where $\boxplus$ and $\boxminus$ respectively denote addition and subtraction modulo $n$. Although the ring topology seems to be the simplest one for protocol design, Gonzalez and Markowitch [14] show that Franklin and Tsudik's protocol [12] is not fair and propose a new protocol.

A ring topology is not sound in non-repudiation or certified e-mail protocols: it does not make sense that one entity receives an e-mail and a distinct entity sends the corresponding receipt. The most natural generalization seems to be a star topology, i.e. a one-to-many protocol, where one entity sends a message to $n-1$ receiving entities who respond to the sender. Kremer and Markowitch [15] proposed the first multi-party non-repudiation protocols with both online and offline $T$. The main motivation for these protocols is a significant performance gain with respect to $n$ two-party protocols. Afterwards, Onieva et al. [17] extended their work with online $T$, in order to permit the sending of different messages to each entity.

Another topology is the more general matrix topology, where each entity may desire items from a set of entities and offer items to a set of entities. Such protocols have been proposed by Asokan et al. on synchronous networks in [2]. However, asynchronous networks are more realistic.

Multi-party contract-signing protocols [1, 13, 6, 5] give raise to yet another topology. The objective is that each signer sends its signed contract on a given contract text to all other signers and that each signer receives all other signers' contract. This corresponds to a complete graph.

In the remaining of the paper we focus on ring topologies.

It is known that security protocols are error-prone and the need for applying formal methods to security protocols has been widely recognised. In this paper, we are analysing the ring fair exchange protocol proposed by Franklin and Tsudik [12] and the "corrected" version by González and Markowitch [14]. As we will see even the version of González and Markowitch contains a subtle flaw, which exploits properties such as commutativity and homomorphism. We come up with a fixed version and give a proof of correctness in the strand space model [21]. Proving this protocol correct introduces several challenges. Firstly, we are dealing with a group protocol, i.e. the number of participants is a parameter. Secondly, the protocol relies on algebraic properties which need to be abstracted properly and included in a classic Dolev-Yao like model [10] used for our analysis. Finally, the property that we are going to prove is *fairness*, which has been studied much less than authentication or secrecy.

There have been applications of formal methods to two-party fair exchange protocols, including automated analysis using model-checkers [16, 20], as well as hand proofs in frameworks, such as CSP [19] and multi-set rewriting [8]. This list is far from being complete. Recently, Chadha et al. [9] have successfully used the model-checker Mocha to discover errors in a multi-party contract signing protocol proposed by Garay and MacKenzie. To the best of our knowledge, this is the only formal analysis of multi-party fair exchange protocols. While a model-checker is very useful to discover errors, it does not give a proof of correctness because the analysed model needs to be considerably simplified. The advantage of a pen and paper proof is that much less simplification is required. In [18], Pereira and Quisquater used the strand space model to prove a generic insecurity result for a family of group authentication protocols. This work already demonstrates that the strand space model can be used in the case of group protocols and can be extended to model algebraic properties, e.g. Diffie-Hellman exponentiation in their case.

Our paper will be organised as follows. In the next section we present Franklin and Tsudik's protocol [12], as well as González and Markowitch's attack and fix [14]. Then we analyse this protocol, show some weaknesses and a more fundamental attack exploiting commutativity and homomorphism of the one way function used in the protocol. We go on to present how the strand space model has been adapted to our needs. Finally, we prove correctness of our fixed protocol in the strand space model and conclude.

## 2 Description of the Protocols

In this section we describe the protocol presented by Franklin and Tsudik in [12]. We also present an attack, discovered by González and Markowitch [14] four years after the original protocol was published, as well as González and Markowitch's fixed version of the protocol. As we will see in section 3 even the fixed version of the protocol is flawed. This fact emphasizes once more the difficulty of designing correct protocols, above all group protocols, and the need for formal methods.

### 2.1 Notations

We use the following notations when describing the protocols:

- $\mathcal{P}$: the set of $n$ participants;
- $P_i$: the $i$th participant of the protocol ($1 \leq i \leq n$);
- $m_i$: the commodity $P_i$ wishes to exchange;
- $R_i$: a random number chosen by $P_i$.

To avoid notational clutter, we suppose that all operations on subscripts are performed modulo $n$. The protocol also relies on a homomorphic one-way function $f$, i.e. $f(x_1) \cdot f(x_2) = f(x_1 \cdot x_2)$, and a function with $n$ arguments $F_n$, such that $F_n(x_1, f(x_2), \ldots, f(x_n)) = f(x_1 \cdot x_2 \cdot \ldots \cdot x_n)$. In [12], Franklin and Tsudik propose $f(x) = x^2 \pmod N$ and $F_n(x_1, \ldots, x_n) = x_1^2 \cdot x_2 \cdot \ldots \cdot x_n$, where $N$ is an RSA modulus.

### 2.2 The Franklin-Tsudik Protocol

A short, informal description of the protocol for $P_i$ is given in protocol 1. Remember that the aim of the protocol is a fair exchange on a ring topology: participant $P_i$ has to send its commodity $m_i$ to $P_{i+1}$ and will receive commodity $m_{i-1}$ from $P_{i-1}$ in exchange.

**Protocol 1** Franklin-Tsudik multi-party fair exchange protocol for $P_i$

```
1. $P_i \rightarrow P_{i+1}$: $R_i$
2. $P_{i-1} \rightarrow P_i$: $R_{i-1}$
3. $P_i \rightarrow T$: $A_i, C_i, f(R_i)$
   where $A_i = F_n(m_i, \langle f(m_k) \rangle_{k \neq i})$ and $C_i = m_i \cdot {R_i}^{-1}$
4. $T \rightarrow P_i$: $\mathcal{C}$
   where $\mathcal{C} = \{C_i \mid 1 \leq i \leq n\}$
```

At the end of a preliminary set-up phase it is assumed that:

- the identities of all participating parties are known;
- all participants agree on $T$ and the functions $f$ and $F_n$;
- the descriptions of the items to be exchanged, $f(m_i)$, are public.

Moreover, all channels are assumed to be private and authentic.

The protocol proceeds as follows. In a first message $P_i$ sends a random number $R_i$ to $P_{i+1}$ and receives another random number $R_{i-1}$ from $P_{i-1}$ in the second message. Then $P_i$ contacts the trusted party $T$ by sending $A_i = F_n(m_i, \langle f(m_k) \rangle_{k \neq i})$, $C_i = m_i \cdot {R_i}^{-1}$ and $f(R_i)$. The trusted party $T$ waits until it has received a message from every participant $P_i$. It then performs two checks:

- equality of all $A_i$'s;
- $F_{n+1}(\prod_{1 \leq i \leq n} C_i, f(R_1), \ldots, f(R_n))$ is equal to $f(\prod_{1 \leq i \leq n} m_i)$, which should be equal to $A_i$.

If both checks succeed then $T$ sends $\mathcal{C}$, the set of all $C_j$'s, via broadcast to each $P_i$. Finally, $P_i$ can check for each $C_j$ in $\mathcal{C}$ whether $f(C_j \cdot R_{i-1}) = f(m_{i-1})$. If the check succeeds for $C_j$, $P_i$ computes $m_{i-1} = C_j \cdot R_{i-1}$.

### 2.3 Attack and "Fix" by González and Markowitch

The checks performed by $T$ in the above described protocol are justified by the authors of the original protocol to "*establish that all $m_i$'s and $R_i$'s are consistent and have been properly committed*", and "*coherence of all $m_i$ values*". However, one may notice from the protocol that the former check does not guarantee consistency of $R_i$'s. As a result, González and Markowitch in [14] found an attack that exploited this weakness: a dishonest participant $P_i$ supplies different values of $R_i$ to $P_i$ and the trusted server $T$. The checks performed by $T$ still hold, while $P_{i+1}$ fails to receive correct multiplicative inverse from $T$ and hence is unable to recover $m_i$.

González and Markowitch [14] suggest a revised protocol. They assume a preliminary setup phase which is similar to the one described above. However, they add a label $\ell$ to identify a protocol run, obtained by applying a one-way hash function to $\mathcal{P}$ and the set of $m_i$'s. Knowledge of $\ell$ is also assumed after the setup phase. In their presentation, the authors drop the hypothesis of private and authentic channels, but sign and encrypt each of the messages explicitly. In our presentation here, for the sake of simplicity, we assume private and authentic channels, as signatures and encryption is only one possible way of achieving these goals. A short description of the protocol is given in protocol 2. The main difference with the previous protocol is that $P_i$ includes $f(R_{i-1})$ in his message when contacting $T$. This change should prevent the previous attack. The authors give an informal argument of its correctness. Unfortunately, as we will show, not all attacks are avoided.

**Protocol 2** González-Markowitch multi-party fair exchange protocol for $P_i$

1. $P_i \rightarrow P_{i+1}$: $\ell, R_i$
2. $P_{i-1} \rightarrow P_i$: $\ell, R_{i-1}$
3. $P_i \rightarrow T$: $\ell, A_i, C_i, f(R_i), f(R_{i-1})$
   where $A_i = F_n(m_i, \langle f(m_k)\rangle_{k\neq i})$ and $C_i = m_i \cdot {R_i}^{-1}$
4. $T \rightarrow P_i$: $\ell, \mathcal{C}$
   where $\mathcal{C} = \{C_i \mid 1 \leq i \leq n\}$

# 3 Analysis

## 3.1 Implicit Assumptions

In both papers [12] and [14], the properties of commodities exchanged are not specified and ambiguous. As a result, no (explicit) restrictions are put on dishonest agents against copying and distributing them. In particular, an honest agent $P$ is left in an unfair state if after the first cycle a dishonest agent $\widetilde{P}$, who received $P$'s item, decides to distribute this item to others.

The above described weakness demonstrates that it is impossible to guarantee fairness for an exchange of arbitrary items. Therefore, we explicitly need to make the following assumptions about the properties of commodities:

- commodities are token-based, i.e. an agent loses ownership of $m_i$ when he gives it away and there is at most one owner of $m_i$ at any time;
- or, a commodity can only be issued by an originator and used by the party it is issued to.

These properties are outside the scope of the protocol and need to be ensured by other techniques. However, we believe that they need to be explicitly stated.

### 3.2 Replay Attacks

In [14], the label $\ell$ was introduced to serve as an identifier of a protocol session. However, the resulting label is not unique, which allows the following attack.

Suppose $P_i$, $P_k$ and $\widetilde{P}$ decide to perform a cyclic exchange, where $P_i$ sends an item to $P_k$ in return for an item from $\widetilde{P}$. After the setup phase, $\widetilde{P}$ observes all messages sent by $P_i$. As protocol messages are assumed to be private and authentic, the intruder can neither elicit their components nor claim to an other party to be their originator; he also can't replay the intercepted message containing a nonce to other honest parties, as it contains the recipient's identity. In this run $\widetilde{P}$ may or may not send a message to $T$, who eventually stops the protocol after some pre-defined amount of time if the latter is chosen.

Suppose that the same cyclic exchange takes place after $\widetilde{P}$ retrieves a nonce $R_i$. The "label" corresponding to this run will be the same as in the previous one. Channels are assumed to be resilient [14], which means that messages can get delayed by an arbitrary but finite amount of time. Therefore, after the setup phase the intruder can delay the nonce $R_i{}'$ from $P_i$ intended to $P_k$, long enough such that: $(i)$ he can replay the message containing $R_i$ from the previous run to $P_k$; $(ii)$ $P_k$ sends his message to $T$. $\widetilde{P}$ also replays the other messages of $P_i$ to $T$, as well as his previous message to $T$ (except $f(R'_k)$ substituted for $f(R_k)$), but sends the unmatching nonce to $P_i$ afterwards. All checks succeed and $T$ broadcasts $\mathcal{C}$. $P_k$ gets $m_i$ and $\widetilde{P}$ gets $m_k$ and $m_i$; $P_i$ does not get $\widetilde{m}$ and even if she acquires $\widetilde{P}$'s message, she is still in an unfair state as the only expected recipient of her message is supposed to be $P_k$.

In a simpler version of the above attack dishonest agents simply send new nonces in a replay of the protocol to leave regular agents in an unfair state.

To conclude, in the analysis of replay weaknesses we need to make a stronger assumption on the setup phase: an intruder cannot simply via replaying messages of the setup phase make $T$ to vacuously believe that another exchange is initiated, and all participants of the exchange know true identities of the others, as otherwise, replay attacks are trivial.

### 3.3 Arithmetic Attack

We now present a more fundamental and interesting attack. The protocol presented in [14] (protocol 2) ensures consistency between the nonces sent by $P_i$ to $P_{i+1}$ with respect to $T$. However, it does not address consistency among $C_i$ and $f(R_i)$: it is not ensured that the former contains a multiplicative inverse of $R_i$. The second check performed by $T$,

$$F_{n+1}(\prod_{1\leq i\leq n} C_i, f(R_1), \ldots, f(R_n)) \stackrel{?}{=} f(\prod_{1\leq i\leq n} m_i)$$

only verifies if an agent has supplied $m_i$ in $C_i$ which is consistent with expectations of other agents[1]. As a result, fairness can be broken by two non-contiguous[2] cooperating dishonest agents, who receive desired exchange messages while not revealing theirs. The following details the attack.

The protocol proceeds as described in [14], except that two dishonest non-contiguous agents $\widetilde{P_i}$ and $\widetilde{P_k}$ send $C_i = m_i \cdot {R_k}^{-1}$ and $C_k = m_k \cdot {R_i}^{-1}$ to $T$, respectively. All the checks of $T$ will succeed, including

$$\begin{aligned} &F_{n+1}(\textstyle\prod_{1\leq i\leq n} C_i, f(R_1), \ldots, f(R_n)) \\ &= F_{n+1}(m_1 R_1^{-1} \cdot \ldots \cdot m_i R_k^{-1} \cdot \ldots \cdot m_k R_i^{-1} \cdot \ldots m_n R_n^{-1}, f(R_1), \ldots, f(R_n)) \\ &= f(\textstyle\prod_{1\leq i\leq n} m_i). \end{aligned}$$

However, honest parties $P_{i+1}$ and $P_{k+1}$ will not receive correct multiplicative inverses. Thus, they are not able to recover $m_i$ and $m_k$, respectively. This attack is weak in the sense that $P_{i+1}$ could recover $m_k$, i.e. a secret of $\widetilde{P_i}$'s conspirator. However, $P_{i+1}$ would need to test all possible item descriptions which is not foreseen in the protocol. Moreover, there exists a stronger attack, where this is not possible: $\widetilde{P_i}$ could send $C_i = m_i \cdot m_k$ and $\widetilde{P_k}$ would send $C_k = R_i^{-1} \cdot R_k^{-1}$.

This attack was first discovered when trying to prove correctness of the González-Markowitch protocol in the strand space model. The failure of the proof hinted directly towards the above attack and illustrates that strand spaces are also useful to discover new flaws.

## 4 Formal Model

We use the well-studied strand space model [21] to represent and prove correct a fixed version of the above analysed protocol. Basic definitions and facts about strand spaces are recalled in Appendix A. In this section we only provide intuitions about strand spaces without giving the formal definitions.

We start defining the set of possible messages. We need to extend the sets of messages classically used in strand spaces in order to deal with algebraic properties, such as the inverses, used in the protocols presented before. This way of abstracting algebraic properties over groups is inspired by [18], where the authors handle Diffie-Hellman exponentiations.

**Definition 1.** *Let:*

1. $\mathsf{T}$ *be the set of texts;*
2. $\mathsf{R}$ *be the set of random values used in the protocol;*
3. $\mathsf{M}$ *be the set of commodities exchanged in the protocol;*
4. $\mathsf{C}$ *be the set of ciphertexts;*

[1] Due to associativity and commutativity of multiplication it does not ensure any further consistency.

[2] Otherwise, one of them will not receive a secret.

5. $(\mathbf{G}\quad,\cdot)$ *be the commutative group freely generated from elements in* $\mathsf{R}$ *and* $\mathsf{M}$*; the unit element is denoted* $1$*;* $\underbrace{g\cdot\ldots\cdot g}_{n\times}$ *is denoted* $g^n$ *and* $g^0=1$*. Similarly,* $\mathsf{C}$ *freely generates group* $(\mathbf{G_c},\cdot)$ *that is also abelian. Let* $\mathbf{G}=\mathbf{G}\quad\cup\mathbf{G_c}$*;*
6. $\mathsf{inv}:\mathbf{G}\to\mathbf{G}$ *be an injective function computing the inverse element, such that* $\mathsf{inv}(x)\cdot x=1$*.* $inv(x)$ *is also denoted* $x^{-1}$*;*
7. $\mathsf{f}:\mathbf{G}\quad\to\mathbf{G_c}$ *be a homomorphic one-way function;*
8. $(\mathsf{A},\cdot)$ *be the free monoid generated by* $\mathbf{G}$*. It represents the free term algebra of our protocol;*
9. $\mathsf{A}'$ *be the set of elements disjoint from the message algebra of the protocol and define a map* $|\cdot|:\mathsf{A}\to\mathsf{A}'$*, such that* $|m|=|n|$ *iff* $m=n$*. This is an auxiliary construct to allow abstraction in defining private and authentic channels;*
10. $\mathsf{A}\cup\mathsf{A}'$ *is the set of all terms that can appear in our model of the protocol.*

We now introduce the subterm relation $\sqsubseteq$.

**Definition 2.** *Let a be an atom. Then* $\sqsubseteq$ *is the smallest inductively defined relation such that*

- $a\sqsubseteq a$*;*
- $a\sqsubseteq gh$ *if* $a\sqsubseteq g$ *or* $a\sqsubseteq h$*;*
- $a\sqsubseteq g\in\mathbf{G}$ *if* $g=g_1^{k_1}\cdot\ldots\cdot g_n^{k_n}$*,* $g_i$*'s are atoms of* $\mathbf{G}\quad$ *or* $\mathbf{G_c}$*,* $i\neq j\Rightarrow g_i\neq g_j$ *and* $\exists\ell(a=g_\ell^{k_\ell}\wedge k_\ell\neq 0)$ $(1\leq i,j,\ell\leq n)$*;*
- $a\sqsubseteq\mathsf{f}(x)$ *if* $a\sqsubseteq x$*;*
- $a\sqsubseteq|m|$ *if* $a\sqsubseteq m$*.*

For example, we have that $R_1\sqsubseteq\mathsf{f}(R_1\cdot R_2)$ and $R_1\not\sqsubseteq R_2\cdot R_3$ (even though $R_2\cdot R_3=R_2\cdot R_3\cdot R_1\cdot R_1^{-1}$), where $R_1$, $R_2$ and $R_3\in\mathsf{R}$.

Informally, a *strand* is a sequence of message transmissions and receptions, representing a role of the protocol. Transmission of the term $t$ is denoted $+t$, while reception of $t$ is written $-t$. Each transmission, respectively reception, corresponds to a *node* of the strand and we denote the *ith* node of strand $s$ as $\langle s,i\rangle$. The relation $n\Longrightarrow n'$ holds between nodes $n$ and $n'$ on the strand $s$ if $n=\langle s,i\rangle$ and $n'=\langle s,i+1\rangle$. The relation $n\longrightarrow n'$ represents communication between strands and holds between nodes $n$ and $n'$ if $term(n)=+t$ and $term(n')=-t$ ($term(n)$ denotes the signed term corresponding to node $n$, $unsterm(n)$ denotes the unsigned term corresponding to node $n$). A *strand space* $\Sigma$ is a set of strands and the relations $\Longrightarrow$ and $\longrightarrow$ impose a graph structure on the nodes of $\Sigma$. A *bundle* is a subgraph of the graph defined by the nodes of $\Sigma$ and $\Longrightarrow\cup\longrightarrow$. It represents a possible execution of the protocol. One possible bundle represents the expected protocol execution. The problem when analyzing a security protocol is that an intruder can insert or manipulate messages and may give raise to bundles not foreseen by the protocol designers.

Next we specify the possible actions of the intruder. The list of possible actions builds in some assumptions about the cryptographic system used. Firstly, we assume that it is impossible for the intruder to do factorisation, in other words given a product $g \cdot h$, the intruder cannot deduce $g$ or $h$ from them. Secondly, the intruder has no operation available which allows it to deduce any non-guessable element from any other element in $G$. Thirdly, the intruder cannot invert hash function $\mathsf{f}$: there is no way to deduce $g$ from $\mathsf{f}(g)$.

We have the following definition of intruder traces:

**Definition 3.** *An* intruder trace *is one of the following:*

- ***M**. Text message:* $\langle +t \rangle$, *where* $t$ *is a guessable element of* $A$ *(i.e. in* $T \cup C$*);*
- ***F**. Flushing:* $\langle -g \rangle$*;*
- ***T**. Tee:* $\langle -g, +g, +g \rangle$*;*
- ***C**. String concatenation:* $\langle -g, -h, +gh \rangle$*;*
- ***S**. String decomposition:* $\langle -gh, +g + h \rangle$*;*
- ***Op**. Arithmetic operation:* $\langle -g, -h, +(g \cdot h) \rangle$, *where the binary operator can be either from* $\mathbf{G}$ *or* $\mathbf{G_c}$.
- ***Apf**. Application of a hash function* $\mathsf{f}$*:* $\langle -g, +\mathsf{f}(g) \rangle$.

We define private, authenticated and resilient channels used in the protocol:

**Definition 4.** *Suppose* $A$ *sends* $B$ *message* $m$*. Then for some strand* $s_A \in A[*]$ $\exists i.term(\langle s_A, i \rangle) = m$*. Let* $\mathcal{C}$ *be any bundle where* $s_A \in \mathcal{C}$*, and consider the set* $S = \{n \in \mathcal{C} | \langle s_A, i \rangle \prec n \wedge k \sqsubseteq term(n)\}$[3] *for some non-guessable* $k \sqsubseteq m$*. We say there is a* private channel *between* $A$ *and* $B$ *if* $\forall n \in S.\ |m| \not\sqsubseteq term(n)$ *implies:*

- *either* $\exists i. \langle s_B, i \rangle \in S$ *and* $\langle s_B, i \rangle \preceq n$*;*
- *or there is* $\prec$*-minimal element* $n''$ *of the set* $\{n' \in \mathcal{C} | n' \prec n \wedge k \sqsubseteq term(n')\}$*, such that* $\langle s_A, i \rangle \not\prec n''$.

Intuitively, in our definition $|m|$ corresponds to an "encryption" of $m$ that the intruder can only duplicate or intercept. Moreover, a non-guessable $m$ (or part of it) can never be derived while transmitted over a private channel, unless the secret was revealed where the sender in the private channel did not causally contribute to the compromise.

**Definition 5.** *Suppose* $B$ *receives a message* $m$ *apparently from* $A$ *- for some strand* $s_B \in B[*]$ $\exists i.term(\langle s_B, i \rangle) = m$*. Let* $\mathcal{C}$ *be any bundle where* $s_B \in \mathcal{C}$*, and consider the set* $S = \{n \in \mathcal{C} | n \prec \langle s_B, i \rangle \wedge m \sqsubseteq term(n)\}$*. We say there is an* authenticated channel *between* $A$ *and* $B$ *if* $\exists n \in S.\ strand(n) = s_A$*.*[4]

A private (or authenticated) channel for a protocol $\Pi$ can be implemented by any protocol that guarantees secrecy (or authetication) in a composition with $\Pi$ (which may also subsume other primitive protocols).

We use the notion of a *reference bundle* in the next definition. Intuitively, it represents the full execution of the protocol without intruder. In the language of

[3] Logical connectives have the largest scope.

[4] Disregarding $m$ this definition corresponds to aliveness - the weakest type of authentication of Lowe's hierarchy. It could be lifted up to non-injective agreement by choosing an appropriate $m$.

strands, a bundle is a reference bundle ($\mathcal{C}_r$) if for any regular role A of a protocol $\exists! s_A \in \mathcal{C}_r$, such that $length(tr(s_A)) = max(\{length(tr(s)) | s \in A[*]\})$, and no intruder strand is in $\mathcal{C}_r$.

**Definition 6.** *Suppose $A$ sends $B$ message $m$. Then for some strand $s_A \in A[*]$ $\exists i.term(\langle s_A, i\rangle) = m$ and let $\mathcal{C}$ be any bundle where $s_A \in \mathcal{C}$. We say there is a* resilient channel *between $A$ and $B$ if $\{R \in roles\ |\exists s_R \in \mathcal{C} \wedge \exists k.term(\langle s_R, k\rangle) = -m \wedge \langle s_A, i\rangle \prec \langle s_R, k\rangle\} = \{R \in roles\ |\exists s'_R \in \mathcal{C}_r \wedge \exists k.\langle s_A, i\rangle \longrightarrow \langle s'_R, k\rangle\}$.*

We are now able to give a description of the protocol in strand spaces.

# 5 Fixing the Protocol and Proof of Correctness

## 5.1 Fixing the Protocol

Replay attacks exploit possible collision in the label space. An easy fix is to make $T$ distribute a fresh (possibly guessable) value to participants of the exchange at setup phase, which needs to be included in all messages in that run of the protocol[5]. However, to guarantee uniqueness in strand spaces $T$'s name needs to be associated with it, viz. we form a tuple $(T, n)$ and call it a *tag*. Obviously, for a tag to work, we need to disallow branching on $T$'s strand, viz. for any run of a setup phase, $T$ executes at most one run of the protocol. Furthermore, if $n$ in a tag is not a timestamp, *identical* setup phase requests must be distinguished, where timestamps or "handshake" mechanism have to be used. Namely, our assumption on a setup phase is: if bundles $B_1$ and $B_2$ represent *identical* setup phase runs, i.e. with the same participants and messages to be exchanged, occurring at times $t_1$ and $t_2$, respectively, then $T$-strands have different tags.

An intuitive fix to the arithmetic attack on the protocol is via making $T$ to verify consistency of multiplicative inverses, e.g. instead of the second check performed by $T$ in the González-Markowitch protocol, we suggest that $T$ verifies that

$$\forall i. \exists C \in \mathbf{C} \text{ such that } f(C \cdot R_{i-1}) = f(m_{i-1})$$

where $m_{i-1}$ is the item required by $P_i$. In order to perform this check $T$ needs to know agent-message correspondence, which can be established either at setup phase, or by $P_i$ sending $f(m_{i-1})$ in the message to $T$. Note that the second protocol proposed in [12], which we did not study here, uses similar tests to ensure consistency.

## 5.2 A Strand Space Model of Our Corrected Protocol

We now specify our corrected version of the protocol using the strand space formalism. The parameterized $P_i$ strand representing any honest participant $P_i$ is as follows:

$$P_i[t, m_i, \mathsf{f}(m_1), \ldots, \mathsf{f}(m_n), R_i] = \langle +t\ R_i, -t\ R_{i-1}, +t\ A_i\ C_i\ \mathsf{f}(R_i), -t\ \mathbf{C}\rangle$$

[5] Thus, timestamps or nonces may suffice for this purpose.

where $t \in \mathsf{T}$, $m_i \in \mathsf{M}$, $R_i, R_{i-1} \in \mathsf{R}$, $A_i = \mathsf{F}_n(m_i, \langle \mathsf{f}(m_k) \rangle_{k \neq i})$, $C_i = m_i \cdot {R_i}^{-1}$ and $\mathcal{C}$ is any concatenation of $C_j$s $(1 \leq i \leq n)$.

The role of the trusted party corresponds to the following strand:

$$T[t, \mathsf{f}(m_1), \ldots, \mathsf{f}(m_n)] = \langle -t\ A_1\ C_1\ \mathsf{f}(R_1), \ldots, -t\ A_n\ C_n\ \mathsf{f}(R_n), +t\ \mathbf{C} \rangle$$

The parametric $P_i$ and $T$ strands represent set of strands, i.e. the union of all possible instantiations. We denote these sets by $P_i[*]$ and $T[*]$ respectively.

Moreover, we make the following assumptions:

- all channels are private, authentic and resilient;
- during a setup phase all participants agree on a trusted party $T$ who generates a unique tag $t$, and at most one strand of each regular role receives it;
- $a = b$ iff $\mathsf{f}(a) = \mathsf{f}(b)$.

### 5.3 Correctness in Strands Model

**Fairness.** Fairness is the central property of our protocol. In both papers [12] and [14], *fairness* is defined as the property, whereby an honest participant receives all expected items corresponding to the items he has provided. This formulation is not formal enough for our analysis and, hence, the following definition is adopted:

**Definition 7.** *A multi-party cyclic fair exchange protocol is* fair *for a honest agent $P_i$ if $P_{i+1}$ obtains $m_i$ only if $P_i$ obtains $m_{i-1}$.*

**Definition 8.** *The protocol space is a collection of the following types of strands:*

1. *honest agent strands $s_i \in P_i[*]$;*
2. *trusted party strands $t \in T[*]$;*
3. *the penetrator strands, modeling dishonest agents.*

### Proof of Fairness

**Proposition 1.** *The protocol guarantees fairness.*

**Lemma 1.** *A fresh value $n$ generated by $T$ in a setup phase uniquely identifies the strand of $T$ in the protocol and a tag $(T, n)$ uniquely identifies regular strands of the protocol.*

*Proof.* Trivial from assumptions.

**Lemma 2.** *Suppose a collection of agents $\mathcal{P}$ runs the protocol, where $P_i \in \mathcal{P}$ is an honest agent, $T$ is the trusted party and $(T, n)$ is a tag. Then no agent $P_j \in \mathcal{P}$ can obtain $m_i$ without obtaining $\mathbf{C}$, where $\mathbf{C} \sqsubseteq unsterm(\langle \mathbf{t}, \mathbf{1} \rangle)$, $(T, n) \sqsubseteq unsterm(\langle t, 1 \rangle)$ and $t \in T[*]$.*

*Proof.* By assumption on a setup phase, all commodities to be exchanged are secret before the execution of the protocol. Let $s_i \in P_i[*]$ in some bundle $\mathcal{C}$. We have that $m_i \not\sqsubseteq \langle s_i, 0\rangle$ and $m_i \not\sqsubseteq \langle s_i, 1\rangle$. Hence, the two first messages do not reveal $m_i$ and the only event that may reveal $m_i$ is a node $nd = \langle s_i, 2\rangle$. By the assumption of resilient channels, $\exists t_T \in \mathcal{C}$, such that $\exists i.term(\langle t_T, i\rangle) = term(nd)$ and the node $nd' = \langle t_T, i\rangle$ is negative.

Consider the message $m = term(nd)$. $m_i \sqsubseteq m$ and $m_i$ uniquely originates on $nd$. Hence, no node $n \in \mathcal{C}$ exists, such that $m_i \sqsubseteq term(n)$ and $nd \not\prec n$. Let $S = \{n \in \mathcal{C} | nd \prec n \wedge m_i \sqsubseteq term(n)\}$. So, by the assumsion of private channels, we have $\forall n \in S.|m_i| \sqsubseteq term(n)$, unless $nd' \prec n$. In other words, $m_i$ is uncompromised up until $T$ receives it. Assuming that $P_i$ and $T$ remain honest in other runs of the protocol, as $(T, n) \sqsubseteq m$, by Lemma 1 replay of the message in any other run of the protocol will be rejected. Lastly, $length(t) = 2$ and $\mathbf{C} \sqsubseteq unsterm(\langle t, 1\rangle)$. Hence, the lemma holds. □

**Corollary 1.** *Fairness is preserved up to and including the third event on each honest agent's strand.*

**Lemma 3.** *For every honest $P_i$, $P_{i+1}$ obtains $m_i$ only if $P_i$ obtains $m_{i-1}$.*

*Proof.* Consider an honest agent $P_i$, participating in a cyclic exchange. Assume that $P_i$ completed all but the last steps of the protocol. There are two cases to consider:

1. $P_i$ does not receive the last message containing $\mathbf{C}$.
   As the agent-server link is assumed to be resilient, $T$ did not send $\mathbf{C}$ to $P_i$[6]. By assumption on a setup phase, $T$ is informed of all participants in the exchange and, as a result, $T$ did not send $\mathbf{C}$: either $T$ did not receive all expected messages or one of the checks did not succeed. In any case, $T$ did not send $\mathbf{C}$ to any other agent, and by Lemma 2 $m_i$ cannot be obtained by any other agent.
2. $P_i$ does receive the last message containing $\mathbf{C}$
   We need to show that $\exists C_{i-1} \in \mathbf{C}$, such that $C_{i-1} \cdot R_{i-1} = m_{i-1}$, where $R_{i-1}$ is sent by $P_{i-1}$ to $P_i$. Assume the opposite, that $\forall C \in \mathbf{C}.\ C \cdot R_{i-1} \neq m_{i-1}$. According to our fix, $T$ checks if $\exists C \in \mathbf{C}$ such that $f(C \cdot R) = f(m_i)$, where $m_i$ is the item required by $P_i$. $R = R_{i-1}$ as $P_i$ sends $f(R_{i-1})$ to $T$. So, $T$'s check also fails and it does not transmit $\mathbf{C}$ – a contradiction. Therefore, $\exists C \in \mathbf{C}$, s.t. $C \cdot R_{i-1} = m_{i-1}$. This means that if $T$ transmits $\mathbf{C}$ then $\forall i.P_i \in \mathcal{P}$ gets $m_i$.
   By Lemma 2 and the above argument the current lemma holds.

□

The last lemma proves our proposition. Indeed, it shows that the protocol is *(n-1)-resilient*, viz. even if all other agents are dishonest, fairness is guaranteed to the honest participant.

[6] Such statements can be made strictly formal by routine unwinding of defintions we gave previously, as it was done in the previous lemma.

# 6 Conclusion

In this paper we analysed a multi-party ring fair exchange protocol. The protocol has first been analysed by González and Markowitch, who discovered an attack and proposed a fix. The correctness of their fix was discussed using informal arguments. We show that their protocol is still flawed. Using the strand space model, which we adapted to model properties such as homomorphism and commutativity, we prove correctness of a modified version. The paper demonstrates again the difficulty of designing correct protocols, above all group protocols, and the crucial need for including formal methods for design and validation.

# References

1. N. Asokan, Birgit Baum-Waidner, Matthias Schunter, and Michael Waidner. Optimistic synchronous multi-party contract signing. Research Report RZ 3089, IBM Research Division, December 1998.
2. N. Asokan, Matthias Schunter, and Michael Waidner. Optimistic protocols for multi-party fair exchange. Research Report RZ 2892 (# 90840), IBM Research, December 1996.
3. N. Asokan, Matthias Schunter, and Michael Waidner. Optimistic protocols for fair exchange. In *4th ACM Conference on Computer and Communications Security*, Zurich, Switzerland, April 1997. ACM Press.
4. Feng Bao, Robert H. Deng, Khanh Quoc Nguyen, and Vijay Varadharajan. Multi-party fair exchange with an off-line trusted neutral party. In *DEXA 1999 Workshop on Electronic Commerce and Security*, Florence, Italy, September 1999.
5. Birgit Baum-Waidner. Optimistic asynchronous multi-party contract signing with reduced number of rounds. In Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen, editors, *Automata, Languages and Programming, ICALP 2001*, volume 2076 of *Lecture Notes in Computer Science*, pages 898–911, Crete, Greece, July 2001. Springer-Verlag.
6. Birgit Baum-Waidner and Michael Waidner. Round-optimal and abuse free optimistic multi-party contract signing. In *Automata, Languages and Programming — ICALP 2000*, volume 1853 of *Lecture Notes in Computer Science*, pages 524–535, Geneva, Switzerland, July 2000. Springer-Verlag.
7. Holger Bürk and Andreas Pfitzmann. Value exchange systems enabling security and unobservability. In *Computers and Security, 9(8):715–721*, 1990.
8. Rohit Chadha, Max Kanovich, and Andre Scedrov. Inductive methods and contract-signing protocols. In *8th ACM Conference on Computer and Communications Security*, Philadelphia, PA, USA, November 2001. ACM Press.
9. Rohit Chadha, Steve Kremer, and Andre Scedrov. Formal analysis of multi-party fair exchange protocols. In Riccardo Focardi, editor, *17th IEEE Computer Security Foundations Workshop*, pages 266–279, Asilomar, CA, USA, June 2004. IEEE Computer Society Press.
10. Danny Dolev and Andrew C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.
11. Shimon Even and Yacov Yacobi. Relations among public key signature systems. Technical Report 175, Technion, Haifa, Israel, March 1980.

12. Matthew K. Franklin and Gene Tsudik. Secure group barter: Multi-party fair exchange with semi-trusted neutral parties. In Ray Hirschfeld, editor, *Second Conference on Financial Cryptography (FC 1998)*, volume 1465 of *Lecture Notes in Computer Science*, pages 90–102, Anguilla, British West Indies, February 1998. International Financial Cryptography Association (IFCA), Springer-Verlag.
13. Juan A. Garay and Philip D. MacKenzie. Abuse-free multi-party contract signing. In *International Symposium on Distributed Computing*, volume 1693 of *Lecture Notes in Computer Science*, Bratislava, Slavak Republic, September 1999. Springer-Verlag.
14. Nicolás González-Deleito and Olivier Markowitch. Exclusion-freeness in multi-party exchange protocols. In *5th Information Security Conference*, volume 2433 of *Lecture Notes in Computer Science*, pages 200–209. Springer-Verlag, September 2002.
15. Steve Kremer and Olivier Markowitch. Fair multi-party non-repudiation. *International Journal on Information Security*, 1(4):223–235, July 2003.
16. Steve Kremer and Jean-François Raskin. A game-based verification of non-repudiation and fair exchange protocols. In Kim G. Larsen and Mogens Nielsen, editors, *Concurrency Theory—CONCUR 2001*, volume 2154 of *Lecture Notes in Computer Science*, pages 551–565, Aalborg, Denmark, August 2001. Springer-Verlag.
17. Jose Onieva, Jianying Zhou, Mildrey Carbonell, and Javier Lopez. A multi-party non-repudiation protocol for exchange of different messages. In *18th IFIP International Information Security Conference*, Athens, Greece, May 2003. Kluwer.
18. Olivier Pereira and Jean-Jacques Quisquater. Generic insecurity of cliques-type authenticated group key agreement protocols. In Riccardo Focardi, editor, *17th IEEE Computer Security Foundations Workshop*, pages 16–29, Asilomar, CA, USA, June 2004. IEEE Computer Society Press.
19. Steve A. Schneider. Formal analysis of a non-repudiation protocol. In *11th IEEE Computer Security Foundations Workshop*, pages 54–65, Washington - Brussels - Tokyo, June 1998. IEEE.
20. Vitaly Shmatikov and John Mitchell. Finite-state analysis of two contract signing protocols. *Theoretical Computer Science, special issue on Theoretical Foundations of Security Analysis and Design*, 283(2):419–450, 2002.
21. F. Javier Thayer Fabrega, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Proving security protocols correct. *Journal of Computer Security*, 7(2/3):191–230, 1999.

# A Strand Spaces

We here give basic definitions about strand spaces and bundles taken from [21].

**Definition 9.** *Let* $\mathsf{A}$ *be the set of all terms. A signed term is a pair* $\langle \sigma, a \rangle$ *with* $a \in \mathsf{A}$ *and* $\sigma$ *one of the symbols* $+, -$. *We will write a signed term as* $+t$ *or* $-t$. $(\pm\mathsf{A})$ *is the set of finite sequences of signed terms. We will denote a typical element of* $(\pm\mathsf{A})$ *by* $\langle\langle \sigma_1, a_1 \rangle, \ldots, \langle \sigma_n, a_n \rangle\rangle$.

*A strand space over* $A$ *is a set* $\Sigma$ *with a trace mapping* $tr : \Sigma \to (\pm A)^*$.

By abuse of language, we will still treat signed terms as ordinary terms. For instance, we shall refer to subterms of signed terms. We will usually represent a strand space by its underlying set of strands.

**Definition 10.** *Fix a strand space $\Sigma$.*

1. *A* node *is a pair $\langle s, i \rangle$, with $s \in \Sigma$ and $i$ an integer satisfying $1 \leq i \leq length(tr(s))$. The set of nodes is denoted by $\mathcal{N}$. We will say the node $\langle s, i \rangle$ belongs to the strand $s$. Clearly, every node belongs to a unique strand.*
2. *If $\langle s, i \rangle \in \mathcal{N}$ then $index(n) = i$ and $strand(n) = s$. Define $term(n)$ to be $(tr(s))_i$, i.e. the ith signed term in the trace of $s$. Similarly, $unsterm(n)$ is $((tr(s))_i)_2$, i.e. the unsigned part of the ith signed term in the trace of $s$.*
3. *There is an edge $n_1 \longrightarrow n_2$ if and only if $term(n_1) = +a$ and $term(n_2) = -a$ for some $a \in \mathsf{A}$. Intuitively, the edge means that node $n_1$ sends the message $a$, which is received by $n_2$, recording a potential causal link between those strands.*
4. *When $n_1 = \langle s, i \rangle$ and $n_2 = \langle s, i+1 \rangle$ are members of $\mathcal{N}$, there is an edge $n_1 \Longrightarrow n_2$. Intuitively, the edge expresses that $n_1$ is an immediate causal predecessor of $n_2$ on the strand $s$. We write $n' \Longrightarrow^+ n$ to mean that $n'$ precedes $n$ (not necessarily immediately) on the same strand.*
5. *An unsigned term $t$* occurs in *$n \in \mathcal{N}$ iff $t \sqsubseteq term(n)$.*
6. *Suppose $I$ is a set of unsigned terms. The node $n \in \mathcal{N}$ is an entry point for $I$ iff $term(n) = +t$ for some $t \in I$, and whenever $n' \Longrightarrow^+ n$, $term(n') \notin I$.*
7. *An unsigned term $t$* originates *on $n \in \mathcal{N}$ iff $n$ is an entry point for the set $I = \{t' \mid t \sqsubseteq t'\}$.*
8. *An unsigned term $t$ is* uniquely originating *in a set of nodes $S \subset \mathcal{N}$ iff there is a unique $n \in S$ such that $t$ originates on $n$.*
9. *An unsigned term $t$ is non-originating in a set of nodes $S \subset \mathcal{N}$ iff there is no $n \in S$ such that $t$ originates on $n$.*

If a term $t$ originates uniquely in a suitable set of nodes, then it can play the role of a nonce or session key, assuming that everything that the penetrator does in some scenario is in that set of nodes.

A parameterized strand, also called a *role*, is a strand which contains variables. The regular strands are generated by filling in the parameters with appropriate values. We write $s \in A[*]$ or, simply $s_A$, to mean that strand $s$ corresponds to a role A. By *roles* we mean the set of all regular participants of the protocol in consideration.

$\mathcal{N}$ together with both sets of edges $n_1 \longrightarrow n_2$ and $n_1 \Longrightarrow n_2$ is a directed graph $\langle \mathcal{N}, (\longrightarrow \cup \Longrightarrow) \rangle$.

A *bundle* is a finite subgraph of $\langle \mathcal{N}, (\longrightarrow \cup \Longrightarrow) \rangle$, for which we can regard the edges as expressing the causal dependencies of the nodes, Causal dependence is expressed by $\prec = (\longrightarrow \cup \Longrightarrow)^+$ and $\preceq$ is the reflexive version of $\prec$.

**Definition 11.** *Suppose $\longrightarrow_\mathcal{B} \subset \longrightarrow$, $\Longrightarrow_\mathcal{B} \subset \Longrightarrow$ and $\mathcal{B} = \langle \mathcal{N}_\mathcal{B}, (\longrightarrow_\mathcal{B} \cup \Longrightarrow_\mathcal{B}) \rangle$ is a subgraph of $\langle \mathcal{N}, (\longrightarrow \cup \Longrightarrow) \rangle$. $\mathcal{B}$ is a bundle if:*

1. *$\mathcal{N}_\mathcal{B}$ and $\longrightarrow_\mathcal{B} \cup \Longrightarrow_\mathcal{B}$ are finite.*
2. *If $n_2 \in \mathcal{N}_\mathcal{B}$ and $term(n_2)$ is negative, then there is a unique $n_1$ such that $n_1 \longrightarrow_\mathcal{B} n_2$.*
3. *If $n_2 \in \mathcal{N}_\mathcal{B}$ and $n_1 \Longrightarrow n_2$ then $n_1 \Longrightarrow_\mathcal{B} n_2$.*
4. *$\mathcal{B}$ is acyclic.*

By abuse of notation we write $n \in \mathcal{B}$ to mean $n \in \mathcal{N}_\mathcal{B}$.

# Achieving Fairness in Private Contract Negotiation*

Keith Frikken and Mikhail Atallah

Department of Computer Sciences,
Purdue University

**Abstract.** Suppose Alice and Bob are two entities (e.g. agents, organizations, etc.) that wish to negotiate a contract. A contract consists of several clauses, and each party has certain constraints on the acceptability and desirability (i.e., a private "utility" function) of each clause. If Bob were to reveal his constraints to Alice in order to find an agreement, then she would learn an unacceptable amount of information about his business operations or strategy. To alleviate this problem we propose the use of Secure Function Evaluation (SFE) to find an agreement between the two parties. There are two parts to this: i) determining whether an agreement is possible (if not then no other information should be revealed), and ii) in case an agreement is possible, coming up with a contract that is *valid* (acceptable to both parties), *fair* (when many valid and good outcomes are possible one of them is selected randomly with a uniform distribution, without either party being able to control the outcome), and *efficient* (no clause is replaceable by another that is better for both parties). It is the fairness constraint in (ii) that is the centerpiece of this paper as it requires novel techniques that produce a solution that is more efficient than general SFE techniques. We give protocols for all of the above in the semi-honest model, and we do not assume the Random Oracle Model.

## 1 Introduction

Suppose Alice and Bob are two entities who are negotiating a joint contract, which consists of a sequence of clauses (i.e., terms and conditions). Alice and Bob are negotiating the specific value for each clause. Example clauses include:

1. How will Alice and Bob distribute the revenue received for jointly performing a task?
2. Given a set of tasks, where Alice and Bob each have a set of tasks they are willing and able to perform, who performs which tasks?

* Portions of this work were supported by Grants EIA-9903545, IIS-0219560, IIS-0312357, and IIS-0242421 from the National Science Foundation, Contract N00014-02-1-0364 from the Office of Naval Research, by sponsors of the Center for Education and Research in Information Assurance and Security, and by Purdue Discovery Park's e-enterprise Center.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 270–284, 2005.
© Springer-Verlag Berlin Heidelberg 2005

3. Given a set of locations to perform certain tasks, in which locations does Alice (respectively, Bob) perform their tasks?

Alice and Bob will each have private constraints on the acceptability of each clause (i.e., rules for when a specific term is acceptable). A specific clause is an agreement between Alice and Bob if it satisfies both of their constraints. In a non-private setting, Alice and Bob can simply reveal their constraints to one another. However, this has two significant drawbacks: i) if there are multiple possible agreements how do Alice and Bob choose a specific agreement (some are more desirable to Alice, others more desirable to Bob), and ii) the revelation of one's constraints and preferences is unacceptable in many cases (e.g., if one's counterpart in the negotiation can use these to infer information about one's strategies or business processes or even use them to gain an information advantage for use in a future negotiation). This second problem is exacerbated when Alice and Bob are competitors in one business sector but cooperate in another sector. We propose a framework and protocols that facilitate contract negotiation without revealing private constraints on the contract. There are two components to such a negotiation: i) the ability to determine if there is a contract that satisfies both parties' constraints (without revealing anything other than "yes/no") and ii) if there is a contract that satisfies both parties' constraints, then a protocol for determining a contract that is *valid* (acceptable to both parties), *fair* (when many valid and good outcomes are possible one of them is selected randomly with a uniform distribution, without either party being able to control the outcome), and *efficient* (no clause is replaceable by another that is better for both parties).

We introduce protocols for both of these tasks in the semi-honest model (i.e., the parties will follow the protocol steps but may try to learn additional information). The results of the paper are summarized as follows:

- The definition of a framework for privacy preserving contract negotiation. This framework allows multiple independent clauses, but can be extended to support dependencies between different clauses.
- Protocols for determining if there is a valid contract according to both parties' constraints.
- Protocols for determining a fair, valid, and efficient contract when there is such a contract in the semi-honest model. The most difficult of these requirements is fairness, and we believe that the ability to choose one of several values without either party having control of the value will have applications in other domains.

The rest of the paper is organized as follows. In Section 2, an overview of related work is given. In Section 3, several building blocks are given that are used in later protocols. Section 4 describes our security model. Section 5 outlines the framework for secure contract negotiation. Section 6 describes protocols for computing the satisfiability of a clause as well as for determining a valid term for a clause. In Section 7, we discuss extensions to our protocols that allow Alice and Bob to make preferences. Section 8 introduces several extensions to our framework. Finally, Section 9 summarizes the results.

## 2 Related Work

The authors are not aware of any directly related work in this area. Much of the previous work in automated contract negotiation ([15, 14, 26]) focuses on creating logics to express contract constraints so that agreements can be computed. Our work is not to be confused with simultaneous contract signing [24], which solves the different problem of achieving simultaneity in the signing of a pre-existing already agreed-upon contract. The closest work is in [25], which deals with user preference searching in e-commerce. The problem addressed there is that a vendor may take advantage of a customer if that vendor learns the customer's constraints on a purchase (type of item, spending range, etc.). To prevent this, [25] suggests using a gradual information release.

Secure Multi-party Computation (SMC) was introduced in [27], which contained a scheme for secure comparison; suppose Alice (with input $a$) and Bob (with input $b$) desire to determine whether or not $a < b$ and without revealing any information other than this result (this is referred to as "Yao's Millionaire Problem"). More generally, SMC allows Alice and Bob with respective private inputs $a$ and $b$ to compute a function $f(a, b)$ by engaging in a secure protocol for some public function $f$. Furthermore, the protocol is private in that it reveals no additional information. By this what is meant is Alice (Bob) learns nothing other than what can be deduced from $a$ ($b$) and $f(a, b)$. Elegant general schemes are given in [11, 10, 1, 4] for computing any function $f$ privately. One of the general results in Two-party SMC is that if given a circuit of binary gates for computing a function $f$ that has $m$ input wires and $n$ gates, then there is a mechanism for securely evaluating the circuit with $m$ chosen 1-out-of-2 Oblivious Transfers(OTs), communication proportional to $n$, and a constant number of rounds [28]. There have been many extensions of this including: multiple parties, malicious adversaries, adaptive adversaries, and universal protocols [9, 3, 17]. Furthermore, [19] implemented the basic protocol along with a compiler for building secure protocols. However, these general solutions are considered impractical for many problems, and it was suggested in [13] that more efficient domain-specific solutions can be developed.

A specific SMC-based application that is similar to our work is that of [6], which introduced protocols for computing set intersection efficiently. Specifically, it introduced protocols (for the semi-honest and malicious models) for computing the intersection between two sets, the cardinality of set intersection, and to determine if the cardinality was above a threshold. That work also introduced protocols for multiple parties, approximating intersection, and for fuzzy set intersection. This is similar to the centerpiece of this paper as our work can be summarized as "choose a random element of the intersection, given that there is such an element".

# 3 Building Blocks

## 3.1 Cryptographic Primitives and Definitions

1. *Oblivious Transfer:* There are many equivalent definitions of Oblivious Transfer (OT), and in this paper we use the definition of chosen 1-out-of-$N$ OT, where Alice has a set of items $x_1, \ldots, x_N$ and Bob has an index $i \in \{1, \ldots, N\}$. The OT protocol allows Bob to obtain $x_i$ without revealing any information about $i$ to Alice and without revealing any information about other $x_j$ $(j \neq i)$ values to Bob. A high-level overview of OT can be found in [24]. Recently, there has been some work on the development of efficient OT protocols [20, 21]. It was also shown in [16] that there was no black-box reduction from OT to one-way functions.
2. *Homomorphic Encryption:* A cryptographic scheme is said to be homomorphic if for its encryption function $E$ the following holds: $E(x) * E(y) = E(x + y)$. Examples of homomorphic schemes are described in [5, 23, 22]. A cryptographic scheme is said to be *semantically secure* if $E(x)$ reveals no information about $x$. In other words $(E(x), E(x))$ and $(E(x), E(y))$ are computationally indistinguishable (defined in Section 4).
3. *Secure Circuit Evaluation*: A well known result in SMC is that boolean circuits composed of 2-ary gates can be evaluated with communication equal to the size of the circuit, a 1-out-of-2 OT per input wire, and a constant number of rounds [28, 12].

# 4 Preliminaries

In this section, we discuss our security model, which is similar to that of [2, 9] (however we use a subset of their definitions as we are in the semi-honest model). At a high level a protocol securely implements a function $f$ if the information that can be learned by engaging in the protocol, could be learned in an ideal implementation of the protocol where the functionality was provided by a trusted oracle. We consider semi-honest adversaries (i.e., those that will follow the protocol but will try to compute additional information other than what can be deduced from their input and output alone).

We now formally review the above notions for two party protocols. We do this by defining the notion of an ideal-model adversary (one for the situation where there is a trusted oracle) and a real-model adversary for the protocol $\Pi$, and then state that a protocol is secure if the two executions are computationally indistinguishable. Assume that $\Pi$ computes some function $f : \{0,1\}^\star \times \{0,1\}^\star \rightarrow \{0,1\}^\star$.

Alice (Bob) is viewed as a Probabilistic Polynomial Time (PPT) algorithm $A$ ($B$) that can be decomposed into two parts $A_I$ and $A_O$ ($B_I$ and $B_O$). Also Alice's (Bob's) private input is represented by $X_A(X_B)$. We represent Alice's view of the protocol as $\mathit{IDEAL}_{A,B}(X_A, X_B) = (A_O(X_A, r_A, Y_A), f(A_I(X_A, r_A), X_B))$ where $r_A$ is Alice's private coin flips.

We now define the actual execution for a protocol $\Pi$ that implements the function $f$. In a real model, the parties are arbitrary PPT algorithms $(A', B')$.

The adversaries are admissible if both parties use the algorithm specified by protocol $\Pi$ (as we are in the semi-honest model). We denote the interaction of protocol $\Pi$ by $REAL_{\Pi,A',B'}(X_A, X_B)$, which is the output from the interaction of $A'(X_A)$ and $B'(X_B)$ for protocol $\Pi$.

As is usual, we say that a protocol $\Pi$ securely evaluates a function $f$ if for any admissible adversary in the real model $(A', B')$, there exists an ideal-model adversary $(A, B)$ such that $IDEAL_{A,B}(X_A, X_B)$ and $REAL_{\Pi,A',B'}(X_A, X_B)$ are computationally indistinguishable. To define what is meant by this we recall the standard definition of computational indistinguishability [8]: Two probability ensembles $X \stackrel{\text{def}}{=} \{X_n\}_{n\in\mathcal{N}}$ and $Y \stackrel{\text{def}}{=} \{Y_n\}_{n\in\mathcal{N}}$ are computationally indistinguishable if for any PPT algorithm $D$, any polynomial $p$, and sufficiently large $n$ it holds that:

$|(Pr(D(X_n, 1^n) = 1)) - (Pr(D(Y_n, 1^n) = 1))| < \frac{1}{p(n)}$.

## 5 Secure Contract Framework

In this section we introduce a framework for secure contract negotiation. we begin with several definitions:

- A *clause* is a public set $S = \{s_0, \ldots s_{N-1}\}$ of possible values. We refer to each of these values as *terms*. We assume that Alice and Bob can agree on $S$ at the start of the negotiation. Furthermore, there is a defined ordering of the terms, so that $s_i$ is the $i$th term in the set.
- For each clause $S$, Alice (Bob) has a set of *constraints* on the acceptability of each of that clause's terms. These constraints are represented by sets $A$ (respectively, $B$), where $A \subseteq S$ ($B \subseteq S$) and $A$ ($B$) is the set of all terms for clause $S$ that are acceptable to Alice (Bob).
- A term $x \in S$ is *acceptable* iff $x \in (A \cap B)$.
- A clause is *satisfiable* iff $A \cap B \neq \emptyset$, i.e., there is a term for the clause that is acceptable to both Alice and Bob.
- A *negotiation* is a sequence of clauses $S_0, \ldots, S_{k-1}$. In this paper, we assume that these clauses are independent (i.e., that the acceptability of one clause does not depend on the outcome of another clause). We briefly discuss how to extend our protocols for dependent clauses in Section 8.3. A negotiation is *satisfiable* iff each clause is satisfiable.
- A *contract* for a negotiation is a sequence of terms $x_0, \ldots, x_{k-1}$ (where $x_i \in S_i$). A contract is *valid* if each term is acceptable to both parties. A valid contract is *efficient* if no term in it is replaceable by another term that is better for both Alice and Bob (according to their respective private valuation functions).

*Example:* Suppose Alice and Bob are entities that jointly manufacture some type of device, and furthermore they must negotiate where to manufacture the devices. The clause could take the form $S$ = {London, New York, Chicago, Tokyo, Paris, Ottawa}. Now suppose Alice's constraints are $A$ = {London, New York, Paris, Ottawa} and Bob's constraints are $B$ = {London, Tokyo, Paris, Ottawa}. The set of acceptable terms are those in $A \cap B$ = {London, Paris, Ottawa}.

We now outline the framework for our protocols. Protocols for computing the satisfiability of a clause and an acceptable term for the clause are given in the next section. However, this needs to be extended to the contract level, because the individual results for each clause cannot be revealed when the negotiation is not satisfiable. Given a protocol that evaluates whether or not a clause is satisfiable in a split manner, it is possible to determine if the contract is satisfiable. This is obvious since a contract is satisfiable iff all clauses are satisfiable, which can be computed easily by computing the AND of many split Boolean values using *Secure Circuit Evaluation* [28]. A key observation is that if a contract is satisfiable, then to find a valid and fair contract one can find the individual fair clause agreements independently. Thus given secure protocols for determining whether a clause is satisfiable and a protocol for determining an acceptable fair term for a satisfiable clause, it is possible to compute these same values at the contract level.

# 6 Secure Contract Term Protocols

In this section we propose private protocols for computing: i) the satisfiability of a clause(yes/no) and ii) a fair agreement for satisfiable clauses (recall that fair means that the term is chosen uniformly from all the set of acceptable terms and that neither party has control over the outcome). We postpone discussion of protocols for computing efficient agreements until Section 7. We now define some notation for our protocols. We assume that Alice and Bob are negotiating a specific clause with $N$ terms. We define Alice's (Bob's) acceptability for term $i$ to be a boolean value $a_i$ ($b_i$).

## 6.1 Determining Satisfiability

A clause is satisfiable iff $\bigvee_{i=0}^{N-1} a_i \wedge b_i$ is true. Clearly this satisfiability predicate be computed (with *Secure Circuit Evaluation*) with $O(N)$ communication and $O(1)$ rounds.

## 6.2 Computing a Fair Acceptable Term

In this section we introduce a protocol for computing a fair acceptable term for a clause that is known to be satisfiable. The protocol can be described as follows:

**Protocol Description:**
**Input:** Alice has a set of binary inputs $a_0, \ldots, a_{N-1}$ and likewise Bob has a set of inputs: $b_0, \ldots, b_{N-1}$. Furthermore it is known that $\exists i \in [0, N)$ such that $a_i \wedge b_i$ is true.
**Output:** An index $j$ such that $a_j \wedge b_j$ is true, and if there are many such indices, then neither party should be able to control which index is chosen (by modifying their inputs).

Figure 1 describes our protocol for computing a fair acceptable term. However, we also discuss three elements about the protocol's difficulty including: i) we show that chosen OT reduces to the above mentioned problem, ii) we discuss a solution using circuit simulation for this problem, and iii) we show a false start for the problem.

**A Reduction from OT:**

Suppose Bob is choosing 1 out of $N$ items (item $i$) from Alice's list of binary values $v_0, \ldots, v_{N-1}$. Alice and Bob define a list of $2N$ values. Alice creates a list where item $a_{2j+v_j}$ is true and $a_{2j+1-v_j}$ is false for all $j \in [0, N)$. Bob creates a

**Input:** Alice has a set of binary inputs $a_0, \ldots, a_{N-1}$ and likewise Bob has a set of inputs: $b_0, \ldots, b_{N-1}$. Furthermore it is known that $\exists i \in [0, N)$ such that $a_i \wedge b_i$ is true.

**Output:** An index $j$ such that $a_j \wedge b_j$ is true, and if there are many such indices, then neither party should be able to control which index is chosen.

1. Alice does the following:
   (a) She chooses a semantically-secure homomorphic encryption function $E_A$ (with modulus $M_A$) and publishes its public keys and public parameters.
   (b) For each item $a_i$ in the list $a_0, \ldots, a_{N-1}$, she creates a value: $\alpha_i \leftarrow E_A(a_i)$. She sends these values to Bob.
2. Bob does the following:
   (a) He chooses a semantically-secure homomorphic encryption function $E_B$ (with modulus $M_B$) and publishes the public keys and the public parameters.
   (b) For each $i$ from 0 to $N-1$, Bob chooses/computes:
      i. A random value $r_i$ chosen uniformly from $\{0, 1\}$.
      ii. If $b_i = 0$, then he sets $\beta_i \leftarrow E_A(0)$, and otherwise he sets it to $\beta_i \leftarrow \alpha_i * E_A(0)$.
      iii. if $r_i = 0$, then $\gamma_i = \beta_i$, and otherwise $\gamma_i = ((\beta_i * E_A(M_A - 1))^{M_A - 1})$.
      iv. $\delta_i[0] \leftarrow E_B(r_i)$ and $\delta_i[1] \leftarrow E_B(1 - r_i)$

   Bob forms ordered triples $(\gamma_i, \delta_i[0], \delta_i[1])$ and randomly permutes all of the tuples (storing the permutation $\Pi_B$), and he sends the permuted list of ordered triples to Alice.
3. Alice permutes the triples using a random permutation $\Pi'$ and then for each triple in the permuted list $(\gamma_i, \delta_i[0], \delta_i[1])$ (note that these $i$ values are not the same ones that Bob sent, but are the new values in the permuted list) she computes/chooses:
   (a) $\zeta_i \leftarrow \delta_i[D_A(\gamma_i)] * E_B(0)$
   (b) $\eta_i \leftarrow \zeta_i * (\eta_{i-1})^2$ (if $i = 0$, then she sets it to $\zeta_0$).
   (c) She chooses a random $q_i$ uniformly from $\mathbb{Z}^*_{M_B}$.
   (d) $\theta_i \leftarrow (\eta_i * E_B(-1))^{q_i}$

   Alice permutes the $\theta$ values using another random permutation $\Pi''$ and she computes the permutation $\Pi_A = \Pi''\Pi'$. She sends the permuted $\theta$ values along with the permutation $\Pi_A$
4. Bob decrypts the values with $D_B$ and finds the value that decrypts to 0; he then finds the original index of this value by inverting the permutation and he announces this index.

**Fig. 1.** Protocol FIND-AGREEMENT

similar list, but sets only values $b_{2i}$ and $b_{2i+1}$ to true (and all other values are set to false). Alice and Bob engage in the above mentioned protocol. Clearly from the returned value, Bob can deduce the value of Alice's bit $v_i$.

**Using Circuit Simulation**

In order to make the term fair, the participants could each input a random permutation into the circuit that would compose the permutations and then permute the list with the composed permutation. The circuit would then choose the first value in the list that was an agreement. This would be fair because if at least one party chose a random permutation than the composed permutation would also be random (making the first acceptable item a fair choice). However, this would require at least $O(N \log N)$ inputs into the circuit (and thus this many 1-out-of-2 OTs) as this is the minimum number of bits to represent a permutation. Also, the circuit would have to perform the permutation, which would involve indirectly accessing a list of size $N$ exactly $N$ times. The direct approach of doing this would require $O(N^2)$ gates. Thus the standard circuit would require at least $O(N \log N)$ OTs (also this many modular exponentiations) and $O(N^2)$ communication. The protocol we outline below requires $O(N)$ modular exponentiations and has $O(N)$ communication. Now, it may be possible to reduce the number of bits input into the circuit to $O(N)$ by using a pseudorandom permutation, however this would require the computation of a permutation, which would be a difficult circuit to construct.

**Some False Starts for Sub-Linear Communication**

It would be desirable for a protocol to have sub-linear (in terms of the number of possible terms) communication. A possible strategy for this is to use a randomized approach. This solution works well if it is known that the sets have a "substantial" intersection, but all that is known is that there is at least one item that is in the intersection. Furthermore, the participants do not want to leak additional information about their own sets, including information about their mutual intersection. And thus any probabilistic strategy must behave as if there is only a single item in the intersection, and such a protocol would not have sub-linear communication. As a final note if it the contract is to be fair and efficient then there is a communication complexity of $\Omega(N)$ for finding such a contract (we prove this in Section 7).

**Proof of Correctness:**

Before discussing the security of the above protocol, we show that the protocol is correct in that it computes an agreement. It is easy to verify that the permutations do not effect the result as they are reversed in the opposite order that they were used, and thus our correctness analysis ignores the permutations. We consider a specific term with respective acceptability for Alice and Bob as $a_i$ and $b_i$ (we use $c_i$ to denote $a_i \wedge b_i$). We now trace the protocol describing each variable:

1. The value $\alpha_i$ is $E_A(a_i)$.
2. The value $\beta_i$ is $E_A(c_i)$.

3. It is easy to verify that the value $\gamma_i$ is $E_A(c_i \oplus r_i)$ (where $\oplus$ denotes exclusive-or).
4. The value $\delta_i[0]$ is $E_B(r_i)$ and the value $\delta_i[1]$ is $E_B(1 - r_i)$
5. Now, $\zeta_i$ is $\delta_i[0]$ when $c_i = r_i$ and is $\delta_i[1]$ otherwise. This implies that $\zeta_i = E_B(c_i)$.
6. Let $\hat{i}$ be the first index where $\zeta_{\hat{i}}$ is $E_B(1)$. For $i < \hat{i}$, the value $\eta_i$ will be $E_B(0)$. Furthermore, the value $\eta_{\hat{i}}$ will be $E_B(1)$. However, for $i > \hat{i}$ the value $\eta_i$ will be something other than $E_B(1)$, because $\eta_i = \zeta_i + {\eta_{i-1}}^2$.
7. If $\eta_i = E_B(x_i)$, the value $\theta_i$ will be $E_B(q_i(x_i - 1))$, this value will be $E_B(0)$ only when $x_i = 1$, which will only happen at $i = \hat{i}$. □

### Proof of Security (Semi-honest Model)

There are two parts to proving that this protocol is secure: i) that Alice (or Bob) does not learn additional information about indices that are not the output index, and ii) since we consider the permutations to be inputs into the protocol, we must show that that a party cannot choose its permutation to affect the outcome. Since Alice and Bob's roles are not symmetrical in the protocol, we must prove security for both cases.

### Alice

We introduce an algorithm $S_B$ that takes Alice's inputs and creates a transcript that is computationally indistinguishable from Alice's view in the real model. This algorithm is shown in Figure 2.

**Input:** Alice sees $a_0, \ldots, a_{N-1}, E_A, D_A, E_B$ and she sees the output index $j$.
**Output:** Alice must generate values indistinguishable from Bob's values in step 2; these values are triples of the form $(\gamma_i, \delta_i[0], \delta_i[1])$

1. Alice generates a sequence of values $\hat{b}_0, \ldots, \hat{b}_{N-1}$ where where $\hat{b}_i$ is chosen uniformly from $\{0, 1\}$ if $i \neq j$ and is 1 if $i = j$.
2. Alice generates a sequence of random bits $r_0, \ldots, r_{N-1}$ chosen uniformly from $\{0, 1\}$.
3. Alice creates tuples of the from $(E_A(r_i \oplus (a_i \wedge \hat{b}_i)), E_B(r_i), E_B(\neg r_i))$.
4. Alice permutes the items using a random permutation and then outputs these tuples.

**Fig. 2.** Algorithm $S_B$

**Lemma 1.** *In the semi-honest model, $S_B$ is computationally indistinguishable from Alice's view from running FIND-AGREEMENT.*

**Proof:** Since $E_B$ is semantically-secure, the second and third elements of the tuple are indistinguishable from the real execution. To show that the first item is computationally indistinguishable, we must show two things: i) that the decrypted values are indistinguishable (since Alice knows $D_A$), and ii) that from Alice's previous information that she created in Step 1 she cannot distinguish the values.

To show (i), the decrypted values from $S_B$ are chosen uniformly from $\{0, 1\}$. In the real execution the values are $E_A(c_i \oplus r_i)$, where $r_i$ is chosen uniformly from $\{0, 1\}$. Thus, the sequences are indistinguishable.

Part (ii) follows from the statement that Bob performs at least one multiplication on each item or he generates the values himself. By the properties of semantically secure homomorphic encryption, this implies that these values are indistinguishable. □

**Lemma 2.** *In the semi-honest model, Alice cannot control which term is chosen by selecting her permutation in the protocol FIND-AGREEMENT.*

**Proof:** The composition of two permutations, with at least one being random, is random. Thus, when Bob randomly permutes the tuples in Step 2, Alice cannot permute them in a way that benefits her, as she does not know Bob's permutation. Thus, when she computes the $\theta$ values the permutation is random and the term is chosen fairly. □

**Bob**

We introduce an algorithm $S_A$ that takes Bob's inputs and creates a transcript that is computationally indistinguishable from Alice's view in the real model. This algorithm is shown in Figure 3.

**Input:** Bob sees $b_0, \ldots, b_{N-1}, E_B, D_B, E_A$ and he sees the output index $j$.
**Output:** Bob must generate values indistinguishable from Alice's values in steps 1 and 3; these values include: $E_A(a_0), \ldots, E_A(a_{N-1}), \theta_0, \ldots, \theta_{N-1}$ and $\Pi$.

1. Bob generates a sequence of values $\hat{a}_0, \ldots, \hat{a}_{N-1}$ where where $\hat{a}_i$ is chosen uniformly from $\{0, 1\}$ if $i \neq j$ and is 1 if $i = j$.
2. Bob generates a list of $N$ items $\bar{\theta}_0, \ldots, \bar{\theta}_{N-1}$ where the $j$th value is 0 and all other values are chosen uniformly from $\mathbb{Z}^*_{M_B}$. He then creates a random permutation $\hat{\Pi}$ and permutes the values. Call this permuted list $\hat{\theta}_0, \ldots, \hat{\theta}_{N-1}$.
3. Bob outputs $E_A(\hat{a}_0), \ldots, E_A(\hat{a}_{N-1}), \hat{\theta}_0, \ldots, \hat{\theta}_{N-1}$ and $\hat{\Pi}$.

**Fig. 3.** Algorithm $S_A$

**Lemma 3.** *In the semi-honest model, $S_A$ is computationally indistinguishable from Bob's view from running FIND-AGREEMENT.*

**Proof:** Since $E_A$ is semantically-secure, the values $E_A(\hat{a}_0), \ldots, E_A(\hat{a}_{N-1})$ are indistinguishable from the real execution and the permutation $\hat{\Pi}$ is also indistinguishable. To show that the the values $\hat{\theta}_0, \ldots, \hat{\theta}_{N-1}$ are computationally indistinguishable from the real execution, we must show two things: i) that the decrypted values are indistinguishable (since Bob knows $D_B$), and ii) that from his computations from Step 2, he cannot distinguish the values.

To show (i), the values in $S_A$ are $N - 1$ random values and a single 0 value where the 0 is placed randomly. And since in Step 3.d of the protocol, Alice multiplies the values by a random value and then permutes the items these values are indistinguishable.

To show (ii), all that needs to be shown is that Alice performs a multiplication on each item, and this is clearly done in Step 3.a of the protocol. □

**Lemma 4.** *In the semi-honest model, Bob cannot control which term is chosen by selecting his permutation in the protocol FIND-AGREEMENT.*

**Proof:** The composition of two permutations, with at least one being random, is random. Thus when Alice permutes the list with $\Pi'$ the values are randomly permuted, and when the first agreement is chosen from this list, it is fairly chosen. □

## 7 Expressing Preferences

It is of course unrealistic to assume that Alice and Bob have sets of acceptable states that are all equally desirable. There are many terms for a clause that are a win-win situation for Alice and Bob (i.e., both prefer a specific term), however the random selection provided by FIND-AGREEMENT does not allow the choice of contracts that are efficient in the sense that both parties may prefer another term. Therefore by efficient we mean Pareto-optimal: Any improvement for Alice must be at the expense of Bob and vice-versa. In this section, we describe an extension that allows Alice and Bob to make preference choices through arbitrary utility functions that assign a desirability score to each term. We then filter out all terms that are not Pareto-optimal.

Let $U_A(x)$ (respectively, $U_B(x)$) denote Alice's (Bob's) utility for term $x$. In this section we introduce a filtering protocol FILTER, that filters out inefficient solutions. We assume that any terms that are deemed unacceptable to a party have utility of 0 for that party, and we assume that all acceptable terms have unique utility (i.e, there are no ties). This last constraint is reasonable since if two terms have equal desirability, then the parties can easily just assign them unique utilities in a random order.

*Example:* Returning to our example where $S$ = {London, New York, Chicago, Tokyo, Paris, Ottawa}, $A$ = {London, New York, Paris, Ottawa} and $B$ = {London, Tokyo, Paris, Ottawa}. Suppose Alice sets her utilities to {London(3), New York(4), Chicago(0), Tokyo(0), Paris(1), Ottawa(2)}, and Bob sets his utilities to {London(3), New York(0), Chicago(0), Tokyo(1), Paris(4), Ottawa(2)}. Recall that the original list of acceptable terms with utilities is {London(3,3), Paris(1,4), Ottawa(2,2)}. In this case Ottawa is an inefficient solution for this negotiation, because both parties prefer London to it.

It suffices to give a protocol for marking the terms of $S$ that are inefficient. We do this by computing a value between Alice and Bob that is XOR-split (i.e., each party has a value, and the exclusive-or of their values is equal to the predicate "term is efficient"). It is a natural extension of the FIND-AGREEMENT protocol to utilize such values and we omit the details. We omit a detailed proof of security, as this is a natural extension to the proofs outlined before. This filtering process is described in Figure 4.

**Input:** Alice has binary values $a_0, \ldots, a_{N-1}$, a set of integer utilities $A_0, \ldots, A_{N-1}$, a homomorphic encryption schemes $E_A$ (where the modulus is $M_A$) and $D_A$. Bob also has a list of binary values $b_0, \ldots, b_{N-1}$, a set of integer utilities $B_0, \ldots, B_{N-1}$, and has $E_A$. It is also known that there is a term where $a_i \wedge b_i = 1$.
**Output:** Alice has binary values $\bar{a}_0, \ldots, \bar{a}_{N-1}$ and Bob has $\bar{b}_0, \ldots, \bar{b}_{N-1}$ where $\bar{a}_i \oplus \bar{b}_i = a_i \wedge b_i$ and the utility $(A_i, B_i)$ is not dominated by another term. Furthermore, this list can be in any order.

1. Alice sends Bob $E_A(A_0), \ldots, E_A(A_{N-1})$.
2. For each $i$ from 0 to $N-1$, Bob does the following:
   (a) Bob chooses a random values $r_i$ in $\mathbb{Z}_{M_A}$.
   (b) If $b_i = 1$, then Bob computes $\alpha_i = E_A(a_i) * E_A(-r_i)$. And if $b_i = 0$, then Bob computes $\alpha_i = E_A(-r_i)$

   Bob sorts the $\alpha$ values in descending order according to his utility function. He sends these "sorted" values to Alice.
3. Alice and Bob engage in a Scrambled Circuit Evaluation that computes the max of the first $i$ items and then if the $(i+1)$st item is smaller than this max it replaces it by 0, otherwise it replaces it with 1. This is done in a XOR-split fashion. Clearly, this circuit can be done with $O(N)$ comparison circuits. One practical matter is the the comparison circuits must be able to compare $\rho$ bits, where $\rho$ is the security parameter for a homomorphic scheme (which has a substantial number of bits). However, the techniques in [7] can be used to reduce the number of bits used by the comparison circuit substantially.

**Fig. 4.** Protocol FILTER

As a final note, we prove that finding a fair and efficient term has a communication complexity of $\Omega(N)$. We do this by showing a reduction from Set Disjointness (which has a lower bound of $\Omega(N)$ [18]. We now give a sketch of this proof:

Suppose Alice has a set $A$ and Bob has a set $B$. Alice and Bob define another item (call it $c$) and both include it in their sets. They assign utilities to all items in their sets randomly, with the condition that the utility of $c$ has to be lower than the utilities of all other items in their sets. They engage in a protocol to find a fair and efficient item. If the item is $c$, then the sets are disjoint and if the item is not $c$ then the sets are not disjoint. □

## 8 Extensions

In this section we outline three extensions (due to page constraints we omit many details). In section 8.1 we discuss extending our scheme to allow various types of interactive negotiation. In section 8.2 we discuss how to make our protocol's communication proportional to $O(|A| + |B|)$ (which could be more efficient than our previous solution). Finally, in section 8.3 we outline how to handle dependent contract terms.

### 8.1 Interactive Negotiations

Consider what happens when the negotiators run the protocol and the output is that no agreement is possible. If the parties stopped here then this system may not be very useful for them. We now outline some strategies that will help them negotiate contracts in a more "interactive" fashion. One of the problems with these approaches is that they allow the entities to perform some level of probing. Some of these strategies require changes to our protocols, but we leave the details for the final version of the paper: i) the parties could change their values and run the protocol again, ii) the parties could make several acceptability sets (in some order of acceptability) and run a protocol that uses all of these sets as a batch, and iii) the protocols could give some feedback to the users. Some possible types of feedback include: what are the clauses without an agreement, if the number of clauses without an agreement is below some threshold than what are the clauses, or based on thresholds the protocols could output some metric as to how far away the parties are from an agreement.

### 8.2 Efficient Communication

The protocols outlined before our not particularly efficient if Alice and Bob's acceptability sets are much smaller than $N$. It would be desirable to have protocols with communication proportional to $|A| + |B|$. The downside to such a system is that it reveals "some" additional information, but we believe there are situations where such values are acceptable to leak. Our protocols can be modified to support such clauses, through usage of the protocols in [6].

### 8.3 Dependent Contract Terms

In this section we briefly outline an extension to our framework for dependent clauses. Two clauses are *dependent* if the value of one clause affects the acceptability set of another clause. For example, the location of a contract might effect which tasks a company is willing/capable to do. Another issue with dependency is if the dependency relationship is known globally or if it must be hidden. Here we assume that information about which clauses are dependent is public.

We now present a more formal definition of two-clause dependency (which can easily be generalized to $n$-clause dependency). Alice views clause $C_2$ as *dependent* on clause $C_1$ if the acceptability set for $C_2$ (call it $A_2$) is a function of the term value chosen for $C_1$. Any contract with dependent clauses can be handled with our framework by taking every group of dependent clauses $C_1, \ldots, C_k$ and making a "super"-clause to represent all of the clauses. The set of states for this "super"-clause would be the $k$-tuples in the set $C_1 \times \ldots \times C_k$.

## 9 Summary

In this paper we define protocols for negotiating a contract between two entities without revealing their constraints for the contract. There are two essential issues that need to be addressed: i) is there an agreement for a contract and ii) if there is an agreement, then what is a valid, fair, and efficient contract. To provide efficiency we propose assigning utilities to terms and then filtering out inefficient solutions. To provide fairness the protocols choose a random efficient term in such a way that neither party has control over the choice of the term; the protocol for achieving fairness is the centerpiece of this exposition. Furthermore, the protocols can be extended to handle contracts with publicly-known inter-clause dependencies. This is a first step in the area of secure contract negotiation; possible future work includes: i) protocols for dependent clauses that are better than the generic equivalents, ii) protocols for specific terms that are more efficient than the generic protocols presented in this paper, iii) extending the framework to more than two parties, iv) extending the protocols to a model of adversary besides semi-honest, and v) extending the framework to allow multiple negotiations with inter-contract dependencies.

## Acknowledgments

The authors would like to thank the anonymous reviewers for their useful comments and suggestions.

## References

1. Michael Ben-Or and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 1–10. ACM Press, 1988.
2. R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.
3. Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 494–503. ACM Press, 2002.
4. David Chaum, Claude Crépeau, and Ivan Damgard. Multiparty unconditionally secure protocols. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 11–19. ACM Press, 1988.
5. Ivan Damgård and Mads Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In *4th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2001*, LNCS 1992, pages 119–136, 2001.
6. M. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *International Conference on the Theory and Application of Cryptographic Techniques, EUROCRYPT 04*, 2004.
7. K. Frikken and M. Atallah. Privacy preserving route planning. In *To appear in Proceeding of the ACM workshop on Privacy in the Electronic Society.* ACM Press, 2004.
8. O. Goldreich. *Foundations of Cryptography: Volume I Basic Tools.* Cambridge University Press, 2001.

9. O. Goldreich. *Foundations of Cryptography: Volume II Basic Application.* Cambridge University Press, 2004.
10. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the nineteenth annual ACM conference on Theory of computing*, pages 218–229. ACM Press, 1987.
11. Oded Goldreich. Secure multi-party computation. Working Draft, 2000.
12. Oded Goldreich. Cryptography and cryptographic protocols. *Distrib. Comput.*, 16(2-3):177–199, 2003.
13. Shafi Goldwasser. Multi party computations: past and present. In *Proceedings of the sixteenth annual ACM symposium on Principles of distributed computing*, pages 1–6. ACM Press, 1997.
14. Guido Governatori, Arthur H.M. ter Hofstede, and Phillipa Oaks. Defeasible logic for automated negotiation. In P. Swatman and P.M. Swatman, editors, *Proceedings of CollECTeR*. Deakin University, 2000. Published on CD.
15. Benjamin N. Grosof, Yannis Labrou, and Hoi Y. Chan. A declarative approach to business rules in contracts: courteous logic programs in XML. In *ACM Conference on Electronic Commerce*, pages 68–77, 1999.
16. R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 44–61. ACM Press, 1989.
17. J. Katz and R. Ostrovsky. Round optimal secure two-party computation. In *CRYPTO 04*, 2004.
18. E. Kushilevitz and N. Nisan. *Communication Complexity.* Cambridge University Press, 1997.
19. D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella. Fairplay - a secure two-party computation system. In *Proceedings of Usenix Security*, 2004.
20. Moni Naor and Benny Pinkas. Oblivious transfer and polynomial evaluation. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 245–254. ACM Press, 1999.
21. Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 448–457. Society for Industrial and Applied Mathematics, 2001.
22. T. Okamoto, S. Uchiyama, and E. Fujisaki. Epoc: Efficient probabilistic public-key encryption, 1998.
23. Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Application of Cryptographic Techniques, EUROCRYPT 99*, LNCS 1592, pages 223–238, 1999.
24. Bruce Schneier. *Applied Cryptography – Protocols, algorithms, and souce code in C.* John Wiley & Sons, Inc., 1996.
25. R. Smith and J. Shao. Preserving privacy when preference searching in e-commerce. In *Proceeding of the ACM workshop on Privacy in the Electronic Society*, pages 101–110. ACM Press, 2003.
26. Michael Strbel. Intention and agreement spaces - a formalism.
27. A.C Yao. Protocols for secure computation. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, pages 160–164, 1982.
28. A.C Yao. How to generate and exchange secrets. In *Proceedings of the 27th Annual IEEE Symposium on Foundations of Computer Science*, pages 162–167, 1986.

# Small Coalitions Cannot Manipulate Voting

Edith Elkind[1] and Helger Lipmaa[2]

[1] Princeton University, USA
[2] Cybernetica AS and University of Tartu, Estonia

**Abstract.** We demonstrate how to make voting protocols resistant against manipulation by computationally bounded malicious voters, by extending the previous results of Conitzer and Sandholm in several important directions: we use one-way functions to close a security loophole that allowed voting officials to exert disproportionate influence on the outcome and show that our hardness results hold against a large fraction of manipulating voters (rather than a single voter). These improvements address important concerns in the field of secure voting systems. We also discuss the limitations of the current approach, showing that it cannot be used to achieve certain very desirable hardness criteria.

**Keywords:** Electronic voting, one-way functions, vote manipulation.

## 1 Introduction

In a democratic society, many important decisions are made based on the results of popular voting. Probably, the most frequently used voting scheme is Plurality: each voter submits a ballot with a candidate's name on it, and the candidate with the largest number of votes wins. While this is the most natural approach for the case of two candidates, when the number of candidates is larger, it does not work so well, as it is notoriously insensitive to the voters' preferences other than their top choice. Consequently, the problem of designing a more expressive voting procedure that is also fair and efficient has been a subject of extensive research.

The notions of expressiveness and efficiency are relatively easy to formalise: we assume that each voter submits a total ordering of the alternatives, best to worst (a more general definition would allow the voter to express the intensity of his preferences, but in many situations an ordering is sufficient), and, as is common in computer science, we restrict ourselves to voting schemes that given the set of votes, compute a winner in polynomial time (one should note, however, that there are interesting voting schemes that do not have this property unless $P = NP$.

On the other hand, a good definition of fairness proved to be much more elusive: in fact, the celebrated theorem of Arrow demonstrates that certain very desirable fairness criteria for a voting scheme are mutually incompatible. A system that is perceived as unfair can provoke a desire to game it (a much publicised example is provided by `http://voteswap.com`). Indeed, it has been

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 285–297, 2005.
© Springer-Verlag Berlin Heidelberg 2005

shown (see [Gib73, Sat73]) that any non-dictatorial voting scheme of the above-described type is susceptible to manipulation by voters, i.e., there are situations when a voter can misrepresent his preferences and achieve an outcome that he likes better than the truthful outcome. If many voters engage in these activities, the outcome of the election may be seriously distorted, and the results will not be representative of the true distribution of preferences. Thus, vulnerability to manipulation is a serious problem that has to be addressed.

While from information-theoretic perspective no solution is possible, computational considerations come to rescue: one can try to discourage potential manipulators by making manipulation infeasible. Indeed, it is known [BO91] that certain voting schemes (e.g., Single Transferable Vote) are NP-hard to manipulate. Furthermore, in a recent sequence of papers [CS02, CS03], Conitzer and Sandholm showed that several other voting schemes can be uniformly modified so that manipulating them becomes computationally hard. This is achieved by adding a pre-round in which candidates are divided into pairs and the voters' preferences are used to determine the winner of each pair; the winners of this stage participate in elections conducted according to the original protocol. Different methods for pairing up the candidates and eliciting the votes give rise to different levels of complexity, such as NP-hardness, #P-hardness, or PSPACE-hardness.

Conitzer and Sandholm leave it as an open question whether this approach can make manipulation as hard as inverting one-way functions, with the main motivation of achieving some kind of an average-case hardness. This question is particularly interesting because the setting of voting manipulation is reminiscent of that of common cryptographic tasks, where the goal is to construct a protocol that can withstand an attack by malicious adversary with overwhelming probability, i.e., on average, assuming that the adversary computationally bounded.

Motivated by this question of Conitzer and Sandholm, we modify their construction so that the pre-round schedule is computed from the votes of all voters using a one-way function. Since the size of the argument to this function is determined by the number of candidates $m$, our results are interesting only if $m$ is large (this is also the case in [CS02, CS03]): for all voting schemes considered in this paper, manipulation is easy when the number of candidates $m$ is very small, and in Section 4, we show that in some sense, this is inevitable. However, in some real-life scenarios (most notably, recent California gubernatorial elections), the number of candidates is sufficient to make our results applicable.

Unfortunately, we do not achieve average-case hardness (some reasons why this is unlikely to be possible are outlined in Section 4). The good news is that by using a deterministic one-way function to fix the scheduling we achieve some other attractive properties.

First, we show that our method can be used to make voting protocols hard to manipulate even by a large minority fraction (1/6 in the case of Plurality, STV and Maximin, $1/m$ in the case of Borda) of voters, while the previous literature concentrated on the case of a single manipulator. Arguably, the voters who want to manipulate the election may and do collude, so constructing voting schemes that are secure against a significant fraction of cheaters is an important goal.

Second, the paper [CS03] assumes that the election officials can be trusted with constructing the pre-round schedule, which is supposed to be generated at random for NP-hardness and #P-hardness results (before and after vote elicitation, respectively). Forcing the election authorities to prove that they, indeed, used a secure random number generator rather than paired up the candidates at their will is hard to achieve in practice. On the other hand, it is clear that in many cases malicious pre-round scheduling can be used to eliminate an "undesirable" candidate or affect the election results in some other way, so if the entities responsible for scheduling are corrupted, there is a huge incentive for them to deviate from the protocol. Our approach addresses this issue by extracting the necessary randomness from the votes themselves (for a more rigorous description, see Section 3); this limits the potential for cheating by (possibly corrupt) officials. Moreover, the voters do not need to rely on any external randomness for their own actions either, since the voting scheme is completely deterministic.

The rest of the paper is organised as follows. In Section 2 we introduce our notation, give a precise definition of what it means to manipulate an election, and describe some well-known voting schemes that can be made secure using our approach. For completeness, we also provide the definition of one-way functions, and state some related facts. In Section 3, we describe our constructions for specific protocols. In Section 4, we discuss the limitations of this approach to making manipulation hard. Section 5 presents our conclusions and future research directions.

## 2 Preliminaries and Notation

We assume that there are $n$ voters and $m$ candidates and denote the set of all voters by $V = \{v_1, \dots, v_n\}$ and the set of all candidates by $C = \{c_1, \dots, c_m\}$. Our complexity results are in terms of $m$ and $n$, i.e., unless specified otherwise, 'polynomial' always means 'polynomial in $m$ and $n$'.

The set of all permutations of $C$ is denoted by $\Pi(C)$; a voter $j$'s preferences are expressed by a list $\pi_i \in \Pi(C)$: the first element is the voter's most preferred candidate, etc. In particular, this means that within one voter's preference list, ties are not allowed. A *voting scheme* is a mapping $P : \langle \Pi(C), \dots, \Pi(C) \rangle \mapsto C$ that selects a winner $c \in C$ based on all voters' preference lists.

To state our results formally, we need to define more precisely what we mean by beneficial manipulation. We distinguish between *constructive manipulation*, which is a misrepresentation of a voter's preferences that makes his top candidate an overall winner, and *destructive manipulation*, i.e., an untruthful vote that replaces the actual winner (according to the true preferences) with a candidate that the manipulator prefers over the actual winner; clearly, the second notion is strictly weaker than the first one.

We say that a voter $v_j$ can *manipulate* a protocol $P$ if he can find a permutation $\pi'_j \in \Pi(C)$ such that for some values of $\pi_i \in \Pi(C)$, $i = 1, \dots, n$, we have

1. $P(\pi_1, \ldots, \pi_n) = c$;
2. $P(\pi_1, \ldots, \pi_{j-1}, \pi'_j, \pi_{j+1}, \ldots, \pi_n) = c' \neq c$;
3. $v_j$ ranks $c'$ above $c$.

We say that $v_j$ manipulates $P$ *constructively* if $v_j$ ranks $c'$ first and *destructively* otherwise; $v_j$ manipulates $P$ *efficiently* if there is a probabilistic polynomial time algorithm that given preference lists $\pi_1, \ldots, \pi_n$ for which such $\pi'_j$ exists, constructs $\pi'_j$ with non-negligible probability (over the coin tosses of the algorithm).

We say that a set of voters $M$ with $\tau = |M|$ can $(\tau, n = n(\tau))$-*manipulate* a protocol $P$ if there is a pair of all voters' preference profiles $(\pi = (\pi_1, \ldots, \pi_n), \pi' = (\pi'_1, \ldots, \pi'_n))$, such that $\pi_i = \pi'_i$ for $i \notin M$, and everyone in $M$ strictly prefers the outcome of $P$ on $\pi'$ to the outcome of $P$ on $\pi$. The manipulation is *constructive* if everyone in $M$ ranks $P(\pi')$ first, and *efficient* if whenever such $\pi'$ exists, it can be constructed by a probabilistic polynomial time algorithm with non-negligible probability.

**Common Voting Protocols.** In this paper, we consider the following common voting protocols (in all these definitions, the candidate with the most points wins):

- *Plurality.* A candidate receives 1 point for every voter that ranks it first.
- *Borda.* For each voter, a candidate receives $m - 1$ points if it is the voter's top choice, $m - 2$ if it is the second choice, ..., 0 if it is the last.
- *Single Transferable Vote (STV).* The winner determination process proceeds in rounds. In each round, a candidate's score is the number of voters that rank it highest among the remaining candidates, and the candidate with the lowest score drops out. The last remaining candidate wins. (A vote transfers from its top remaining candidate to the next highest remaining candidate when the former drops out.)
- *Maximin.* A candidate's score in a pairwise election is the number of voters that prefer it over the opponent. A candidate's number of points is the lowest score it gets in any pairwise election.

**Pre-Round.** We reproduce the definition of *pre-round* [CS03] for reader's convenience:

1. The candidates are paired. If there is an odd number of candidates, one candidate gets a bye.
2. In each pairing of two candidates, the candidate losing the pairwise election between the two is eliminated. A candidate with a bye is never eliminated.
3. On the remaining candidates, the original protocol is executed to produce the winner. For this, the implicit votes over the remaining candidates are used.

The schedule of the pre-round is an ordering $S_m$ of $m$ candidates (it is assumed that in the pre-round, candidate $S_m(2i - 1)$ is matched with $S_m(2i)$, $i = 1, \ldots, \lfloor m/2 \rfloor$, and if $m$ is odd, $S_m(m)$ gets a bye). We denote the protocol that consists of a base protocol $P$ (such as Plurality or Borda) preceded by a pre-round that is scheduled according to $S_m$ by $S_m - P$.

**One-Way Functions.** A function $f : \{0,1\}^* \mapsto \{0,1\}^*$ is *one-way* if

- There exists a probabilistic polynomial-time algorithm (PPT) that on input $x$ outputs $f(x)$;
- For every PPT $A$ and every polynomial $p(k)$, for sufficiently large $k$ it holds that
$$P\left[f(z) = f(x) : x \overset{R}{\leftarrow} \{0,1\}^k; z \leftarrow A(1^k, f(x))\right] \leq \frac{1}{p(k)} \ .$$

Note that to prove that a function $f$ is *not* one-way it suffices to exhibit an infinite sequence $k_1, k_2, \dots$ and an efficient algorithm $A$ that inverts $f$ on inputs of length $k_i$. It is well-known that any one-way function can be transformed into a *length-preserving* one-way function, i.e., one that maps inputs of length $k$ to outputs of length $k$. Hence, assuming that one-way functions exist is equivalent to assuming that length-preserving one-way functions exist.

## 3 Reduction Based on One-Way Functions

Here we show that if one-way functions exist, then for several protocols adding a pre-round with a carefully constructed schedule makes constructive manipulation hard. We consider a family of pre-round schedules parameterised by a pair of functions $(k, f)$, where $k : \mathbb{N} \mapsto \mathbb{N}$ is any function that satisfies $k(m) < \log_2(\lfloor m/2 \rfloor!)$ and $f : \{0,1\}^* \mapsto \{0,1\}^*$ is a length-preserving function; an element of this family is denoted by $S_m^{k,f}$.

We demonstrate that if $f$ is a length-preserving one-way function and $k(m)$ is chosen in a certain way (which might be different for different base protocols), then manipulating $S_m^{k,f} - P$ is as hard as inverting $f$. In what follows, we describe the $S_m^{k,f}$ used in our construction in full detail.

**Definition of $S_m^{k,f}$.** As in [CS03], we define a match-up slot to be a space in the pre-round in which two candidates can face each other. Fix $m$ and set $k = k(m)$. Let $t$ be the smallest integer that satisfies $t! > 2^k$. Choose $2t$ candidates arbitrarily; assign $t$ of them to the first $t$ match-up slots. Pair up the remaining $m - 2t$ candidates; if there is an odd number of them, one candidate gets a bye. Assign them to the remaining slots. Now all that has to be chosen is the match-up slots for the $t$ unscheduled candidates. Renumber them from 1 to $t$. Elicit the votes.

For each voter $v_i$, $i = 1, \dots, n$, compute a string $s_i \in \{0,1\}^k$ as follows. Suppose that $v_i$ orders the unscheduled candidates as $(c_1^i, \dots, c_t^i)$. Find the lexicographic number of $(c_1^i, \dots, c_t^i)$ in the set of all permutations over $\{1, \dots, t\}$ and let $s_i$ be the last $k$ digits in the binary representation of this number. Note that since $t! > 2^k$, every string of length $k$ can be obtained in this way. Let $s = \oplus_{i=1}^n s_i$. Compute $f(s)$, and denote the permutation whose lexicographic number is $f(s)$ by $(c_{i_1}, \dots, c_{i_t})$ (again, the existence of such permutation is guaranteed since $t! > 2^k$). Assign the $c_{i_1}$th candidate to the first slot, $c_{i_2}$nd candidate to the second slot, etc. This method of pairing up the candidates implicitly describes $S_m^{k,f}$.

Our reduction is based on the following idea: we choose the preferences of non-manipulating voters so that the actual vote of the manipulators does not affect the election results apart from its use for pre-round scheduling. Furthermore, given the votes of others, there is only one way to schedule the candidates in the pre-round so that the preferred candidate will win. Hence, to achieve their goal, the manipulators must guess the pre-image of the desired permutation under the chosen one-way function.

Note that for some protocols (namely, Plurality and STV) it is possible to set $t = \lfloor m/2 \rfloor$. In this case, our pre-round scheduling method has a very natural interpretation: we separate the candidate pool into two approximately equal components, and stipulate that two candidates from the same component should not compete with each other in the pre-round. Furthermore, candidates from different components are matched randomly, where randomness is derived from the input itself (i.e., the votes), rather than from an external source.

**Results and Proofs for Specific Protocols.** In this subsection, we give three rather similar proofs for pre-round versions of Plurality, STV, and Maximin voting protocols. All of them are proven to be secure against approximately 1/6 of manipulators.

**Theorem 1.** *Assume that $m$ is even, and let $t = m/2$ and $k = \lfloor \log_2(t!) \rfloor$ (for $k \geq 80$ it must be that $t \geq 25$ and thus $m \geq 52$). Then there is a polynomial-time algorithm that can invert $f$ on inputs of length $k$ using an oracle that can constructively $(\tau, 6\tau + 5)$-manipulate $S_m^{k,f} - \mathsf{Plurality}$.*

**Corollary 1.** *If one-way functions exist, there is a pair of functions $f : \{0,1\}^* \mapsto \{0,1\}^*$ and $k : \mathbb{N} \mapsto \mathbb{N}$ such that no polynomial-time adversary can constructively $(\tau, 6\tau + 5)$-manipulate $S_m^{k,f} - \mathsf{Plurality}$ for infinitely many values of $m$.* □

Similar corollaries can be derived from other theorems in this section in a straightforward manner; we will not state them explicitly.

*Proof (of Thm. 1).* We show how to invert $f$ on a random input using an algorithm that allows $\tau$ voters to find a constructive manipulation of the protocol for $m$ candidates and $6\tau + 5$ voters whenever one exists, and carefully constructed preference lists for the $5\tau + 5$ non-manipulators. That is, we describe an algorithm that when given $Y = f(X)$, where $X$ is chosen uniformly at random from $\{0,1\}^k$, finds a $Z$ such that $f(Z) = Y$ with non-negligible probability.

First, find a permutation $(a_1, \dots, a_t)$ whose lexicographic number is $Y$. Let the $m$ candidates be

$$x_1, y_1, x_2, y_2, \dots, x_{t-1}, y_{t-1}, p, z,$$

and let each of the $\tau$ manipulators prefer $p$ to any other candidate. Assign $x_{a_1}, \dots, x_{a_{t-1}}, p$ to the first $t$ match-up slots. Set the non-manipulator votes as follows:

$$x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_{t-1}} > y_{a_{t-1}} > p > z$$
$$\text{— } 2(\tau+1) \text{ votes}$$
$$y_{a_1} > y_{a_2} > \cdots > y_{a_{t-1}} > p > x_{a_1} > \cdots > x_{a_{t-1}} > z$$
$$\text{— } 2(\tau+1) \text{ votes}$$
$$p > x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_{t-1}} > y_{a_{t-1}} > z$$
$$\text{— } \tau+1 \text{ votes.}$$

We observe the following:

(1) In the pairwise election, $y_{a_1}$ can only be eliminated by $x_{a_1}$, $y_{a_2}$ can only be eliminated by $x_{a_1}$, $y_{a_1}$, and $x_{a_2}$, etc., so to eliminate all $y_i$s, $x_{a_j}$ has to be scheduled with $y_{a_j}$ for $j = 1, \dots, t-1$, and $p$ has to be scheduled with $z$.

(2) If all $y_i$ are eliminated in the pre-round, $p$ gets at least $2\tau + 2 + \tau + 1$ votes, i.e., a majority, and wins.

(3) Suppose that some of the $y_i$ survive the pre-round. Then $y_{a_i}$ with the smallest $i$ among them gets at least $2\tau+2$ votes, while $p$ gets at most $\tau+1+\tau = 2\tau+1$ votes. Hence, in this case $p$ does not win.

Now, the rest of the reduction is straightforward. We have seen that the manipulators' vote only affects the outcome by being used for pre-round scheduling, and furthermore, $p$ only wins if all $y_i$'s are eliminated in the pre-round. Hence, to get $p$ to win, the manipulators need to order $y_1, \dots, y_{t-1}, z$ in their votes so that the resulting $s_i$'s (and, consequently, $s$) are such that when $f(s)$ is used for pre-round scheduling, $y_{a_1}$ gets assigned to the 1st slot, $y_{a_2}$ gets assigned to the 2nd slot, etc. By observing their votes, we can compute $s_i$; since all other votes are publicly known, we can also compute $s_j$, $j \neq i$, and, consequently, $s$. This $s$ satisfies $f(s) = Y$, so we have found a pre-image of $Y$ under $f$.

□

**Theorem 2.** *Assume that $m$ is even and let $t = m/2$ and $k = \lfloor \log_2(t!) \rfloor$. Then there is a polynomial-time algorithm that can invert $f$ on inputs of length $k$ using an oracle that can constructively $(\tau, 6\tau+5)$-manipulate $S_m^{k,f} - \mathsf{STV}$.*

*Proof.* Again, we are given $Y = f(X)$, where $X$ is chosen uniformly at random from $\{0,1\}^k$, and want to find a $Z$ such that $f(Z) = Y$ with non-negligible probability. We start by finding a permutation $(a_1, \dots, a_t)$ whose lexicographic number is $Y$. Let the set of candidates be

$$x_1, y_1, x_2, y_2, \dots, x_{t-1}, y_{t-1}, p, z,$$

and let each of the $\tau$ manipulators prefer $p$ to any other candidate.

Assign $x_{a_1}, \dots, x_{a_{t-1}}$ and $p$ to the first $t$ match-up slots. Set the non-manipulator votes as follows:

$$z > x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_{t-1}} > y_{a_{t-1}} > p$$
— $2(\tau+1)$ votes

$$y_{a_1} > y_{a_2} > \cdots > y_{a_{t-1}} > p > z > x_{a_1} > x_{a_2} > \ldots > x_{a_{t-1}}$$
— $2(\tau+1)$ votes

$$p > x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_{t-1}} > y_{a_{t-1}} > z$$
— $\tau+1$ votes.

We observe the following:

(1) In the pairwise election, $p$ is preferred over $z$, but not over any of the $y_i$, so, to survive the pre-round, $p$ has to be scheduled with $z$.

(2) In the pairwise election, $y_{a_1}$ can only be eliminated by $x_{a_1}$, $y_{a_2}$ can only be eliminated by $x_{a_1}$, $y_{a_1}$, and $x_{a_2}$, etc., so to eliminate all $y_i$, we have to schedule $x_{a_j}$ with $y_{a_j}$ for $j = 1, \ldots, t-1$.

(3) If all $y_i$ are eliminated in the pre-round, in the beginning of the main round $p$ has more than a half of the votes, so he wins.

(4) Suppose that some of the $y_i$ survive the pre-round. Then after $\tau$ rounds of STV the first $2\tau+2$ votes go to either $z$ or $x_1$, the highest ranking of the surviving $y_i$ gets $2\tau+2$ votes as well, and $p$ gets at most $2\tau+1$ votes. Hence, at this point $p$ will be eliminated, so he does not win the election.

The rest of the argument is as in the previous case. The total number of voters is $n = 6\tau + 5$. □

**Theorem 3.** *Suppose that $m$ is even and let $t = \lfloor m/2 \rfloor - 4$ and $k = \lfloor \log_2(t!) \rfloor$. Then there is a polynomial-time algorithm that can invert $f$ on inputs of length $k$ using an oracle that can constructively $(\tau, 6\tau+5)$-manipulate $S_m^{k,f}$ – Maximin.*

*Proof.* Again, we are given $Y = f(X)$, where $X$ is chosen uniformly at random from $\{0,1\}^k$, and want to find a $Z$ such that $f(Z) = Y$ with non-negligible probability. We start by finding a permutation $(a_1, \ldots, a_t)$ whose lexicographic number is $Y$. Let the set of candidates be

$$x_1, y_1, x_2, y_2, \ldots, x_t, y_t, p, z_1, z_2, z_3,$$

and let each of the $\tau$ manipulators prefer $p$ to any other candidate.

Assign $x_{a_1}, \ldots, x_{a_t}$ to the first $t$ match-up slots. Pair up $p$, $z_1$, $z_2$, and $z_3$, and assign them to the last two slots. Set the non-manipulator votes as follows:

$$x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_t} > y_{a_t} > p > z_1 > z_2 > z_3$$
— $2(\tau+1)$ votes

$$z_1 > z_2 > z_3 > y_{a_1} > y_{a_2} > \cdots > y_{a_t} > p > x_{a_1} > x_{a_2} > \ldots > x_{a_t}$$
— $2(\tau+1)$ votes

$$p > z_1 > z_2 > z_3 > x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_t} > y_{a_t}$$
— $\tau+1$ votes.

We observe the following:

(1) Both $p$ and exactly one of the $z_i$, which we denote by $z$, survive the pre-round.
(2) In the pairwise election, $y_{a_1}$ can only be eliminated by $x_{a_1}$, $y_{a_2}$ can only be eliminated by $x_{a_1}$, $y_{a_1}$, and $x_{a_2}$, etc., so to eliminate all $y_i$, we have to schedule $x_{a_j}$ with $y_{a_j}$ for $j = 1, \ldots, t$.
(3) Suppose that any of the $y_i$ survives the pre-round. Then $p$'s score is at most $2\tau + 1$ (there are only $\tau + 1$ honest voters that prefer it over any of the remaining $y_i$), and $z$'s score is at least $2\tau + 2$ (there are $2\tau + 2$ honest voters that rank it first), so $p$ cannot win the elections.
(4) Suppose that none of the $y_i$ survives the pre-round. Then $p$'s score is at least $3\tau + 3$ (there are $3\tau + 3$ honest voters that prefer it over any of the $x_i$ and $3\tau + 3$ honest voters that prefer it over $z$), $z$'s score is at most $3\tau + 2$ (there are only $2\tau + 2$ honest voters that prefer it over $p$), and the score of any of the $x_i$ is at most $3\tau + 2$, since there are only $2\tau + 2$ honest voters that prefer it over $z$. Hence, in this case $p$ wins.

The rest of the argument is as in the previous case. The total number of voters is $n = 6\tau + 5$. □

While the previous results guaranteed security against a constant fraction of manipulators, in the case of the Borda protocol the allowable fraction of manipulators depends on the total number of candidates.

**Theorem 4.** *Let $m > 4$ be the number of candidates and $n$ be the number of voters. Suppose that $f$ is a one-way function, $m - 4t^2 - 8t + 1 > 0$, $m = \mathrm{poly}(t)$, and $k = \lfloor \log_2(t!) \rfloor$ (for $k \geq 80$ it must be that $t \geq 25$ and thus $m \geq 1300$). Then there is a polynomial-time algorithm that can invert $f$ using an oracle that can find a constructive $(\tau, (m + 4t + 8)\tau)$-manipulation of $S_m^{k,f} - \mathsf{Borda}$ whenever one exists.*

*Proof.* For Borda protocol, set $d = m - 2t - 1$, and let the set of candidates be

$$x_1, y_1, \ldots, x_t, y_t, p, z_1, \ldots, z_d,$$

and there are $\tau$ manipulators who rank $p$ first.

Assign $x_{a_1}, \ldots, x_{a_t}$ to the first match-up slots. Pair up $p, z_1, \ldots, z_d$ and assign them to the remaining slots. Set the non-manipulator votes as follows:

$$x_{a_1} > y_{a_1} > \ldots > x_{a_t} > y_{a_t} > p > z_1 > \ldots > z_d \text{ — } \alpha \text{ votes,}$$
$$p > y_{a_1} > \ldots > y_{a_t} > z_1 > \ldots > z_d > x_{a_1} > \ldots > x_{a_t} \text{ — } \beta \text{ votes,}$$

where $\alpha$ and $\beta$ are to be determined later. Assume for convenience that $m$ is even. Discarding the votes of $\tau$ manipulators, we observe the following:

(1) The preferred candidate $p$ survives the pre-round, and in the main round, $p$ gets more points than any of the surviving $z_i$.

(2) If all $y_i$ are eliminated in the pre-round (and hence all $x_i$ survive), $x_{a_1}$ gets $\left(\frac{m}{2}-1\right)\alpha+(t-1)\beta$ points and other $x_i$ get fewer points, while $p$ gets $\left(\frac{m}{2}-t-1\right)\alpha+\left(\frac{m}{2}-1\right)\beta$ points. Thus $p$ wins as soon as $\left(\frac{m}{2}-t\right)\beta-t\alpha>0$.

(3) If any of the $y_i$ is not eliminated in the pre-round, then in the main round the highest-ranking of the surviving $y_i$, which we denote by $y_{i_0}$, gets at least $\alpha-\beta$ more points than $p$ (the first $\alpha$ votes rank $y_{i_0}$ higher than $p$, and in the last $\beta$ votes, after we delete all $y_i$ that were eliminated in the pre-round, $y_{i_0}$ immediately follows $p$). Hence, if $\alpha-\beta>0$, $p$ cannot win.

(4) In the pairwise election, as long as $\alpha-\beta>0$, $y_{a_1}$ can only be eliminated by $x_{a_1}$, $y_{a_2}$ can only be eliminated by $x_{a_1}, y_{a_1}$ and $x_{a_2}$, etc, so to eliminate all $y_i$ we have to schedule $x_{a_i}$ with $y_{a_i}$ for $i=1,\dots,t$.

Moreover, if we replace 0 with $(m-1)\tau$ in the right-hand side of the inequalities in (2)–(4), then these properties hold even if we add $\tau$ additional votes.

Hence, we would like to choose $\alpha$ and $\beta$ so that $(m/2-t)\beta-t\alpha>(m-1)\tau$, $\alpha-\beta>(m-1)\tau$, and $\alpha+\beta$ is as small as possible. From the second condition, it is clear that $\alpha+\beta=\Omega(m\tau)$; choosing $\alpha=(m+2t+4)\tau$, $\beta=(2t+4)\tau$ matches this lower bound and satisfies both conditions provided that $m>4t^2+8t-1$. □

## 4 Limitations

The examples constructed in the previous section (as well as the ones in [CS03]) show that it is possible to construct a voting scheme that does not allow for a universal polynomial-time algorithm for finding a beneficial manipulation (under standard assumptions, such as $P\neq NP$).

However, this does not exclude the possibility that in many contexts the manipulator can figure out what to do. It would be desirable to have a voting scheme with the following property: for any voter and any vector of other voters' preference profiles finding an action that is always no worse and sometimes better than truthfully reporting your preferences is hard. The appropriate notion of hardness would be some flavour of hardness on average, such as inverting one-way functions. Moreover, we can relax this criterion by requiring it to hold with an overwhelming probability over the honest voters' preference profiles rather than all preference profiles (note, however, that to formalise this, we need to know the distribution of the voters' preferences).

Unfortunately, it turns out that this goal is impossible to achieve by simply adding a pre-round. To formally show this, we construct an example in which a (destructive) manipulation is always easy to find. Namely, we demonstrate a family of preferences profiles such that

- if everyone votes honestly, then under any pre-round schedule candidate $p$ survives the pre-round and goes on to win the elections;
- there is a manipulation by a single voter such that for any pre-round schedule the result of the elections is a draw between $p$ and some other candidate (and the manipulator prefers this candidate to $p$).

Our example is constructed for Plurality protocol, but it is possible to construct similar examples for other protocols as well.

Suppose that $m \geq 8$ is even, and set $t = m/2 - 2$. Let the set of candidates be $p, a, b, c_1, c_2, \ldots, c_{2t+1}$. Choose an arbitrary $k$ so that $n/3 + 1 < k < n/2$. Suppose that the honest voters can be divided into three groups

- $k + 1$ honest voters whose votes are of the form
$$p > a > b > c_{j,i_1} > \cdots > c_{j,i_{2t+1}},$$
where $j = 1, \ldots, k + 1$, and each $(c_{j,i_1}, \ldots, c_{j,i_{2t+1}})$ is a permutation of $c_1, \ldots, c_{2t+1}$;
- $k$ honest voters whose votes are of the form
$$a > b > p > c_{j,i_1} > \cdots > c_{j,i_{2t+1}},$$
where $j = 1, \ldots, k$, and each $(c_{j,i_1}, \ldots, c_{j,i_{2t+1}})$ is a permutation of $c_1, \ldots, c_{2t+1}$;
- $n - 2k - 2$ honest voters whose votes are of the form
$$c_{j,i_1} > \cdots > c_{j,i_{2t+1}} > p > a > b,$$
where $j = 1, \ldots, n - 2k - 2$, and each $(c_{j,i_1}, \ldots, c_{j,i_{2t+1}})$ is a permutation of $c_1, \ldots, c_{2t+1}$.

Suppose also that the manipulator's (honest) preference list is

$$c_{i_1} > \cdots > c_{i_{2t+1}} > a > b > p,$$

where $(c_{i_1}, \ldots, c_{i_{2t+1}})$ is a permutation of $c_1, \ldots, c_{2t+1}$.

Observe the following:

1. No matter how the manipulator votes, $p$ always survives the pre-round.
2. If either of $a$ or $b$ is not matched with $p$ in the pre-round, he survives the pre-round, so at least one of them participates in the main round.
3. At least one of $c_i$ survives the pre-round.

Hence, if everyone votes honestly, after the pre-round there will be $k+1$ votes for $p$, $k$ votes for $a$ or $k$ votes for $b$, and at most $n - 2k - 1 < k$ votes for any of the $c_i$. However, if the manipulator puts $a$ and $b$ on the top of his list, i.e., votes

$$a > b > c_1 > \cdots > c_{2m+1} > p,$$

he can achieve a draw between $a/b$ and $p$. Unless the draws are always resolved so that $p$ wins, this strictly improves the outcome of the elections from the manipulator's point of view, and in the latter case we can modify the example by increasing the number of honest voters who rank $a$ first and $b$ second to $k+1$, in which case the manipulator can change the situation from a draw between $p$ and another candidate to a win by another candidate.

While under uniform distribution of preferences, the likelihood of this type of profile is not very high, the uniformity assumption itself is hardly applicable to real-life scenarios. In a more polarised society, this preference profile has a natural interpretation: suppose that there are three established parties, two of which have similar positions on many issues, and a multitude of independent candidates, and the voters can be divided into two groups: traditionalists, who do not trust any of the independent candidates, and protesters, who rank the established candidates after the independent candidates. Under some additional assumptions, the situation described above becomes quite likely.

**Manipulation for Small Number of Candidates.** Clearly, when the number of candidates $m$ is constant, and there is only one manipulator, he can easily figure out what to do simply by checking the outcome for all $m!$ possible votes and submitting the one that produces the best possible result. When the number of manipulators is large, as is the case in our scenario, enumerating the space of all possible votes by the manipulating coalition becomes infeasible (even if we assume that all voters are treated symmetrically, the size of this space is still exponential in the coalition size), so it might still be the case that choosing the best possible action is hard. However, if the manipulators' goal is simply to submit a set of votes that results in a specific pre-round schedule, and the scheduling algorithm treats all candidates symmetrically, then, since the number of possible pre-round schedules is constant, this goal is likely to be attained by random guessing. Therefore, making manipulation infeasible when the number of candidates is small cannot be achieved by this method.

## 5 Conclusions and Future Research

Our work extends the results of [CS02, CS03] in several important directions. All our improvements address important concerns in the field of secure voting systems. First, we show that our hardness results hold against a large fraction of manipulating voters (rather than a single voter). Also, while the original protocol of [CS03] makes it possible for dishonest election authorities to affect the results by constructing the pre-round schedule in a way that suits their goals (rather than randomly), we eliminate this loophole by making the schedule dependent on the contents of the voters' ballots. Finally, voters do not need to trust any external randomness since their voting procedure is completely deterministic; in a certain sense, our pre-round construction extracts randomness from the votes.

It is important to note that our methodology, as well as the one of [CS03] works for a wide range of protocols: while some voting procedures are inherently hard to manipulate, they may not reflect the decision-making procedures that are acceptable in a given culture, and may thus be deemed inappropriate. On the other hand, a pre-round followed by an execution of the original protocol retains many of the desirable properties of the latter. All of the voting protocols described in Section 2, as well as many others, are used in different contexts;

it is unreasonable to expect that all of them will be replaced, say, by STV just because it is harder to manipulate.

Note also that while our results have been worded in the terms of polynomial time, it is relatively simple to estimate the tightness of the reductions. This is since in all four cases, the attacker that inverts $f$ invites the $S_m^{k,f}$ – Mechanism-breaking oracle only once.

In Section 4, we discuss the limitations of the current approach, showing that it cannot be used to achieve certain very desirable hardness criteria. We leave it as an open problem whether there are other methods that satisfy these criteria, or whether there is a less ambitious set of desiderata that is acceptable in practice. Another question relates to the fraction of manipulators against which our system is secure: it would be interesting to raise this threshold from 1/6th fraction of the voters for Plurality, STV, and Maximin, and $1/m$th for Borda, or to show that this is impossible.

## References

[BO91] John J. Bartholdi, III and James B. Orlin. Single Transferable Vote Resists Strategic Voting. *Social Choice and Welfare*, 8(4):341–354, 1991.

[CS02] Vincent Conitzer and Tuomas Sandholm. Complexity of Manipulating Elections with Few Candidates. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence*, pages 314–319, Edmonton, Alberta, Canada, July 28 — August 1 2002. AAAI Press.

[CS03] Vincent Conitzer and Tuomas Sandholm. Universal Voting Protocol Tweaks to Make Manipulation Hard. In *In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 781–788, Acapulco, Mexico, August 9–15 2003.

[Gib73] Allan F. Gibbard. Manipulation of Voting Schemes: A General Result. *Econometrica*, 41:597–601, 1973.

[Sat73] Mark A. Satterthwaite. *The Existence of Strategy-Proof Voting Procedures: A Topic in Social Choice Theory.* PhD thesis, University of Wisconsin, Madison, 1973.

# Efficient Privacy-Preserving Protocols for Multi-unit Auctions

Felix Brandt[1] and Tuomas Sandholm[2]

[1] Stanford University, Stanford CA 94305, USA
brandtf@cs.stanford.edu
[2] Carnegie Mellon University, Pittsburgh PA 15213, USA
sandholm@cs.cmu.edu

**Abstract.** The purpose of multi-unit auctions is to allocate identical units of a single type of good to multiple agents. Besides well-known applications like the selling of treasury bills, electrical power, or spectrum licenses, multi-unit auctions are also well-suited for allocating CPU time slots or network bandwidth in computational multiagent systems. A crucial problem in sealed-bid auctions is the lack of trust bidders might have in the auctioneer. For one, bidders might doubt the correctness of the auction outcome. Secondly, they are reluctant to reveal their private valuations to the auctioneer since these valuations are often based on sensitive information. We propose privacy-preserving protocols that allow bidders to jointly compute the auction outcome without the help of third parties. All three common types of multi-unit auctions (uniform-price, discriminatory, and generalized Vickrey auctions) are considered for the case of marginal decreasing valuation functions. Our protocols are based on distributed homomorphic encryption and can be executed in a small constant number of rounds in the random oracle model. Security merely relies on computational intractability (the decisional Diffie-Hellman assumption). In particular, no subset of (computationally bounded) colluding participants is capable of uncovering private information.

## 1 Introduction

Auctions are not only wide-spread mechanisms for selling goods, they have also been applied to a variety of computer science settings like task assignment, bandwidth allocation, or finding the shortest path in a network with selfish nodes. A crucial problem in sealed-bid auctions is the lack of trust bidders might have in the auctioneer. For one, bidders might doubt the correctness of the auction outcome. Secondly, they are reluctant to reveal their private valuations to the auctioneer since these valuations are often based on sensitive information. We tackle both problems by providing cryptographic protocols that allow bidders to jointly compute the auction outcome without revealing any other information.

More specifically, our setting consists of one seller and $n$ bidders that intend to come to an agreement on the selling of $M$ indistinguishable units of a particular

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 298–312, 2005.
© Springer-Verlag Berlin Heidelberg 2005

type of good.[1] Each bidder submits a vector of $M$ sealed bids $(b^i_1, b^i_2, \ldots, b^i_M)$ to the auctioneer, expressing how much he is willing to pay for each additional unit. In other words, $\sum_{j=1}^{m} b^i_j$ is the amount bidder $i$ is willing to pay for $m$ units. A common assumption that we also make is that bidders have *marginal decreasing valuations*, *i.e.*, $b^i_1 \geq b^i_2 \geq \cdots \geq b^i_M$. This is justified by the fact that bidders usually want to pay less for each additional unit the more units they already have.[2] The auctioneer then clears the auction by allocating units to the bidders that value them most. Let $W$ be the set of winning bids, *i.e.*, the set containing the $M$ highest bids. Clearly, if bidder $i$ submitted $m^i$ bids that belong to $W$, any economically efficient auction should allocate $m^i$ units to bidder $i$. There are three common ways of pricing units that are sold in multi-unit auctions: uniform-price, discriminatory, and generalized Vickrey (see *e.g.*, [Kri02] or [Kle99]).

– *Uniform-Price Auction*
  All bidders pay the same price per unit, given by the $(M + 1)$st-highest bid.
– *Discriminatory Auction*
  The discriminatory auction is the natural extension of the $1^{\text{st}}$-price sealed-bid auction (for one unit) to the case of $M$ units. Every bidder pays exactly what he bid for each particular unit he receives. In other words, if bidder $i$ receives $m^i$ units, he pays $\sum_{j=1}^{m^i} b^i_j$.
– *Generalized Vickrey Auction*
  The generalized Vickrey auction is an extension of the Vickrey (or $2^{\text{nd}}$-price sealed-bid) auction. A bidder that receives $m$ units pays the sum of the $m$ highest losing bids submitted by other bidders, *i.e.*, excluding his own losing bids. This auction format belongs to the praised family of VCG mechanisms [Vic61, Cla71, Gro73] and provides various desirable theoretical properties.

There is an ongoing debate in economic theory which auction format is most favorable. For example, the uniform-price auction is sometimes rejected because it suffers from an effect called *demand reduction* which states that bidders are better off reducing their bids for additional units. In contrast to both other auction types, the generalized Vickrey auction is *economically efficient*, *i.e.*, the total welfare of all bidders is maximized in a strategic equilibrium, and *strategy-proof*, *i.e.*, each bidder is best off bidding his true valuations no matter what other bidders do. On the other hand, the generalized Vickrey auction is vulnerable to strategic collusion and can result in outcomes that might be considered unfair. Summing up, it seems as if different application scenarios require different auction types. For example, the US government began to use uniform-price auctions to sell treasury bills in 1992, after a long tradition of discriminatory auctions. On the other hand, UK electricity generators switched from uniform-price to

[1] All the presented protocols also work for procurement or so-called reverse auctions where there is one buyer and multiple sellers.

[2] However, this is not always the case. For instance, in a tire auction, a car owner might value the forth tire higher than the third.

discriminatory auctions in 2000. A detailed discussion of the pros and cons of multi-unit auctions is beyond the scope of this paper (see *e.g.*, [Kri02] or [Kle99] for further information).

In this paper, we propose cryptographic protocols for all three common types of multi-unit auctions. These protocols allow bidders to "emulate" a virtual auctioneer, thus enabling privacy of bids without relying on third parties. The only information revealed in addition to the auction outcome is minor statistical data in the case of certain ties (*e.g.*, the number of tied bids). As round efficiency is usually considered to be the most important complexity measure in a distributed setting, the main goal when designing these protocols was to minimize the number of rounds required for executing the protocols. In fact, all our protocols only need a low constant number of rounds in the random oracle model. Communication and computation complexity, on the other hand, is linear in the number of different prices. Nevertheless, the proposed protocols should be practically feasible for moderately sized scenarios.

The remainder of this paper is structured as follows. In Section 2, we describe the general security model underlying this work. Recent related research on cryptographic auction protocols is reviewed in Section 3. In Section 4, we give a detailed description of the vector notation and order statistic subprotocol to be used in the multi-unit auction protocols presented in Section 5. Concrete implementation details regarding El Gamal encryption and efficient (honest-verifier) zero-knowledge proofs are discussed in Section 6. The paper concludes with an overview of the obtained results in Section 7.

## 2 Security Model

Our primary goal is privacy that cannot be broken by any coalition of third parties or bidders. For this reason, we advocate a security model in which bidders themselves jointly compute the auction outcome so that any subset of bidders is incapable of revealing private information. Clearly, extensive interaction by bidders is undesirable in practice (but unavoidable given our objective). In order to minimize interaction, our secondary goal is to keep round complexity at a minimum (*i.e.*, small constants). The main drawbacks implied by our setting are low resilience and high computational and communication complexity. However, auctions that require such a high degree of privacy typically take place with few, well-known (*i.e.*, non-anonymous) bidders, for instance when auctioning off spectrum licenses.

We consider cryptographic protocols for $n$ bidders and one seller. Each bidder possesses a private input consisting of $M$ bids. Agents engage in a multi-party protocol to jointly and securely compute the outcome function $f$. In our context, security consists of correctness ($f$ is computed correctly) and full privacy (aka. $(n-1)$-privacy, *i.e.*, no subset of agents learns more information than what can be inferred from the outcome and the colluding agents' private inputs). When allowing premature protocol abort, any such function $f$ can be computed securely and fairly when trapdoor permutations exist, and a designated agent does not quit or reveal information prematurely.[3] In the auction protocols presented in this paper, the seller will take the role of the designated agent. It is important to note that even when the seller quits or reveals information early, the worst thing that can happen is that an agent learns the outcome and quits the protocol before the remaining agents were able to learn the outcome.[4] Bid privacy is not affected by premature abort.

Whenever a malicious bidder disrupts the protocol by sending faulty messages or failing to prove the correctness of his behavior in zero-knowledge, this bidder will be removed, and the protocol will be restarted (termination is guaranteed after at most $n-1$ iterations). We presume that the "public" is observing the protocol and therefore a malicious bidder can undoubtedly be identified, independently of how many remaining agents are trustworthy. As malicious bidders can easily be fined and they do not gain any information, there should be no incentive to disrupt the auction and we henceforth assume that a single protocol run suffices.

## 3 Related Work

Numerous cryptographic protocols for *single-unit* auctions have been proposed in the literature (*e.g.*, [AS02, BS01, Bra03a, Di 00, JS02, Kik01, LAN02, NPS99]). We follow our previous approach [Bra03a] where bidders jointly compute the auction outcome without the help of trusted third parties.

There are few privacy-preserving protocols for the selling of more than just a single good. Suzuki et al [SY02, SY03] proposed protocols for general *combinatorial auctions* (see *e.g.*, [CSS05]), where bidders can bid on arbitrary combinations of items for sale, based on a secure dynamic programming subprotocol. The problem of determining the winners in this type of auction is $\mathcal{NP}$-complete. Clearly, adding cryptographic overhead to winner determination results in protocols whose complexity is prohibitively large for most practical settings. Multi-unit auctions, in which a specific number of identical units of a single item is sold, are an important, yet still intractable [SS01], subcase of combinatorial auctions. Instead of bidding on every conceivable combination of items, bidders simply specify their willingness to pay for any number of units. In contrast to general combinatorial auctions, multi-unit auctions are already widely used, *e.g.*, for selling treasury bills or electrical power. Suzuki et al formulate the winner determination problem in multi-unit auctions as a dynamic programming optimization problem, thus enabling their secure dynamic programming protocol to compute the optimal allocation of units [SY02, SY03]. However, when making the reasonable assumption that bidders' valuations are *marginal decreasing* in

[3] This useful restriction to circumvent fairness problems was also used in our previous work (*e.g.*, [Bra03b, Bra03a]). Independently, the security of such a model was generally analyzed by Goldwasser et al [GL02].

[4] Another common way to obtain fairness without a trusted majority is the gradual release of secrets (*e.g.*, [Yao86, GL90].

the number of units, *i.e.*, the $(m+1)$th unit a bidder receives is never more valuable to him than the $m$th unit, computing the optimal allocation of units becomes tractable [Ten00], thus making computationally demanding techniques like dynamic programming unnecessary. To the best of our knowledge, cryptographic protocols for multi-unit auctions with marginal decreasing valuations have only been presented for the considerably simple subcase where each bidder only demands a single unit [AS02, Bra03a, Kik01].[5]

Parallel to our work on fully private auction and social choice protocols (*e.g.*, [Bra02, Bra03b, BS04b, BS04a]), there is an independent, yet quite similar, stream of research on self-tallying elections [KY02, KY03, Gro04]. In both settings, agents jointly determine the outcome of a social choice function without relying on trusted third parties. What we call "full privacy" is termed "perfect ballot secrecy" in Kiayias et al's work. Similarly, the terms "self-tallying" and "dispute-free" [KY02] can be translated to "bidder-resolved" and "weakly robust" [Bra02], respectively. In order to achieve fairness, both approaches assume a weakly trustworthy party (a "dummy voter" and the auction seller, respectively). Besides these similarities, Kiayias et al's approach mainly differs in the emphasis of non-interactiveness (once the random-generating preprocessing phase is finished) while computing rather simple outcome functions (*e.g.*, the sum of input values).

## 4 Building Blocks

Distributed homomorphic encryption allows agents to efficiently add secret values without extensive interaction. For this reason, our protocols only require the computation of *linear combinations* of secret inputs values (which can be solely based on addition) and *multiplications with jointly created random numbers* (for which we propose an efficient sub-protocol in Section 6.1). When computing on *vectors* of secrets, the computation of linear combinations enables the addition (and subtraction) of secret vectors, and the multiplication of vectors with predefined known matrices. Furthermore, the vector representation allows for efficient zero-knowledge proofs of correctness.

### 4.1 Vector Representation

Let $\boldsymbol{p}$ be a vector of $k$ possible prices (or valuations), $\boldsymbol{p} = (p_1, p_2, \ldots, p_k)$, and $bid \in \{1, 2, \ldots, k\}$ a bid. The *bid vector* $\boldsymbol{b}$ of this bid is defined so that component $b_{bid} = 1$ (the bidder bids $p_{bid}$) and all other components are 0. This representation allows efficient proofs of the vector's correctness by showing $\forall j \in \{1, 2, \ldots, k\} : \; b_j \in \{0, 1\}$ and $\sum_{j=1}^{k} b_j = 1$ (see Section 6 for details). Yet, the main advantage of the vector representation is the possibility to efficiently perform certain computations. For example, the "integrated" bid

[5] In this so-called *unit demand* case, the uniform-price and the generalized Vickrey auction collapse to the same auction type: the $(M+1)$st-price auction.

vector $\boldsymbol{b}'$ (a notion introduced in [AS02]) can be derived by multiplying the bid vector with the $k \times k$ lower triangular matrix $\mathsf{L}$.[6]

$$\boldsymbol{b} = \begin{pmatrix} b_k \\ \vdots \\ b_{bid-1} \\ b_{bid} \\ b_{bid+1} \\ \vdots \\ b_1 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \qquad \boldsymbol{b}' = \mathsf{L}\,\boldsymbol{b} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \text{ where } \mathsf{L} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ 1 & \cdots & \cdots & 1 \end{pmatrix}$$

The price we pay for round-efficiency enabled by this unary representation is communication and computation complexity that is linear in the number of different prices $k$. On the other hand, the unary notation allows us to easily adapt the given protocols to emulate *iterative* (*e.g.*, ascending-price or descending-price) auctions (see *e.g.*, Chapter 2 of [CSS05]) in which bidders gradually express their unit demand for sequences of prices. In fact, there are common iterative equivalences for each of the three sealed-bid auction mechanisms considered in this paper: the multi-unit English auction (uniform-price), the multi-unit Dutch auction (discriminatory), and the Ausubel auction (generalized Vickrey). Iterative auctions are sometimes preferred over sealed-bid auctions because bidders are not required to exhaustively determine their valuations and because they can lead to higher revenue if valuations are interdependent.

### 4.2 Order Statistic Subprotocol

The most essential building block of our auction protocols is a subprotocol that determines the $m$th order statistic, *i.e.*, the $m$th highest bid, in a given vector of $N$ bids. Some $k \times k$ matrices that we will use in addition to $\mathsf{L}$ are the upper triangular matrix $\mathsf{U}$, the identity matrix $\mathsf{I}$, and random multiplication matrices $\mathsf{R}^*$. Furthermore, we will utilize the $k$-dimensional unit vector $\boldsymbol{e}$.

$$\mathsf{U} = \begin{pmatrix} 1 & \cdots & \cdots & 1 \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \end{pmatrix}, \quad \mathsf{I} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}, \quad \mathsf{R}^* = \begin{pmatrix} * & 0 & \cdots & 0 \\ 0 & * & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & * \end{pmatrix}, \quad \boldsymbol{e} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$$

The components on the diagonal of $\mathsf{R}^*$ are random numbers unknown to the agents. They are jointly created using a special sub-protocol. Multiplication with $\mathsf{R}^*$ turns all vector components *that are not zero* into meaningless random numbers. For this reason, it is usually a final masking step in our protocols.

[6] Please note that matrices are only used to facilitate the presentation. The special structure of all used matrices allows us to compute matrix-vector multiplications in $\mathcal{O}(k)$ steps.

Our approach to detect the $m$th-highest bid requires special techniques if there is a tie at the $m$th-highest bid. Information that is revealed in case of a tie is the number of tied bids ($t$) and the number of bids that are greater than the $m$th-highest bid ($u$). Let us for now assume that there is always a *single* $m$th-highest bid ($t = 1$ and $u = m - 1$). When given vector $\boldsymbol{B}$ where each component of $\boldsymbol{B}$ denotes the number of bids at the corresponding price (see Example 1), we will specify how to compute a vector that merely reveals the $m$th-highest bid.

$$\boldsymbol{stat}^{m}_{1,m-1}(\boldsymbol{B}) = \Big((2\mathsf{L} - \mathsf{I})\boldsymbol{B} - (2m-1)\boldsymbol{e}\Big)\mathsf{R}^*$$

yields a vector in which the component denoting the $m$th-highest bid is zero. All other components are random values.

*Example 1.* Let the vector of possible prices be $\boldsymbol{p} = (10, 20, 30, 40, 50, 60)$ and consider the computation of the second highest bid ($m = 2$) in a vector that represents bids 20 and 50:

$$\boldsymbol{B} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

All computations take place in the finite field $\mathbb{Z}_{11}$. Asterisks denote arbitrary random numbers that have no meaning to bidders.

$$\boldsymbol{stat}^{2}_{1,1}(\boldsymbol{B}) = \left(\begin{pmatrix} 1&0&0&0&0&0 \\ 2&1&0&0&0&0 \\ 2&2&1&0&0&0 \\ 2&2&2&1&0&0 \\ 2&2&2&2&1&0 \\ 2&2&2&2&2&1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \end{pmatrix}\right)\mathsf{R}^* = \left(\begin{pmatrix} 0 \\ 1 \\ 2 \\ 2 \\ 3 \\ 4 \end{pmatrix} - \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \end{pmatrix}\right)\mathsf{R}^* = \begin{pmatrix} 8 \\ 9 \\ 10 \\ 10 \\ 0 \\ 1 \end{pmatrix}\mathsf{R}^* = \begin{pmatrix} * \\ * \\ * \\ * \\ 0 \\ * \end{pmatrix}$$

The resulting vector $\boldsymbol{stat}^{2}_{1,1}(\boldsymbol{B})$ indicates that the second highest bid is 20. ▲

When two or more bids qualify as the $m$th-highest bid (because they are equal), the technique described above does not work ($\boldsymbol{stat}^{m}_{1,m-1}(\boldsymbol{B})$ contains no zeros). For this reason, we compute additional vectors that yield the correct outcome in the case of such a tie. The following method marks the $m$th-highest bid while not revealing any information about other ties. Subtracting $t\boldsymbol{e}$ from input vector $\boldsymbol{B}$ yields a vector that contains zeros if there is a tie of $t$ bids ($1 < t \le N$ where $N$ is the number of bids). As we are only interested in ties involving the $m$th-highest bid, other ties are masked by adding $(N+1)\,(\mathsf{L}\,\boldsymbol{B} - (t+u)\boldsymbol{e})$ where $u \in \{\max(0, m-t), \ldots, \min(m-1, N-t)\}$ for each $t$. The resulting vector contains a zero when $t$ bids are equal and there are $u$ bids higher than the tie. The preceding factor $(N+1)$ is large enough to ensure that both addends do not add up to zero. Finally, in the case of a tie, the $m$th-highest bid can be determined by computing the following additional vectors.

$$\boldsymbol{stat}^{m}_{t,u}(\boldsymbol{B}) = \Big(\boldsymbol{B} - t\boldsymbol{e} + (N+1)\,(\mathsf{L}\,\boldsymbol{B} - (t+u)\boldsymbol{e})\Big)\mathsf{R}^*$$

*Example 2.* Suppose that two bids are 50 and two are 20 ($m = 2$, computation takes place in $\mathbb{Z}_{11}$ and $\boldsymbol{p} = (10, 20, 30, 40, 50, 60)$):

$$\boldsymbol{B} = \begin{pmatrix} 0 \\ 2 \\ 0 \\ 0 \\ 2 \\ 0 \end{pmatrix}$$

$\boldsymbol{stat}^2_{1,1}(\boldsymbol{B})$ yields no information due to the tie at price 50. The first two ($t = 2, u \in \{0, 1\}$) additional order statistic vectors look like this:

$$\boldsymbol{stat}^2_{2,0}(\boldsymbol{B}) = \left( \begin{pmatrix} 0 \\ 2 \\ 0 \\ 2 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix} + 5 \left( \begin{pmatrix} 0 \\ 2 \\ 2 \\ 4 \\ 4 \\ 4 \end{pmatrix} - \begin{pmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix} \right) \right) R^* = \begin{pmatrix} 10 \\ 0 \\ 9 \\ 10 \\ 8 \\ 8 \end{pmatrix} R^* = \begin{pmatrix} * \\ 0 \\ * \\ * \\ * \\ * \end{pmatrix}$$

$$\boldsymbol{stat}^2_{2,1}(\boldsymbol{B}) = \left( \begin{pmatrix} 0 \\ 2 \\ 0 \\ 2 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix} + 5 \left( \begin{pmatrix} 0 \\ 2 \\ 2 \\ 4 \\ 4 \\ 4 \end{pmatrix} - \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \end{pmatrix} \right) \right) R^* = \begin{pmatrix} 5 \\ 6 \\ 4 \\ 5 \\ 3 \\ 3 \end{pmatrix} R^* = \begin{pmatrix} * \\ * \\ * \\ * \\ * \\ * \end{pmatrix}$$

For $t > 2$ the first difference contains no zeros, leading to random vectors. The $m$th-highest bid is indicated in vector $\boldsymbol{stat}^2_{2,0}(\boldsymbol{B})$ (revealing that the two highest bids are equal). ▲

Concluding, in order to obtain the $m$th order statistic of $N$ bids, agents jointly compute function $\boldsymbol{stat}^m_{t,u}(\boldsymbol{B})$ where $t = \{1, 2, \ldots, N\}$ and $u \in \{\max(0, m - t), \ldots, \min(m - 1, N - t)\}$ for each t. Thus, a total amount of $m(N - m)$ vectors of size $k$ needs to be computed.

## 5 Multi-unit Auction Protocols

In this section, we present methods to compute the outcome of three common multi-unit auction types based on the vector notation and the order statistic subprotocol proposed in the previous section.

Before determining the auction outcome, bidders have to prove that their bids are marginal decreasing, *i.e.*, $b^i_m \geq b^i_{m+1}$ for each $m < M$. This can be achieved by computing

$$\boldsymbol{dec}^i_m = \left( \mathsf{L}\, \boldsymbol{b}^i_m + (\mathsf{U} - \mathsf{I})\, \boldsymbol{b}^i_{m+1} \right) \mathsf{R}^i$$

where $\mathsf{R}^i$ is a random matrix chosen by bidder $i$ and $\boldsymbol{b}^i_m$ is bidder $i$'s bid vector for the $m$th unit. Each $\boldsymbol{dec}^i_m$ is jointly decrypted. If any component equals zero, bidder $i$ submitted malformed, *i.e.*, increasing, bids. There is no $\mathsf{R}_i$ bidder $i$ could use to hide the fact that one of the components is zero.

As noted in Section 1, the three auction formats only differ in the pricing of units. The *number* of units each bidder receives is identical in all three auction types. The number of units $m^i$ that bidder $i$ receives can be determined by computing a vector where the component denoting the $M$th-highest bid is zero and then adding all integrated bid vectors of bidder $i$. This yields a vector whose components are random except for the single component containing the number of units bidder $i$ receives. In order to squeeze all $m^i$ in the same vector $\boldsymbol{alloc}_{t,u}$, we represent the allocation of units as a base-$(M+1)$ number.[7] Furthermore, bidders jointly compute vector $\boldsymbol{pos}_{t,u}$ which simply indicates the position of the $M$th-highest bid so that bidders know at which position they find the allocation of units in vector $\boldsymbol{alloc}_{t,u}$.

$$\begin{aligned}\boldsymbol{pos}_{t,u} &= \boldsymbol{stat}^{M}_{t,u}\left(\sum_{i=1}^{n}\sum_{m=1}^{M}\boldsymbol{b}^i_m\right)\\ \boldsymbol{alloc}_{t,u} &= \boldsymbol{stat}^{M}_{t,u}\left(\sum_{i=1}^{n}\sum_{m=1}^{M}\boldsymbol{b}^i_m\right) + \mathsf{L}\sum_{i=1}^{n}\left((M+1)^{i-1}\sum_{m=1}^{M}\boldsymbol{b}^i_m\right)\end{aligned}$$

Due to certain ties, it is possible that bidders qualify for more units than there are units available. This is the case when there is a tie involving the $M$th-highest and the $(M+1)$st-highest bid ($t > 1$ and $t + u > M$). Computing additional vectors that reveal the number of bids each bidder is contributing to the tie allow both bidders and the seller to apply fair (*e.g.*, randomized) methods to select how many units each tied bidder receives.

$$\boldsymbol{surplus}_{t,u} = \boldsymbol{stat}^{M}_{t,u}\left(\sum_{i=1}^{n}\sum_{m=1}^{M}\boldsymbol{b}^i_m\right) + \sum_{i=1}^{n}\left((M+1)^{i-1}\sum_{m=1}^{M}\boldsymbol{b}^i_m\right)$$

By computing the above three vectors, bidders are able to determine $m^i$ for each bidder. In the following sections, we show how bidders can privately compute unit prices given by three common multi-unit auction types.

### 5.1 Uniform-Price Auction

In the uniform-price auction, all bidders pay the same price per unit, given by the $(M+1)$st-highest bid which can be straightforwardly computed using the order statistic subprotocol.[8]

$$\boldsymbol{price}_{t,u} = \boldsymbol{stat}^{M+1}_{t,u}\left(\sum_{i=1}^{n}\sum_{m=1}^{M}\boldsymbol{b}^i_m\right)$$

[7] There are certainly more compact representations, but when assuming that $(M+1)^n$ is less than the size of the underlying finite field, a radix representation has the advantage of being efficiently computable.

[8] In order to hide the $M$th-highest bid, vectors $\boldsymbol{pos}_{t,u}$, $\boldsymbol{alloc}_{t,u}$, and $\boldsymbol{surplus}_{t,u}$ can also be computed based on the $(M+1)$st-highest bid by appropriately shifting down the second addends in $\boldsymbol{alloc}_{t,u}$ and $\boldsymbol{surplus}_{t,u}$.

### 5.2 Discriminatory Auction

In the discriminatory auction bidders pay exactly the sum of amounts they specified in each winning bid. Once, $m^i$ is determined, the price bidder $i$ has to pay can be revealed by computing $price^i$ as defined below (please note that this is not a vector).

$$price^i = \sum_{m=1}^{m^i} \sum_{j=1}^{k} j \cdot b^i_{m,j}$$

It is advisable to compute $price^i$ so that only bidder $i$ and the seller get to know it. Other bidders do not need to be informed about the total price bidder $i$ has to pay.

### 5.3 Generalized Vickrey Auction

The generalized Vickrey auction has the most complex pricing scheme of the auction types we consider. A bidder that receives $m^i$ units pays the sum of the $m^i$ highest losing bids submitted by other bidders, *i.e.*, excluding his own losing bids. Unfortunately, this sophisticated pricing scheme also leads to a higher degree of complexity needed to privately compute Vickrey prices based on our vector representation. The unit prices bidder $i$ has to pay can be determined by invoking the order statistic subprotocol $m^i$ times. In contrast to the discriminatory auction protocol proposed in the previous section, all unit prices have to be computed separately instead of just computing the total price each bidder has to pay. Vector

$$\boldsymbol{price}^i_{m,t,u} = \boldsymbol{stat}^m_{t,u}\left(\sum_{h=1,h\neq i}^{n} \sum_{\ell=m^h+1}^{M} \boldsymbol{b}^h_\ell\right)$$

indicates the price of the $m$th unit bidder $i$ receives $(m = \{1, 2, \ldots, m^i\})$. Obviously, heavy use of the order statistic protocol results in more information to be revealed in the case of ties. As in the discriminatory auction protocol, unit prices should only be revealed to the seller and corresponding bidders.

## 6 Implementation Using El Gamal Encryption

Any homomorphic encryption scheme that besides the, say, additive homomorphic operation allows efficient multiplication of encrypted values with a jointly generated random number can be used to implement the auction schemes described in the previous sections. It turns out that El Gamal encryption [El 85], even though it is multiplicative, is quite suitable because

- agents can easily create distributed keys, and
- encrypted values can be exponentiated with a shared random number in a single round.

As El Gamal cipher is a multiplicative homomorphic encryption scheme, the entire computation as described in the previous sections will be executed in the exponent of a generator. In other words, a random exponentiation implements the random multiplication of the additive notation. As a consequence, the $m$th-highest bid is marked by ones instead of zeros in the order statistic protocol.

### 6.1 El Gamal Encryption

El Gamal cipher [El 85] is a probabilistic and homomorphic public-key cryptosystem. Let $p$ and $q$ be large primes so that $q$ divides $p-1$. $\mathbb{G}_q$ denotes $\mathbb{Z}_p^*$'s unique multiplicative subgroup of order $q$.[9] As argued in Footnote 7, $q$ should be greater than $(M+1)^n$. All computations in the remainder of this paper are modulo $p$ unless otherwise noted. The *private key* is $x \in \mathbb{Z}_q$, the *public key* $y = g^x$ ($g \in \mathbb{G}_q$ is an arbitrary, publicly known element). A message $m \in \mathbb{G}_q$ is *encrypted* by computing the ciphertext tuple $(\alpha, \beta) = (my^r, g^r)$ where $r$ is an arbitrary random number in $\mathbb{Z}_q$, chosen by the encrypter. A message is *decrypted* by computing $\frac{\alpha}{\beta^x} = \frac{my^r}{(g^r)^x} = m$. El Gamal is homomorphic as the component-wise product of two ciphertexts $(\alpha\alpha', \beta\beta') = (mm'y^{r+r'}, g^{r+r'})$ represents an encryption of the plaintexts' product $mm'$. It has been shown that El Gamal is semantically secure, *i.e.*, it is computationally infeasible to distinguish between the encryptions of any two given messages, if the decisional Diffie-Hellman problem is intractable [TY98].

We will now describe how to apply the El Gamal cryptosystem as a fully private multiparty computation scheme.[10] If a value represents an additive share, this is denoted by a "+" in the index, whereas multiplicative shares are denoted by "×". Underlying zero-knowledge proofs will be presented in the next section.

***Distributed key generation:*** Each agent chooses $x_{+i}$ at random and publishes $y_{\times i} = g^{x_{+i}}$ along with a zero-knowledge proof of knowledge of $y_{\times i}$'s discrete logarithm. The public key is $y = \prod_{i=1}^n y_{\times i}$, the private key is $x = \sum_{i=1}^n x_{+i}$. Broadcast round complexity and exponentiation complexity of the key generation are $\mathcal{O}(1)$.

***Distributed decryption:*** Given an encrypted message $(\alpha, \beta)$, each agent publishes $\beta_{\times i} = \beta^{x_{+i}}$ and proves its correctness. The plaintext can be derived by computing $\frac{\alpha}{\prod_{i=1}^n \beta_{\times i}}$. Like key generation, the decryption can be performed in a constant number of rounds.

***Random Exponentiation:*** A given encrypted value $(\alpha, \beta)$ can easily be raised to the power of an unknown random number $E = \sum_{i=1}^n e_{+i}$ whose addends can be freely chosen by the agents if each bidder publishes $(\alpha^{e_{+i}}, \beta^{e_{+i}})$ and proves the equality of logarithms. The product of published ciphertexts yields $(\alpha^E, \beta^E)$ in a single step.

### 6.2 Zero-Knowledge Proofs

In order to obtain security against *malicious* or so-called *active* adversaries, bidders are required to prove the correctness of each protocol step. One of the objectives when designing the protocols presented in Section 5 was to enable *efficient* proofs of correctness for protocol steps. In fact, the proposed protocols can be

[9] We will focus on multiplicative subgroups of finite fields here, although El Gamal can also be based on other groups such as elliptic curve groups.

[10] Please note that this multiparty scheme is limited in the sense that it does not allow the computation of *arbitrary* functions.

proven correct by only using so-called $\Sigma$-protocols which just need three rounds of interaction [Dam02, CDS94]. $\Sigma$-protocols are not known to be zero-knowledge, but they satisfy the weaker property of *honest-verifier* zero-knowledge. This suffices for our purposes as we can use the Fiat-Shamir heuristic [FS87] to make these proofs non-interactive. As a consequence, the obtained proofs are indeed zero-knowledge *in the random oracle model* and only consist of a single round.[11] We will make use of the following three $\Sigma$-protocols:

- Proof of knowledge of a discrete logarithm [Sch91]
- Proof of equality of two discrete logarithms [CP92]
- Proof that an encrypted value is one out of two values [CDS94]

### 6.3 Protocol Implementation

Using El Gamal encryption, the computation schemes described in Section 5 can be executed in the exponent of an arbitrary value in $\mathbb{G}_q\backslash\{1\}$ that is known to all bidders. When enabling non-interactive zero-knowledge proofs by applying the Fiat-Shamir heuristic, protocols only require a low constant number of rounds of broadcasting.[12] The allocation of units can be computed in four rounds as described below (see [Bra03a] for further details). Additional rounds may be required to compute unit prices depending on the auction type.

- Round 1: Distributed generation of El Gamal keys.
- Round 2: Publishing El Gamal encryptions of bids and proving their correctness.
- Round 3: Joint computation of $\boldsymbol{pos}_{t,u}$, $\boldsymbol{alloc}_{t,u}$, and $\boldsymbol{surplus}_{t,u}$ as defined in Section 5. One round of interaction is needed for random exponentiation.
- Round 4: Distributed decryption of $\boldsymbol{pos}_{t,u}$, $\boldsymbol{alloc}_{t,u}$, and $\boldsymbol{surplus}_{t,u}$.

These four rounds suffice to determine the outcome of the uniform-price auction. The discriminatory auction requires one additional round of interaction for computing $price^i$. This cannot be integrated in Round 3 because $m^i$ needs to be known for computing $price^i$. The generalized Vickrey auction requires two additional rounds due to random exponentiations needed for computing $\boldsymbol{price}^i_{m,t,u}$.

In Round 4, bidders send decrypted shares of the outcome to the seller rather than publishing them immediately. After the seller received all shares, he publishes them. This ensures that no bidder can quit the protocol prematurely after learning the outcome, thus leaving other bidders uninformed (see also [Bra03a]). The same procedure is applied in Round 5 or 6, respectively, with the difference that the seller does not need to publish shares. As mentioned in Sections 5.2

[11] The additional assumption of a random oracle is only made for reasons of efficiency. Alternatively, we could employ non-interactive zero-knowledge proofs in the *common random string model* (see [DDO$^+$01] and references therein). However, it has become common practice to use secure hash functions like MD5 or SHA-1 as random oracles in practice.

[12] As explained in Section 2, we do not consider the additional overhead caused by bidders that abort the protocol.

and 5.3, it suffices to send information on unit prices to the corresponding bidder.

## 7 Conclusion

We proposed general cryptographic protocols for three common types of multi-unit auctions based on distributed homomorphic encryption and concrete implementations of these protocols using El Gamal cipher. The security of El Gamal encryption as well as the applied zero-knowledge proofs can be based on the decisional Diffie-Hellman assumption. Under this assumption, privacy can not be breached (unless *all* bidders collude). Our protocols reveal the following information if there is a tie at the $(M+1)$st-highest bid: the number of tied bids $(t)$ and the number of bids greater than the tie $(u)$. The generalized Vickrey auction protocol additionally reveals the price of each unit (rather than just the summed up prices each bidder has to pay) and related tie information. Protocols only fail when the random exponentiation "accidently" yields a one. Due to the exponential size of $\mathbb{G}_q$ the probability of this event is negligible.

In the discriminatory and generalized Vickrey auction protocol, sanctions or fines need to be imposed on bidders that quit prematurely because the allocation and the prices of units are revealed in two consecutive steps. A bidder that learns that he will not receive a single unit might decide to quit the protocol. However, his continuing participation is required to compute the prices of units.

**Table 1.** Protocol Complexity (Computation per Bidder)

| Auction Type | # of Rounds | Exponentiations/Communication |
|---|---|---|
| Uniform-Price | 4 | $\mathcal{O}(nM^2k)$ |
| Discriminatory | 5 | $\mathcal{O}(nM^2k)$ |
| Generalized Vickrey | 6 | $\mathcal{O}(nM^3k)$ |

$n$: bidders, $k$: prices/possible bids, $M$: units to be sold

Table 1 shows the complexity of the proposed protocols (in the random oracle model). Round complexity is very low, but communication and computation complexity is linear in $k$ (rather than logarithmic when using binary representations of bids). On the other hand, an advantage of the unary vector representation is that protocols can easily be turned into iterative auction protocols.

## Acknowledgements

This material is based upon work supported by the Deutsche Forschungsgemeinschaft under grant BR 2312/1-1, by the National Science Foundation under grants IIS-9800994, ITR IIS-0081246, and ITR IIS-0121678, and a Sloan Fellowship.

## References

[AS02] M. Abe and K. Suzuki. M+1-st price auction using homomorphic encryption. In *Proc. of 5th International Conference on Public Key Cryptography (PKC)*, volume 2274 of *LNCS*, pages 115–224. Springer, 2002.

[Bra02] F. Brandt. Secure and private auctions without auctioneers. Technical Report FKI-245-02, Technical University of Munich, 2002. ISSN 0941-6358.

[Bra03a] F. Brandt. Fully private auctions in a constant number of rounds. In R. N. Wright, editor, *Proc. of 7th FC Conference*, volume 2742 of *LNCS*, pages 223–238. Springer, 2003.

[Bra03b] F. Brandt. Social choice and preference protection - Towards fully private mechanism design. In N. Nisan, editor, *Proc. of 4th ACM Conference on Electronic Commerce*, pages 220–221. ACM Press, 2003.

[BS01] O. Baudron and J. Stern. Non-interactive private auctions. In *Proc. of 5th FC Conference*, pages 300–313, 2001.

[BS04a] F. Brandt and T. Sandholm. (Im)possibility of unconditionally privacy-preserving auctions. In C. Sierra and L. Sonenberg, editors, *Proc. of 3rd AAMAS Conference*, pages 810–817. ACM Press, 2004.

[BS04b] F. Brandt and T. Sandholm. On correctness and privacy in distributed mechanisms. In P. Faratin and J. A. Rodriguez-Aguilar, editors, *Selected and revised papers from the 6th AAMAS Workshop on Agent-Mediated Electronic Commerce (AMEC)*, LNAI, pages 1–14, 2004.

[CDS94] R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Proc. of 14th CRYPTO Conference*, volume 893 of *LNCS*, pages 174–187. Springer, 1994.

[Cla71] E. H. Clarke. Multipart pricing of public goods. *Public Choice*, 11:17–33, 1971.

[CP92] D. Chaum and T. P. Pedersen. Wallet databases with observers. In *Proc. of 12th CRYPTO Conference*, volume 740 of *LNCS*, pages 3.1–3.6. Springer, 1992.

[CSS05] P. Cramton, Y. Shoham, and R. Steinberg, editors. *Combinatorial Auctions.* MIT Press, 2005. To appear.

[Dam02] I. Damgård. On $\Sigma$-protocols. Lecture Notes, University of Aarhus, Department for Computer Science, 2002.

[DDO$^+$01] A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai. Robust non-interactive zero knowledge. In *Proc. of 21th CRYPTO Conference*, volume 2139 of *LNCS*, pages 566–598. Springer, 2001.

[Di 00] G. Di Crescenzo. Private selective payment protocols. In *Proc. of 4th FC Conference*, volume 1962 of *LNCS*. Springer, 2000.

[El 85] T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.

[FS87] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Proc. of 12th CRYPTO Conference*, LNCS, pages 186–194. Springer, 1987.

[GL90] S. Goldwasser and L. Levin. Fair computation of general functions in presence of immoral majority. In *Proc. of 10th CRYPTO Conference*, volume 537 of *LNCS*, pages 77–93. Springer, 1990.

[GL02] S. Goldwasser and Y. Lindell. Secure computation without agreement. In *Proc. of 16th International Symposium on Distributed Computing (DISC)*, volume 2508 of *LNCS*, pages 17–32. Springer, 2002.

[Gro73] T. Groves. Incentives in teams. *Econometrica*, 41:617–631, 1973.

[Gro04] J. Groth. Efficient maximal privacy in boardroom voting and anonymous broadcast. In *Proc. of 8th FC Conference*, volume 3110 of *LNCS*, pages 90–104. Springer, 2004.

[JS02] A. Juels and M. Szydlo. A two-server, sealed-bid auction protocol. In M. Blaze, editor, *Proc. of 6th FC Conference*, volume 2357 of *LNCS*. Springer, 2002.

[Kik01] H. Kikuchi. (M+1)st-price auction protocol. In *Proc. of 5th FC Conference*, volume 2339 of *LNCS*, pages 351–363. Springer, 2001.

[Kle99] P. Klemperer. Auction theory: A guide to the literature. *Journal of Economic Surveys*, 13(3):227–286, 1999.

[Kri02] V. Krishna. *Auction Theory.* Academic Press, 2002.

[KY02] A. Kiayias and M. Yung. Self-tallying elections and perfect ballot secrecy. In *Proc. of 5th PKC Conference*, number 2274 in LNCS, pages 141–158. Springer, 2002.

[KY03] A. Kiayias and M. Yung. Non-interactive zero-sharing with applications to private distributed decision making. In *Proc. of 7th FC Conference*, volume 2742 of *LNCS*, pages 303–320. Springer, 2003.

[LAN02] H. Lipmaa, N. Asokan, and V. Niemi. Secure Vickrey auctions without threshold trust. In M. Blaze, editor, *Proc. of 6th FC Conference*, volume 2357 of *LNCS*. Springer, 2002.

[NPS99] M. Naor, B. Pinkas, and R. Sumner. Privacy preserving auctions and mechanism design. In *Proc. of 1st ACM Conference on E-Commerce*, pages 129–139. ACM Press, 1999.

[Sch91] C. P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[SS01] T. Sandholm and S. Suri. Market clearability. In *Proc. of 17th IJCAI*, pages 1145–1151, 2001.

[SY02] K. Suzuki and M. Yokoo. Secure combinatorial auctions by dynamic programming with polynomial secret sharing. In *Proc. of 6th FC Conference*, volume 2357 of *LNCS*. Springer, 2002.

[SY03] K. Suzuki and M. Yokoo. Secure generalized Vickrey auction using homomorphic encryption. In *Proc. of 7th FC Conference*, volume 2742 of *LNCS*, pages 239–249. Springer, 2003.

[Ten00] M. Tennenholtz. Some tractable combinatorial auctions. In *Proc. of 17th AAAI Conference*, pages 98–103. AAAI Press / The MIT Press, 2000.

[TY98] Y. Tsiounis and M. Yung. On the security of ElGamal-based encryption. In *Proc. of 1st International Workshop on Practice and Theory in Public Key Cryptography (PKC)*, volume 1431 of *LNCS*, pages 117–134. Springer, 1998.

[Vic61] W. Vickrey. Counter speculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16(1):8–37, 1961.

[Yao86] A. C. Yao. How to generate and exchange secrets. In *Proc. of 27th FOCS Symposium*, pages 162–167. IEEE Computer Society Press, 1986.

# Event Driven Private Counters

Eu-Jin Goh[1] and Philippe Golle[2]

[1] Stanford University
eujin@cs.stanford.edu
[2] Palo Alto Research Center
pgolle@parc.com

**Abstract.** We define and instantiate a cryptographic scheme called "private counters", which can be used in applications such as preferential voting to express and update preferences (or any secret) privately and non-interactively. A private counter consists of an encrypted value together with rules for updating that value if certain events occur. Updates are private: the rules do not reveal how the value of the counter is updated, nor even whether it is updated for a certain event. Updates are non-interactive: a counter can be updated without communicating with its creator. A private counter also contains an encrypted bit indicating if the current value in the counter is within a pre-specified range.

We also define a privacy model for private counters and prove that our construction satisfies this notion of privacy. As an application of private counters, we present an efficient protocol for preferential voting that hides the order in which voters rank candidates, and thus offers greater privacy guarantees than any other preferential voting scheme.

## 1 Introduction

There are many applications in which it is desirable to keep one's personal preferences private. For example, consider the Australian national election, which uses preferential voting. Preferential or instant runoff voting is an election scheme that favors the "most preferred" or "least disliked" candidate. In preferential voting, voters rank their candidates in order of preference. Vote tallying takes place in rounds. In each round, the number of first place votes are counted for all remaining candidates and if no candidate has obtained a majority of first place votes, the candidate with the lowest number of first place votes is eliminated. Ballots ranking the eliminated candidate in first place are given to the second place candidate in those ballots. Every voters' preferences must be made publicly available in an anonymous manner for universal verifiability; that is, anyone can verify that the tallying is done correctly. Unfortunately, revealing all the preferences allows a voter to easily "prove" to a vote buyer that she has given her vote to a specific candidate by submitting a pre-arranged unique permutation out of the $(n-1)!$ ($n$ is the number of candidates) possible candidate preference permutations. Note that $n$ need not be large — $n = 11$ already gives over three and a half million such permutations.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 313–327, 2005.
© Springer-Verlag Berlin Heidelberg 2005

Ideally, we would like to keep the preferences of voters private to the extent that the correct outcome of the election can still be computed.

**Our Contribution.** In this paper, we define a cryptographic scheme called "private counters", which can be used in any application where participants wish to hide but yet need to update their preferences (or any secret) non-interactively. We then present an efficient instantiation of a private counter using semantically secure encryption schemes [17]. We also define a privacy model for a private counter, and use this model to relate the security of our private counter instantiation to the semantic security of the encryption schemes.

Using private counters, we develop a protocol to run preferential elections. In our election scheme, the preferences of voters are kept private throughout vote tabulation and are never revealed, which even a standard non-cryptographic preferential voting scheme cannot achieve. Our solution can also be used in a real world preferential election with physical voting stations for creating each voter's ballot to ensure privacy during vote tabulation. In addition, our election scheme provides voter privacy, robustness, and universal verifiability.

Our preferential election scheme has computational cost $O(nt^4)$ for $n$ voters and $t$ candidates, whereas the current best cryptographic solution for preferential voting without using mixnets has exponential cost $O(n(t! \log(n))^2)$ and it also reveals all the (unlinked) voter preferences for vote tabulation. Note that a mix network solution also reveals all the voter preferences.

We note that our election scheme requires voters to submit a zero-knowledge proof that the private counters that express their vote are well formed. For efficiency, we require that these proofs be non-interactive, which typically involves applying the Fiat-Shamir heuristic [14]. Hence, security of the voting scheme is shown only in the random oracle model.

**Simple Solutions That Do Not Work.** We further motivate our construction of private counters by discussing briefly two natural but unworkable approaches to preferential elections. Consider a preferential election with $t$ candidates. Let $E$ denote a semantically secure encryption scheme with an additive homomorphism.

**Binary Counter.** In "yes/no" elections based on homomorphic encryption, a voter's ballot typically consists of a ciphertext $C_i$ for each candidate $i$, where $C_i = E(1)$ for the candidate $i$ for whom a vote is cast, and $C_j = E(0)$ for all other candidates $j \neq i$. These ciphertexts can be viewed as binary counters associated with the candidates. The encrypted votes can easily be tallied because $E$ has an additive homomorphism.

This approach fails in a preferential election because ballots cannot be efficiently updated after one or more candidates are eliminated. Roughly speaking, the difficulty is that binary counters are essentially stateless (they encode only one bit of information), whereas updating counters requires keeping track of more state. For example, a ballot needs to encode enough information so that the ballot is transferred to the third most preferred candidate if the first and second most preferred candidate have been eliminated. The space cost required to update these binary counters non-interactively is exponential in $t$: a voter must give update instructions for all $2^t$ possible subsets of eliminated candidates. The space cost can be decreased if interaction with the voters is allowed during vote tallying, which is undesirable for any reasonably sized election.

**Stateful Counters.** Another approach is to associate with each candidate $i$ an encryption $C_i = E(r_i)$ of her current rank $r_i$ among the other candidates. This approach also requires a $O(t^2)$ matrix containing encryptions of 0 and 1; row $i$ is used to update the counters when candidate $i$ is eliminated. These ciphertexts allow efficient updates: when a candidate is eliminated, we decrease by one the rank of all the candidates ranked behind the eliminated candidate (that is, we multiply the corresponding rank ciphertexts by $E(-1)$) and leave unchanged the ranks of other candidates (that is, we multiply them by $E(0)$, which is indistinguishable from $E(-1)$). On the other hand, it does not seem possible to tally such stateful counters efficiently without also revealing the preferences of individual voters; recall that in preferential elections, a vote goes to a candidate if and only if that candidate is ranked first – other candidates do not receive "partial" credit based on their ranking.

**Related Work.** We first note that our notion of a private counter is different from that of a cryptographic counter defined by Katz et al. [20]. Among other differences, a cryptographic counter can be updated by anyone whereas a private counter can only be updated by an authority or group of authorities holding a secret key. In addition, a private counter has a test function that outputs an encrypted bit denoting whether the counter's current value belongs to a range of values; this test function is crucial for our applications.

Cryptographic "yes/no" election schemes were proposed by Benaloh [8, 6, 3] and such elections have since received much research interest [5, 25, 10, 11, 18, 15, 12, 2, 19]. It is easy to see that any mix network scheme [7, 25, 18, 19, 16] immediately gives a solution to a preferential election: the mix network mixes encrypted ballots containing preferences and all ballots are decrypted at the end of the mixing; the normal tallying for preferential voting takes place with the decrypted preferences. The disadvantage of a mix network solution is that the preferences for all voters are revealed after mixing. Although the preferences cannot be linked to individual voters, revealing all the preferences allows a voter to "prove" to a vote buyer that she has given her vote to a specific candidate by submitting a unique permutation.

The only cryptographic solution not using mix networks to preferential voting that we are aware of is by Aditya et al. [1]. Let $n$ be the number of voters and $t$ be the number of candidates. They propose a solution using Paillier encryption [22] that has communication cost $t! \log(n)$ bits per vote and a computation cost of $O(n(t! \log(n))^2)$ to decide the outcome of the election. This exponential inefficiency resulted in Aditya et al. recommending the use of mix networks for a preferential election. Furthermore, their solution is no better (in terms of privacy) than a mixnet solution in that it also reveals all the permutations at the end of the election. In this paper, we show that it is possible to have an efficient solution to preferential voting, and yet provide more privacy than a standard non-cryptographic solution (or a mixnet solution).

**Notation.** For the rest of the paper, we denote that $x$ is a vector by writing $\overrightarrow{x}$. For a vector $\overrightarrow{x}$, $\overrightarrow{x}_i$ denotes the $i$th element in the vector. Similarly, for a matrix $Y$, $Y_{i,j}$ refers to the element in the $i$th row and $j$th column. If we have $k$ multiple instances of an object $Z$, then we differentiate these instances with superscripts $Z^1, \ldots, Z^k$. We denote the cartesian product of $k$ integer rings of order $M$ (modulo $M$) $\mathbb{Z}_M \times \cdots \times \mathbb{Z}_M$ as $\mathbb{Z}_M^k$. If $E$ is an encryption scheme, $E(X)$ denotes an encryption of $X$ using $E$, and $E^{-1}(Y)$ denotes the decryption of ciphertext $Y$. Finally, we say that a function $f : \mathbb{Z} \to \mathbb{R}$ is *negligible* if for any positive $\alpha \in \mathbb{Z}$ we have $|f(x)| < 1/x^\alpha$ for sufficiently large $x$.

## 2 Private Counters with Encrypted Range Test

The following parameters define a private counter:

- An integer $M > 1$. We let $\mathbb{Z}_M = \{0, \ldots, M-1\}$ represent the integers modulo $M$. We call $\mathbb{Z}_M$ the domain of the private counter.
- A range $R \subseteq \mathbb{Z}_M$, and an initial value $v_0 \in \mathbb{Z}_M$.
- A set of events $S_1, \ldots, S_k$ and corresponding update values $u_1, \ldots, u_k \in \mathbb{Z}_M$.
- Two (possibly the same) semantically secure encryption schemes $E$ and $F$, with corresponding public/private key pairs $(\mathcal{PK}_E, \mathcal{SK}_E)$ and $(\mathcal{PK}_F, \mathcal{SK}_F)$. Note that the choice of a security parameter for the counter is implicit in the choice of $E$ and $F$.

These parameters define a private counter comprised of a state $C$, and three functions $\mathsf{Eval}, \mathsf{Test}$ and $\mathsf{Apply}$ where:

- $C$ is the state of the private counter;
- the function $\mathsf{Eval}(C, \mathcal{SK}_F) = v \in \mathbb{Z}_M$ returns the current value of the counter;
- the function $\mathsf{Test}(C)$ returns $E(1)$ if $\mathsf{Eval}(C, \mathcal{SK}_F) \in R$ and $E(0)$ otherwise;
- the function $\mathsf{Apply}(C, S_i, \mathcal{SK}_F) = C'$ outputs the new counter state $C'$ after event $S_i$;

and the following properties hold:

- if we denote $C_0$ the initial state of the counter, we have $\mathsf{Eval}(C_0, \mathcal{SK}_F) = v_0$;
- if $C' = \mathsf{Apply}(C, S_i, \mathcal{SK}_F)$, then $\mathsf{Eval}(C', \mathcal{SK}_F) = \mathsf{Eval}(C, \mathcal{SK}_F) + u_i \mod M$;
- the function $\mathsf{Apply}$ can be called at most once per event $S_i$ (this restriction is perfectly natural for the applications we consider).

The function $\mathsf{Eval}$ plays no operational role in a private counter. It is introduced here only to define $\mathsf{Test}$ and $\mathsf{Apply}$ (later we also use $\mathsf{Eval}$ in proofs), but is never invoked directly. For that reason, we define a private counter as a triplet $(C, \mathsf{Test}, \mathsf{Apply})$, leaving out $\mathsf{Eval}$.

**Extension.** We can define more general counters that can handle tests of subset membership instead of just ranges. Since our applications do not require such general counters, we will not consider them further.

### 2.1 Privacy Model

Informally, a private counter should reveal no information about either its initial value or the update values associated with events. Note that these two properties imply that the subsequent value of the counter after one or several invocations of Apply remains private. We formally define privacy with the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

**Privacy Game 0**

**Setup:** $\mathcal{C}$ generates public/private key pairs $(\mathcal{PK}_E, \mathcal{SK}_E)$ and $(\mathcal{PK}_F, \mathcal{SK}_F)$ for encryption schemes $E$ and $F$, and also chooses a domain $\mathbb{Z}_M$ and a set of events $S_1, \ldots, S_k$. $\mathcal{C}$ gives $\mathcal{A}$ the public keys $\mathcal{PK}_E, \mathcal{PK}_F$, together with the domain and set of events. $\mathcal{A}$ outputs the range $R \subseteq \mathbb{Z}_M$, together with two initial values $v^*, v' \in \mathbb{Z}_M$ and two sets of corresponding update values $\overrightarrow{u}^*, \overrightarrow{u}' \in \mathbb{Z}_M^k$ for the $k$ events.

**Challenge:** $\mathcal{C}$ flips a random bit $b$. $\mathcal{C}$ constructs the challenge private counter $(C_b, \mathsf{Test}, \mathsf{Apply})$ from $\mathcal{A}$'s parameters in the following way — If $b = 0$, $\mathcal{C}$ constructs a private counter $C_0$ using initial value $v^*$ and update values $\overrightarrow{u}^*$; if $b = 1$, $\mathcal{C}$ constructs private counter $C_1$ with initial value $v'$ and update values $\overrightarrow{u}'$.

**Queries:** $\mathcal{A}$ can request invocations to the function Apply from $\mathcal{C}$.

**Output:** $\mathcal{A}$ outputs its guess $g$ for the bit $b$.

We say that $\mathcal{A}$ wins privacy game 0 if $\mathcal{A}$ guesses bit $b$ correctly.

**Definition 1.** *A counter scheme is private according to game 0 if all polynomial (in security parameter $t$) time algorithms win game 0 only with negligible advantage* $\mathbf{Adv}(t) = |\Pr[g = b] - 1/2|$.

We use sets of counters in our applications so we extend privacy game 0 to multiple counters and denote the extended game as privacy game 1. Extending game 0 is straightforward and we give a precise definition in Appendix A. Privacy game 1 allows us to prove that we can use a set of private counters simultaneously while preserving privacy of individual counters; the proposition and proof is also found in Appendix A.

**Note.** The privacy requirements for our applications may appear different than the definition given by privacy game 0. For example, an adversary in an application may perform actions that are not described in privacy game 0 such as requesting for decryptions of the output of Test. In later sections describing each application, we will define precisely their privacy requirements and then show that the privacy definition given by game 0 is sufficient.

## 2.2 Construction

We present a private counter construction with domain $\mathbb{Z}_M$, subset $R \subseteq \mathbb{Z}_M$, initial value $v_0 \in \mathbb{Z}_M$ and a set of $k$ events $S_1, \ldots, S_k$ with corresponding update values $u_1, \ldots, u_k \in \mathbb{Z}_M$. Furthermore, we restrict the domain $\mathbb{Z}_M$ to be at most polynomial in size. Let $E$ denote any semantically secure encryption scheme such as ElGamal [13] or Pailler [22], and let $F$ be a semantically secure encryption scheme with an additive homomorphism modulo $M$ such as Naccache-Stern [21] or Benaloh [4].

**Counter State.** The counter state consists of three parts: a $(k+1)$-by-$M$ matrix of ciphertexts called $Q$, a pointer $p$ that points to an element of the matrix $Q$, and two vectors of ciphertexts called $\overrightarrow{u}$ and $\overrightarrow{a}$. The matrices $Q$ and two vectors $\overrightarrow{a}, \overrightarrow{u}$ are fixed and the function Apply only affects the value of the pointer $p$. The matrix $Q$, pointer $p$, and vectors $\overrightarrow{a}, \overrightarrow{u}$ are defined as follows:

**Matrix $Q$.** We first define a vector $\overrightarrow{w} = (w_0, \ldots, w_{M-1})$: let $w_j = E(1)$ if $j \in R$ and $w_j = E(0)$ if $j \notin R$. We now define the $(k+1)$-by-$M$ matrix $Q$ using $\overrightarrow{w}$.
Let $Q^0, \ldots, Q^k$ denote the rows of $Q$. Let $a_0, \ldots, a_k$ be $k+1$ random values chosen uniformly independently at random from $\mathbb{Z}_M$. For $i = 0, \ldots, k$, we define the row $Q^i$ as the image of the vector $\overrightarrow{w}$ cyclically shifted $a_i$ times to the right. That is, if we let $Q_{i,j}$ denote the element of $Q$ in row $Q^i \in \{0, \ldots, k\}$, column $j \in \{0, \ldots, M-1\}$, we have $Q_{i,j} = w_{j-a_i}$, where the subscript $j - a_i$ is computed modulo $M$.

**Pointer $p$.** The pointer is a pair of integers $p = (i, j)$, where $i \in [0, k]$ and $j \in [0, M-1]$, that refer to ciphertext $Q_{i,j}$ in matrix $Q$. The initial state of the counter is defined as $p = (0, a_0 + v_0)$.

**Vectors $\overrightarrow{a}, \overrightarrow{u}$.** Vector $\overrightarrow{a}$ contains $k+1$ ciphertexts $F(a_0), \ldots, F(a_k)$, which are the encryptions of the $k+1$ random values $a_0, \ldots, a_k$ chosen for matrix $Q$. Vector $\overrightarrow{u}$ contains $k$ ciphertexts $F(u_1), \ldots, F(u_k)$, which are the encryptions of the $k$ update values $u_1, \ldots, u_k$.

Only the public key for $E$ is needed to construct $Q$, and only the public key for $F$ is required to build $\overrightarrow{a}$ and $\overrightarrow{u}$.

**Computing Eval.** Recall that the function Eval plays no operational role in a counter. Nevertheless, we describe how to compute Eval to help the reader understand the intuition behind our construction. Let $(i, j)$ be the current value of the pointer $p$. Eval$(C, \mathcal{SK}_F)$ returns the current value of the counter as the integer $(j - a_i) \mod M$.

**Computing Test.** Let $(i, j)$ be the pointer's current value. Test$(C)$ returns the ciphertext $Q_{i,j}$.

**Computing Apply.** We show how to compute the function Apply$(C, S_l, \mathcal{SK}_F)$. Let $p = (i, j)$ be the current value of the pointer. Compute the ciphertext $F(a_l - a_i + u_l)$ by using the additive homomorphism of $F$ on the appropriate ciphertexts from $\overrightarrow{a}$ and $\overrightarrow{u}$. Let $d$ be the decryption of the ciphertext $F(a_l - a_i + u_l)$. Apply$(C, S_l, \mathcal{SK}_F)$ outputs the new pointer $p' = (l, j + d)$ where the value $j + d$ is computed modulo $M$.

**Privacy.** Our counter construction is only private according to the privacy game 0 if the function Apply is never called twice for the same event. We note that our voting application always satisfies this condition. Furthermore, the function Apply takes as input the secret key $\mathcal{SK}_F$, which lets the owner(s) of $\mathcal{SK}_F$ enforce this condition. We give a detailed proof of privacy in Section 2.3.

**Cost.** The size of our counter is dominated by $O(kM)$ ciphertexts from $E$ and $O(k)$ ciphertexts from $F$. The computational cost of building a private counter is dominated by the cost of creating the ciphertexts. Computing the function Apply requires one decryption of $F$.

### 2.3 Proof of Privacy

We now prove that the construction of Section 2.2 is private provided the encryption schemes $E$ and $F$ are semantically secure and the function Apply is never called twice for the same event.

Recall that semantic security for an encryption scheme is defined as a game where the challenger $\mathcal{C}$ first provides the public parameters to the adversary $\mathcal{A}$, upon which $\mathcal{A}$ chooses and sends two equal length messages $M_0, M_1$ back to $\mathcal{C}$. $\mathcal{C}$ then chooses one of the messages $M_b$ and returns the encryption of $M_b$ to $\mathcal{A}$. The goal of the adversary is to guess the bit $b$. In our security proof, we use a variant of the semantic security game where the challenger returns both $E_0 = E(M_b)$ and $E_1 = E(M_{1-b})$ to the adversary. It is easy to see that this variant is equivalent (with a factor of two loss in the security reduction) to the standard semantic security game.

In the privacy game, recall that the adversary outputs two sets of initial values and update values $v^*, u_1^*, \ldots, u_k^*$ and $v', u_1', \ldots, u_k'$ as the choice for its challenge. The main difficulty in the security proof is in embedding the semantic security challenge ciphertexts $E_b, E_{1-b}$ into the private counter's matrix $Q$ so that if $b = 0$, the matrix $Q$ represents initial value $v$, and if $b = 1$, the matrix $Q$ represents initial value $v'$. Similarly, we have to embed the challenge ciphertexts $F_b, F_{1-b}$ into the private counter's vector $\overrightarrow{u}$ so that if $b = 0$, vector $\overrightarrow{u}$ contains $F(u_1^*), \ldots, F(u_k^*)$, and $F(u_1'), \ldots, F(u_k')$ otherwise.

**Proposition 1.** *If the encryption schemes $E$ and $F$ are both semantically secure, the counter of Section 2.2 is private according to privacy game 0.*

*Proof.* We prove the proposition using its contrapositive. Suppose the counter of Section 2.2 is not private. Then there exists an algorithm $\mathcal{A}$ that wins the privacy game with non-negligible advantage; that is, $\mathcal{A}$ non-trivially distinguishes between a private counter with initial value $v^*$ with update values $u_1^*, \ldots, u_k^*$ and a private counter with initial value $v'$ with update values $u_1', \ldots, u_k'$. A standard hybrid argument shows that $\mathcal{A}$ can distinguish between two private counters with non-negligible advantage when the two counters have either —

**Case 1:** different initial values ($v^* \neq v'$) but the same update values ($u_i^* = u_i'$ for $1 \leq i \leq k$).

**Case 2:** the same initial values ($v^* = v'$) but different update values ($u_i^* \neq u_i'$ for at least one $i$ where $1 \leq i \leq k$).

Case 1 implies that $\mathcal{A}$ distinguishes between two private counters based solely on the initial value and case 2 implies that $\mathcal{A}$ distinguishes based solely on the update values. If case 1 holds, then we build an algorithm $\mathcal{B}_1$ that breaks $E$. If case 2 holds, then we build an algorithm $\mathcal{B}_2$ that breaks $F$. Recall that $F$ has an additive homomorphism modulo $k$.

**Algorithm $\mathcal{B}_1$.** We define an algorithm $\mathcal{B}_1$ that uses $\mathcal{A}$ to break the semantic security of $E$ with non-negligible advantage. Algorithm $\mathcal{B}_1$ simulates $\mathcal{A}$ as follows:

**Setup:** Algorithm $\mathcal{B}_1$ is given the encryption scheme $E$ with the public key $\mathcal{PK}_E$ for a security parameter $t$. $\mathcal{B}_1$ generates the key pair $(\mathcal{PK}_F, \mathcal{SK}_F)$ for encryption scheme $F$. $\mathcal{B}_1$ begins by choosing two plaintexts $M_0 = 0$ and $M_1 = 1$ and receives as its challenge two ciphertexts $E_0 = E(M_b)$ and $E_1 = E(M_{1-b})$ for a random bit $b$. The goal of $\mathcal{B}_1$ is to guess the bit $b$.
$\mathcal{B}_1$ runs $\mathcal{A}$ with initial input the public keys $\mathcal{PK}_E, \mathcal{PK}_F$, an arbitrarily chosen domain $\mathbb{Z}_M$ (where $M$ is polynomially large), and a set of events $S_1, \ldots, S_k$. In return, $\mathcal{A}$ outputs the range $R \subseteq \mathbb{Z}_M$, together with two initial values $v^*, v' \in \mathbb{Z}_M$ where $v^* \neq v'$ and two sets of update values $u_1^*, \ldots, u_k^* \in \mathbb{Z}_M$ and $u_1', \ldots, u_k'$ for the events.

**Challenge:** $\mathcal{B}_1$ constructs a private counter starting with matrix $Q$. We define two vectors:

1. $\overrightarrow{w}^* = (w_0^*, \ldots, w_{M-1}^*)$ where $w_j^* = 1$ if $j \in R$, and $w_j^* = 0$ if $j \notin R$, and
2. $\overrightarrow{w}' = (w_0', \ldots, w_{M-1}') = (w_{v'-v^*}^*, w_{v'-v^*+1}^*, \ldots, w_{M-1}^*, w_0^*, \ldots, w_{v^*-v'-1}^*)$ where the subscripts are computed modulo $M$. Note that $\overrightarrow{w}'$ is $\overrightarrow{w}^*$ cyclicly shifted by $v^* - v'$ (a negative value results in a right shift).

We want to construct the vector $\overrightarrow{w} = (w_0, \ldots, w_{M-1})$ with the following property: if $b = 0$, then $\overrightarrow{w}$ is the encryption of $\overrightarrow{w}^*$ and is defined exactly as described in Section 2.2 with domain $\mathbb{Z}_M$ and subset $R \subseteq \mathbb{Z}_M$; if $b = 1$, then $\overrightarrow{w}$ is the encryption of $\overrightarrow{w}'$. To obtain this property, vector $\overrightarrow{w}$ is built from $\overrightarrow{w}^*$ and $\overrightarrow{w}'$ as follows:

- If $w_j^* = 0$ and $w_j' = 0$, we let $w_j = E(0)$.
- If $w_j^* = 0$ and $w_j' = 1$, we let $w_j = E_0$.
- If $w_j^* = 1$ and $w_j' = 1$, we let $w_j = E(1)$.
- If $w_j^* = 1$ and $w_j' = 0$, we let $w_j = E_1$.

The $(k+1)$-by-$M$ matrix $Q$ is constructed exactly as described in Section 2.2 with our vector $\overrightarrow{w}$ and a set of random values $a_0, \ldots, a_k$. The vector $\overrightarrow{a}$ is built as $(F(a_0), \ldots, F(a_k))$ and initial value of pointer $p$ is set to $(0, a_0 + v^*)$. To construct vector $\overrightarrow{u}$, $\mathcal{B}_1$ flips a coin and if 0 uses $u_1^*, \ldots, u_k^*$ to build $\overrightarrow{u}$, and if 1 uses $u_1', \ldots, u_k'$ instead.
$\mathcal{B}_1$ gives the resulting counter $C$ to $\mathcal{A}$. Note that if $b = 0$, $\mathcal{A}$ receives a counter for initial value $v^*$, whereas if $b = 1$, $\mathcal{A}$ receives a counter for initial value $v'$.

**Queries:** $\mathcal{B}_1$ can compute $\mathsf{Apply}$ for $\mathcal{A}$ because $\mathcal{B}_1$ knows the values $a_0, \ldots, a_k$ and also $u_1^*, \ldots, u_k^*$ (respectively $u_1', \ldots, u_k'$).

**Output:** $\mathcal{B}_1$ outputs $g = 0$ if $\mathcal{A}$ guesses that $C$ has initial value $v^*$. Otherwise, $\mathcal{B}_1$ outputs $g = 1$.

With probability 1/2, $\mathcal{B}_1$ chooses the right set of update values for vector $\overrightarrow{u}$ and the counter is well formed. It follows directly that $\mathcal{B}_1$ wins the semantic security game with non-negligible advantage.

**Algorithm $\mathcal{B}_2$.** We define an algorithm $\mathcal{B}_2$ that uses $\mathcal{A}$ to break the semantic security of $F$ with non-negligible advantage. Algorithm $\mathcal{B}_2$ simulates $\mathcal{A}$ as follows:

**Setup:** Algorithm $\mathcal{B}_2$ is given the encryption scheme $F$ with the public key $\mathcal{PK}_F$ for a security parameter $t$. $\mathcal{B}_2$ generates the key pair $(\mathcal{PK}_E, \mathcal{SK}_E)$ for encryption scheme $E$. $\mathcal{B}_2$ begins by choosing two plaintexts $M_0 = 0$ and $M_1 = 1$ and receives as its challenge two ciphertexts $F_0 = F(M_b)$ and $F_1 = F(M_{1-b})$ for a random bit $b$. The goal of $\mathcal{B}_2$ is to guess the bit $b$. The rest of the Setup phase is identical to that for algorithm $\mathcal{B}_1$.

**Challenge:** $\mathcal{B}_2$ constructs a private counter as follows. The $(k+1)$-by-$M$ matrix $Q$ and vector $\overrightarrow{a}$ is constructed exactly as described in Section 2.2. To construct pointer $p$, $\mathcal{B}_2$ flips a coin and if 0 uses initial value $v^*$ to build $p$, and if 1 uses initial value $v'$ instead. The vector of encrypted update values $\overrightarrow{u} = (F(u_1), \ldots, F(u_k))$ is created from $u_1^*, \ldots, u_k^*$ and $u_1', \ldots, u_k'$ as follows: for all $1 \leq i \leq k$,

1. if $u_i^* = u_i'$, then $F(u_i) = F(u_i^*) = F(u_i')$.
2. if $u_i^* < u_i'$, then $F(u_i) = F(u_i^*) \cdot F_0^{u'_i - u_i^*} = F(u_i^* + b(u_i' - u_i^*))$.
3. if $u_i^* > u_i'$, then $F(u_i) = F(u_i') \cdot F_i^{u_i^* - u'_i} = F(u_i' + (i - b)(u_i^* - u_i'))$.

If $b = 0$, then $\overrightarrow{u}$ is the update vector created using $u_1^*, \ldots, u_k^*$. If $b = 1$, then $\overrightarrow{u}$ is update vector created using $u_1', \ldots, u_k'$. Note that the update vector $\overrightarrow{u}$ is computable because $F$ has an additive homomorphism modulo $k$, and also because $u_i^*, u_i' \in \mathbb{Z}_M$ and $\mathbb{Z}_M$ is polynomial in size. Finally, $\mathcal{B}_2$ gives the resulting counter to $\mathcal{A}$.

**Queries:** Before any Apply queries are answered, $\mathcal{B}_2$ flips a coin and if 0 uses $u_1^*, \ldots, u_k^*$ to answer Apply queries, otherwise $\mathcal{B}_2$ uses $u_1', \ldots, u_k'$ instead. With this guess, $\mathcal{B}_2$ can answer Apply queries because $\mathcal{B}_2$ generates (and knows) $a_0, \ldots, a_k$.

**Output:** Algorithm $\mathcal{B}_2$ outputs $g = 0$ if $\mathcal{A}$ guesses that the counter contains the update values $u_1^*, \ldots, u_k^*$. Otherwise, $\mathcal{B}_2$ outputs $g = 1$.

With probability 1/2, $\mathcal{B}_2$ uses the correct set of update values to answer Apply queries, in which case, $\mathcal{B}_2$ wins the semantic security game for $F$ with non-negligible probability. □

## 3 Preferential Voting

In this section, we give a cryptographic solution to preferential voting using our private counter construction. The participants of the election are:

1. $n$ voters labelled $b_1, \ldots, b_n$.
2. $t$ candidates standing for election labelled $x_1, \ldots, x_t$.
3. a number of election authorities that collect the votes, verify them, and collectively compute the result of the election. These election authorities share a single public key but the corresponding private key is shared among all of them. Encryptions are performed with the public key of the election authority but (threshold) decryption requires the consent of a quorum.

In voting, a voter's preferences must remain anonymous and her current first place candidate must never be revealed during vote tabulation. Despite this restriction, the election authorities must 1) tally up votes for each candidate, and 2) verify that ballots are valid by ensuring that a ballot has exactly one first place candidate and that preferences do not change arbitrarily from round to round.

**Setup.** The election authorities jointly generate the public/private key pair using a threshold protocol and publish the public parameters. For preferential voting, we require that $E$ is the Paillier encryption scheme [22], which has an additive homomorphism. In addition, $E$ should be a threshold version of the Paillier encryption scheme [15, 12]. The encryption scheme $F$ can be any scheme with an additive homomorphism modulo $t$ such as Naccache-Stern [21] and Benaloh [4].

**Vote Creation.** Each voter ranks the candidates in decreasing order of preference. For example, if a voter ranks 3 candidates in the order $(x_2, x_3, x_1)$ where candidate $x_2$ is the first choice and candidate $x_1$ the last, we say that candidate $x_2$ has rank 0, candidate $x_3$ has rank 1, and candidate $x_1$ has rank 2. Note that the rank of candidate $x_i$ is equal to the number of candidate ranked ahead of $x_i$.

Before describing how votes are created, we explore how eliminating a candidate affects the current preferences of voter $b_i$. Suppose that candidate $x_1$ has been eliminated. If $b_i$ ranked $x_1$ ahead of $x_2$, the rank of $x_2$ should now be decreased by 1, moving it closer to first place. If $x_1$ was ranked behind $x_2$, then the rank of $x_2$ is unaffected by the removal of $x_1$. Note that this statement holds true regardless of the number of candidates (up to $t-2$) that have been eliminated so far. Therefore, the change in rank of $x_2$ when $x_1$ is eliminated depends only on whether $x_2$ ranked ahead or behind $x_1$ in the initial ranking of $b_i$.

Voter $b_i$ creates her vote as follows. The vote consists of $t$ private counters $P^{b_i,x_1}, \ldots, P^{b_i,x_t}$ (one counter for each candidate $x_j$ where $j \in [1,t]$), together with zero knowledge proofs that these counters are well-formed (see Section 3.1). The domain $D$ of each private counter is $D = [0, t-1)$ (the range of possible ranks) and the range $R$ is $0 \in D$. In private counter $P^{b_i,x_j}$:

1. The initial value $v$ is the initial rank assigned to candidate $x_j$ by voter $b_i$.
2. Events $S_1, \ldots, S_t$ are the events that candidate $x_l$ for $l \in [1,t]$ is eliminated.
3. The update value $u_k$ associated with event $S_k$ for $k \in [1,t]$ is $u_k = 0$ if voter $b_i$ ranks $x_k$ with a higher rank than $x_j$, and $u_k = -1$ if $x_k$ has a lower rank than $x_j$.

Note that the the number of update values that are -1 is equal to the initial rank of candidate $x_j$, since the rank denotes the number of candidates preferred to $x_j$. Thus when a counter reaches 0 (first place choice), it can go no lower. **Vote**

**Checking.** The election authority checks that all the votes are well-formed (see Section 3.1), and discards invalid votes.

**Vote Tallying.** Recall that Test returns $E(1)$ if candidate $x_j$ is the first place candidate and $E(0)$ otherwise. During each round of vote tallying, the election authorities compute the encrypted tally of first place votes for each candidate $x_j$ as $\prod_{i=1}^{n} \mathsf{Test}(P_{b_i,x_j})$. The tally for each candidate is decrypted, requiring the consent of a quorum of private key holders. Note that since Test can be computed publicly, an honest election authority participates in the threshold decryption of the tally only if it is correctly computed.

**Vote Update.** If no candidate wins a majority of votes, the candidate with the fewest number of votes (say, candidate $x_k$) is eliminated. Voters who ranked the eliminated candidate $x_k$ in first place now have their vote transferred to their second most preferred candidate in subsequent rounds. To do so, the election authorities update every voter's private counters to reflect their new first place preferences. The election authorities:

1. remove the $k$-th private counter from all votes; that is, remove counters $P^{b_i,x_k}$ for $i \in [1,n]$.
2. invoke Apply on every vote's $t-1$ remaining private counters with the event that candidate $x_k$ is removed. That is, for voter $b_i$ where $i \in [1,n], i \neq k$, the election authorities invoke $\mathsf{Apply}(P^{b_i,x_j}, S_k, \mathcal{SK}_F)$ for $j \in [1,t]$. Note that no single election authority possesses $\mathcal{SK}_F$ and a quorum must be obtained for the necessary threshold decryptions for the Apply function.

The vote tallying, updating, and verifying process continues with successively less candidates until a candidate wins a majority of first place votes.

**Cost.** Each vote contains $t$ private counters and so the space cost is $O(t^3)$ ciphertexts; as we will see in the next section, the space cost of the proofs is $O(t^4)$ ciphertexts. The computation required to create a vote is dominated by the cost of $O(t^4)$ encryptions for the proofs. Verifying a single vote costs $O(t^4)$. Tallying the first place votes for a single candidate requires one decryption. Updating a vote after each candidate is eliminated requires $t$ decryptions. In summary, the election authority performs $O(nt^4)$ decryptions to compute the outcome of the election.

**Security.** During vote tabulation, the outputs of the function Test on the private counters in each ballot for all voters are tallied and decrypted. The adversary thus learns the decryption of the function Test "in aggregate". Informally, as long as $\mathcal{A}$ controls no more than a small fraction of the total number of voters, the aggregate tally reveals little about the individual vote of any voter. We note that every voting scheme that tallies votes using a homomorphic encryption scheme à la Cramer et al. [10, 11] has the same weakness.

Assuming that decryption of the aggregate counters is safe, the privacy of each voter's ballot follows directly from the privacy guaranteed by Proposition 1. That is, a voter's preferences are never revealed throughout vote submission and tabulation (even to the election authority). In preferential voting, a voter can submit a unique permutation of preferences (which is revealed for universal verifiability) to "prove" how she voted. Non-cryptographic preferential voting and preferential voting using mix networks cannot prevent such privacy leaks but our scheme can because each voter's preferences are never revealed. Furthermore, the election is universally verifiable and anyone can verify that the submitted votes are valid and that the tallies every round are correctly computed. Lastly, the quorum of election authorities ensures that the voting scheme is robust, provided no more than a fraction of them are malicious.

### 3.1 Proving That a Vote Is Valid

In many electronic election schemes, the voter must attach with her ballot, a proof (typically zero-knowledge or witness indistinguishable) that the ballot is correctly formed. For example, in a yes/no election, the voter must prove that the ballot really is an encryption of 0 or 1. Otherwise, the tally may be corrupted by a voter sending an encryption of an arbitrary value.

Efficient interactive zero knowledge proofs of bit encryption can be constructed for well known homomorphic encryption schemes such as Paillier [22, 12, 2] and Benaloh [4, 11]. These proofs are made non-interactive by applying the Fiat-Shamir heuristic, which replaces communication with an access to a random oracle [14]. In practice, the random oracle is replaced by a cryptographic hash function. Security holds in the random oracle model and not in the standard model [23]. Instead of applying the Fiat-Shamir heuristic, we could instead use a trusted source of random bits such as a beacon [24] so as to obtain security in the standard model, but the resulting constructions are less efficient.

In our election scheme, a voter must prove to the election authority in non-interactive zero-knowledge (NIZK) that the $t$ counters expressing her vote are well-formed. Specifically, the voter must prove 1) that the counters only express one first place vote at any given time, and 2) that the transfer of votes as candidates are eliminated proceeds according to a fixed initial ranking of candidates; that is, a vote must always be transferred to the next most preferred candidate among those remaining.

We require NIZK proofs that the decryption $E^{-1}(C)$ (resp. $F^{-1}(C)$) of a ciphertext $C$ lies within a given set of messages $m_0, \ldots, m_t$; we denote such a proof as $\mathsf{NIZKP}\left\{E^{-1}(C) \in \{m_0, \ldots, m_t\}\right\}$ (resp. with $F$). The size and computational cost to create and verify such a proof is linear in $t$ (the size of the set) [12, 2, 11]. These proofs can also be combined conjunctively and disjunctively using standard techniques [9, 26].

Recall that we denote by $t$ the number of candidates. A vote consists of $t$ counters, with matrices $Q^1, \ldots, Q^t$, initial pointers $p^1, \ldots, p^t$, cyclic shift vectors $\overrightarrow{a}^1, \ldots, \overrightarrow{a}^t$ and update vectors $\overrightarrow{u}^1, \ldots, \overrightarrow{u}^t$. To prove that these counters are well-formed, a voter does the following:

1. The voter commits to her initial ranking of candidates. This commitment takes the form of $t$ ciphertexts, $C_1, \dots, C_t$, where $C_i$ is an encryption with $F$ of the initial rank of candidate $x_i$.
2. The voter proves that the commitment given in step 1 is well-formed; that is, the voter proves that the ciphertexts $C_1, \dots, C_t$ are encryptions of the values $0, \dots, t-1$ permuted in a random order. This property is proved by showing that for all $i \in \{0, \dots, t-1\}$, there exists $j$ such that $C_j = F(i)$. Formally, the voter proves for all $i \in \{0, \dots, t-1\}$ that $\bigvee_{j=1,\dots,t} \mathsf{NIZKP}\left\{F^{-1}(C_j) \in \{i\}\right\}$.
3. The voter proves that each matrix $Q^k$ for $k \in \{1, \dots, t\}$ is well-formed:
   - for all $k \in \{1, \dots, t\}$, $i \in \{0, \dots, t\}$, and $j \in \{1, \dots, t\}$, the entry $Q^k_{i,j}$ of matrix $Q^k$ is an encryption of either 0 or 1. Formally, the voter creates $\mathsf{NIZKP}\left\{E^{-1}(Q^k_{i,j}) \in \{0,1\}\right\}$.
   - for all $k \in \{1, \dots, t\}$ and for all $i \in \{0, \dots, t\}$, there is one and only one entry in row $i$ of matrix $Q^k$ that is an encryption of 1. Since the encryption scheme $E$ has an additive homomorphism and we know already that $E^{-1}(Q^k_{i,j}) \in \{0,1\}$, the proof is $\mathsf{NIZKP}\left\{E^{-1}(\prod_{j=1}^{t} Q^k_{i,j}) \in \{1\}\right\}$.
4. The voter proves that the pointers are well-formed; that is, for all $k \in \{1, \dots, t\}$ we have $p^k = F^{-1}(C_k) + F^{-1}(\overrightarrow{a}^k_1)$. Formally, the voter gives $\mathsf{NIZKP}\left\{F^{-1}(C_k \cdot \overrightarrow{a}^k_1) \in \{p^k\}\right\}$.
5. The voter proves that the cyclic shift vectors are well-formed; that is, for all $k \in \{1, \dots, t\}$ and all $i \in \{0, \dots, t\}$, if $\overrightarrow{a}^k_i = F(j)$ then $Q^k_{i,j} = E(1)$. Formally, the proof is $\mathsf{NIZKP}\left\{F^{-1}(\overrightarrow{a}^k_i) \in \{j\}\right\} \bigvee \mathsf{NIZKP}\left\{E^{-1}(Q^k_{i,j}) \in \{0\}\right\}$.
6. The voter proves that the update vectors are well-formed; that is, show that for all $k \in \{1, \dots, t\}$ and all $i \in \{0, \dots, t\}$, we have $\overrightarrow{u}^k_i = F(-1)$ if $F^{-1}(C_i) < F^{-1}(C_k)$ and $\overrightarrow{u}^k_i = F(0)$ otherwise. Formally, the voter gives

$$\Bigg( \bigvee_{\lambda \in \{0,\dots,t-1\}} \Big( \mathsf{NIZKP}\left\{F^{-1}(C_k) \in \{\lambda\}\right\} \bigwedge \mathsf{NIZKP}\left\{F^{-1}(C_i) \in \{0, \dots, \lambda - 1\}\right\}\Big) \bigwedge \mathsf{NIZKP}\left\{F^{-1}(\overrightarrow{u}^k_i) \in \{-1\}\right\}\Bigg) \bigvee \mathsf{NIZKP}\left\{F^{-1}(\overrightarrow{u}^k_i) \in \{0\}\right\}.$$

## Acknowledgments

The authors would like to thank Kobbi Nissim and the anonymous reviewers for their comments.

## References

1. R. Aditya, C. Boyd, E. Dawson, and K. Viswanathan. Secure e-voting for preferential elections. In R. Traunmller, editor, *Proceedings of Electronic Government 2003*, volume 2739 of *LNCS*, pages 246–249, 2003.

2. O. Baudron, P.-A. Fouque, D. Pointcheval, J. Stern, and G. Poupard. Practical multi-candidate election system. In N. Shavit, editor, *Proceedings of the ACM Symposium on the Principles of Distributed Systems 2001*, pages 274–283, 2001.
3. J. Benaloh. *Verifiable Secret-Ballot Elections.* PhD thesis, Yale University, 1987.
4. J. Benaloh. Dense probabilistic encryption. In *Proceedings of the Workshop on Selected Areas in Cryptography 1994*, pages 120–128, May 1994.
5. J. Benaloh and D. Tuinstra. Receipt-free secret-ballot elections. In *Proceedings of the 26th ACM Symposium on Theory of Computing*, pages 544–553, 1994.
6. J. Benaloh and M. Yung. Distributing the power of a government to enhance to privacy of voters. In *Proceedings of the 5th Symposium on Principles of Distributed Computing*, pages 52–62, 1986.
7. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.
8. J. Cohen and M. Fischer. A robust and verifiable cryptographically secure election scheme. In *Proceedings of 26th IEEE Symposium on Foundations of Computer Science*, pages 372–382, 1985.
9. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Y. Desmedt, editor, *Proceedings of Crypto 1994*, volume 893 of *LNCS*, pages 174–187, 1994.
10. R. Cramer, M. Franklin, B. Schoenmakers, and M. Yung. Multi-authority secret-ballot elections with linear work. In U. Maurer, editor, *Proceedings of Eurocrypt 1996*, volume 1070 of *LNCS*, pages 72–83, 1996.
11. R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multi-authority election scheme. *European Transactions on Telecommunications*, 8(5):481–490, Sep 1997.
12. I. Damgård and M. Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In K. Kim, editor, *Proceedings of Public Key Cryptography 2001*, volume 1992 of *LNCS*, pages 119–136, 2001.
13. T. ElGamal. A public key cryptosystem and a signature scheme based on disc rete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, Jul 1985.
14. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A. Odlyzko, editor, *Proceedings of Crypto 1986*, volume 263 of *LNCS*, pages 186–194, 1986.
15. P.-A. Fouque, G. Poupard, and J. Stern. Sharing decryption in the context of voting or lotteries. In Y. Frankel, editor, *Proceedings of Financial Cryptography 2000*, volume 1962 of *LNCS*, pages 90–104, 2000.
16. J. Furukawa, H. Miyauchi, K. Mori, S. Obana, and K. Sako. An implementation of a universally verifiable electronic voting scheme based on shuffling. In *Proceedings of Financial Cryptography 2002*, pages 16–30, 2002.
17. S. Goldwasser and S. Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the 14th ACM Symposium on Theory of computing*, pages 365–377, 1982.
18. M. Hirt and K. Sako. Efficient receipt-free voting based on homomorphic encryption. In B. Preneel, editor, *Proceedings of Eurocrypt 2000*, volume 1807 of *LNCS*, pages 539–556, 2000.
19. M. Jakobsson, A. Juels, and R. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In D. Boneh, editor, *Proceedings of USENIX Security Symposium 2002*, pages 339–353, 2002.
20. J. Katz, S. Myers, and R. Ostrovsky. Cryptographic counters and applications to electronic voting. In B. Pfitzmann, editor, *Proceedings of Eurocrypt 2001*, volume 2045, pages 78–92, 2001.

21. D. Naccache and J. Stern. A new public key cryptosystem based on higher residues. In *Proceedings of the 5th ACM Symposium on Computer and Communications Security*, pages 59–66, 1998.
22. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In J. Stern, editor, *Proceedings of Eurocrypt 1999*, volume 1592 of *LNCS*, pages 223–238, 1999.
23. D. Pointcheval and J. Stern. Security proofs for signature schemes. In U. Maurer, editor, *Proceedings of Eurocrypt 1996*, volume 1070 of *LNCS*, pages 387–398, 1996.
24. M. Rabin. Transaction protection by beacons. *Journal of Computer and System Science*, 27(2):256–267, 1983.
25. K. Sako and J. Kilian. Receipt-free mix-type voting scheme. In L. C. Guillou and J.-J. Quisquater, editors, *Proceedings of Eurocrypt 1995*, volume 921 of *LNCS*, pages 393–403, 1995.
26. A. D. Santis, G. D. Crescenzo, G. Persiano, and M. Yung. On monotone formula closure of SZK. In *Proceedings of the IEEE Symposium on Foundations of Computer Science 1994*, pages 454–465, 1994.

# A Privacy Game 1 (Multiple Counters)

## Privacy Game 1 (for $z$ counters)

**Setup:** same as Privacy Game 0, except that $\mathcal{A}$ outputs two sets of $z$ initial values $\overrightarrow{V}^*, \overrightarrow{V}' \in \mathbb{Z}_M^z$ and their corresponding sets of update values $\overrightarrow{U}^* \in \mathbb{Z}_M^{zk}$ and $\overrightarrow{U}' \in \mathbb{Z}_M^{zk}$ for the events.

**Challenge:** $\mathcal{C}$ flips a random bit $b$. $\mathcal{A}$ constructs the challenge set of private counters $C_b$ from $\mathcal{SK}_E, \mathcal{SK}_F$, and $\mathcal{A}$'s parameters in the following way — If $b = 0$, $\mathcal{C}$ constructs $z$ private counters using initial values in $\overrightarrow{V}^*$ and the update values in $\overrightarrow{U}^*$; if $b = 1$, $\mathcal{C}$ constructs $z$ private counters with initial values in $\overrightarrow{V}'$ and update values in $\overrightarrow{U}'$.

**Queries:** $\mathcal{A}$ can request invocations to the function Apply from $\mathcal{C}$.

**Output:** $\mathcal{A}$ outputs its guess $g$ for the bit $b$.

We say that $\mathcal{A}$ wins privacy game 1 if $\mathcal{A}$ guesses bit $b$ correctly.

**Definition 2.** *A counter scheme is private according to game 1 if all polynomial (in security parameter $t$) time algorithms win game 1 only with negligible advantage $\mathtt{Adv}(t) = |\Pr[g = b] - 1/2|$.*

**Proposition 2.** *If a counter scheme is private according to Game 1, then that same counter scheme is private according to Game 0.*

The proof follows from a standard hybrid argument.

# Secure Distributed *Human* Computation

Craig Gentry[1], Zulfikar Ramzan[1], and Stuart Stubblebine[2]

[1] DoCoMo Communications Laboratories USA, Inc
{cgentry, ramzan}@docomolabs-usa.com
[2] Stubblebine Research Labs
stuart@stubblebine.com

**Abstract.** We suggest a general paradigm of using large-scale distributed computation to solve difficult problems, but where humans can act as agents and provide candidate solutions. We are especially motivated by problem classes that appear to be difficult for computers to solve effectively, but are easier for humans; e.g., image analysis, speech recognition, and natural language processing. This paradigm already seems to be employed in several real-world scenarios, but we are unaware of any formal and unified attempt to study it. Nonetheless, this concept spawns interesting research questions in cryptography, algorithm design, human computer interfaces, and programming language / API design, among other fields. There are also interesting implications for Internet commerce and the B24b model. We describe this general research area at a high level and touch upon some preliminary work; a more extensive treatment can be found in [6].

## 1 Introduction

In Peha's Financial Cryptography 2004 invited talk, he described the Cyphermint PayCash system (see www.cyphermint.com), which allows people without bank accounts or credit cards (a sizeable segment of the U.S. population) to automatically and instantly cash checks, pay bills, or make Internet transactions through publicly-accessible kiosks. Since PayCash offers automated financial transactions and since the system uses (unprotected) kiosks, security is critical; e.g., the kiosk must decide whether a person cashing a check is really the person to whom the check was made out. At first, one might expect that the kiosk uses sophisticated biometric tools, advanced facial recognition algorithms, and the like (which is unsettling since such schemes produce false positives, and can often be outwitted by a clever adversary; e.g., someone can try to hold a photograph up to the camera on the kiosk). However, Cyphermint's solution is very simple: a "human computer" at the back end. The kiosk simply takes a digital picture of the person cashing the check and transmits this picture electronically to a central office, where a human worker compares the kiosk's picture to one that was taken when the person registered with Cyphermint. If both pictures are of the same person, then the human worker authorizes the transaction.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 328–332, 2005.
© Springer-Verlag Berlin Heidelberg 2005

In this example, a human assists in solving problems which are easy for humans but still difficult for even the most powerful computers. Many problems fall into this category; e.g., so called "AI-complete" problems which occur in fields such as image analysis, speech recognition, and natural language processing. Motivated by the above example, we put forth the notion of secure distributed *human* computation (DHC). Although DHC might sound far-fetched, several present-day situations exemplify this paradigm:

– **Spam Prevention:** Recognizing that humans can more easily identify junk mail than computers, some spam prevention mechanisms [11][12][13] leverage human votes. Each email recipient presses a button if it receives what it considers to be spam. If enough people vote that a given email is spam, it is flagged as such, and an appropriate action is taken.
– **CAPTCHA Solutions:** Ironically, spammers can hypothetically use DHC to further their goal [1], [2]. Consider free email providers who have incorporated special puzzles, known as CAPTCHAs, that are easily solved by humans, but challenging for computers, during the account creation phase to prevent spammers from automatically creating email accounts; spammers, in turn, can farm these CAPTCHAs out to humans in exchange for access to illicit content.
– **The ESP Game:** In the ESP Game [3], two players are randomly paired over the Internet; they are not permitted to communicate, but both view the same image on their respective web browsers. Each player types in words that describe the image. As soon as both players enter the same word, they get a new image. The goal is to get through fifteen images in $2\frac{1}{2}$ minutes, and the players' scores increase according to various factors. The players get entertainment value and the game organizers now have labels for their images, which is valuable for improving image search.
– **Distributed Proofreaders:** Distributed proofreaders (www.pgdp.net) is a project that aims to eliminate optical character recognition (OCR) errors in Project Gutenberg (www.gutenberg.net) electronic books. A (small) portion of the image file and corresponding text (generated by OCR) is given side-by-side to a human proofreader who, in-turn, fixes remaining errors. By giving the same piece of text to several proofreaders, errors can be reliably eliminated.
– **Other examples:** Open source software development loosely falls into the DHC paradigm; here the difficult problem is not something crisp like image recognition, but instead that computers have a hard time automatically generating source code. As another example, consider Wikis, which are online encyclopedias that are written by Internet users; the writing is distributed in that essentially almost anyone can contribute to the Wiki.

Applications to E-Commerce and B24b. Web sites typically rely on three revenue sources: advertisements, subscription fees, and e-commerce. Earning sustainable revenues from the first two sources is hard (e.g., click-through rates on advertisements are around 0.7% [5], and outside of specific niche industries, few will pay subscription fees for premium Internet content).

However, DHC yields another revenue source: companies who want specific problems solved can farm them out to the hundreds of millions of Internet users.

In exchange for solving the problem, some service or good is provided. We note that DHC payments have several advantages over credit cards. First, solving a human computation problem might be faster than fetching a credit card and entering the billing details. Second, credit card information can be compromised (e.g., if the merchant web server is compromised). Finally, credit card transaction fees are substantial, so cannot be used for low-value content. In a sense, then, human computation can form a new type of online currency or bartering system.

As an example, such a mechanism might be useful on the New York Times web site (www.nytimes.com) which provides free access to the day's news articles, but charges a fee for archived articles. Such a fee (while necessary from a business perspective) might deter users – especially since they can probably (illegally) obtain the article text; e.g., it was posted to a mailing list. However, instead of charging a fee, the New York Times could give the user a human computation problem (e.g., transcribing an audio feed into text). In exchange for solving the problem, the archived article can be provided. This concept extends to other service offerings; e.g., music downloads or long-distance minutes for solutions. DHC may also enable the Business-to-Four-Billion (B24b) model [10] which aims to provide digital services (wireless communication, Internet, etc.) to the world's four-billion poorest people. Individually these people have annual incomes less than $1500 – yet they have large collective buying power. Although the economic feasibility of B24b is still very much an open question, providing services in exchange for solving DHC problems seems like a useful approach, since it depends on an abundance of human resources, while avoiding cash transactions. (On the other hand, since we are talking about *human* computation, there are ethical issues to consider – in particular, as with any human service, we should ensure that the market for human computation is not unduly exploitative.)

Related Fields. DHC is relevant to several research disciplines. With respect to information security, one can superficially view DHC as a type of secure multi-party computation (for a survey see chapter 7 of [7]), since it may involve multiple human computations, but perhaps the differences are more striking than the similarities. First, the parties are human beings instead of computers; second, the parties are themselves not providing actual inputs, but are instead providing candidate answers (which themselves can be construed as inputs into a group decision-making process); third, the "function" to be computed may not always have a clear-cut answer; fourth, the computation may be facilitated by a semi-trusted[1], but computationally "weak" server (i.e., it cannot solve AI-complete problems itself); fifth, we may not always be restricted by privacy concerns, although they are important in a number of motivating applications.

To analyze security, we may consider the case where the adversaries are rational, and use game-theoretic tools. Also, since DHC is a form of currency, we may use cryptographic tools that have been developed in connection with e-cash.

[1] Server trust can be minimized by augmenting a DHC system with a voter and results-verifiable voting protocol [4].

Finally, we remark that some related work on secure distributed computation and CAPTCHAs ([8], [9], [2], [1]) has appeared in cryptographic literature. We are well aware that "security" is less of a cut-and-dried issue in the human computation context than in the cryptographic context, but we view this as an interesting research challenge. Of course, DHC also has interesting implications for algorithm & programming language design, and human-computer interaction.

Early Thoughts. We have used basic tools from probability theory and decision theory in the design and analysis of secure DHC systems. First, our analysis shows, interestingly, that in the presence of certain types of adversaries, standard tools like Bayesian inference are worse than simple approaches like majority vote for combining individual answers. Next, by trying to model candidate utility functions for end users, we find several design principles: we should provide payouts to clients in direct proportion to a rating that measures the accuracy with which they provide answers; we should decrease the rating substantially if a provided answer seems to be incorrect and increase it only slowly for answers that appear correct; and finally, we should take extra measures if a client's payout from cheating is potentially high. We discuss these issues in greater detail in [6].

While our work is preliminary, it seems that secure *human* computing presents a new paradigm that is likely to suggest a rich set of research problems.

**Acknowledgements.** We thank Naresh Apte, Frank Bossen, Neil Daswani, Minoru Etoh, Ravi Jain, Khosrow Lashkari, Phil Mackenzie, and David Molnar, for helpful discussions and pointing us to various examples of DHC.

## References

[1] L. von Ahn, M. Blum and J. Langford. Telling humans and computers apart automatically. *Communications of the ACM*, 47(2):5660, February 2004.

[2] L. von Ahn, M. Blum, N. Hopper and J. Langford. CAPTCHA: Using hard AI problems for security. *Eurocrypt 2003.*

[3] L. von Ahn and L. Dabbish. Labeling Images with a Computer Game. *ACM CHI 2004.* See also `http://www.espgame.org/`

[4] R. Cramer, R. Gennaro and B. Schoenmakers. A Secure and Optimally Efficient Multi-Authority Election Scheme. *EUROCRYPT 97.*

[5] X. Drèze and F. Hussherr. Internet Advertising: Is Anybody Watching? *Journal of Interactive Marketing,* 2003, Vol. 17 (4), 8-23.

[6] C. Gentry, Z. Ramzan, and S. Stubblebine. Secure Distributed Human Computation. *Proc. ACM Conference on Electronic Commerce, 2005.*

[7] O. Goldreich. Foundations of Cryptography – Volume 2. *Cambridge University Press,* 2004.

[8] P. Golle and I. Mironov. Uncheatable Distributed Computations. *RSA Conference, Cryptographers' Track 2001.*

[9] P. Golle and S. Stubblebine. Distributed computing with payout: task assignment for financial- and strong- security. *Financial Cryptography 2001.*

[10] C. K. Prahalad and S. Hart. The Fortune at the Bottom of the Pyramid. *Strategy + Business*, Issue 26, Q1 2000.

[11] Spam Net Web Site. `http://www.cloudmark.com.`

[12] Vipul's Razor Web Site. `http://sourceforge.net/projects/razor.`

[13] F. Zhou, L. Zhuang, B. Zhao, L. Huang, A. D. Joseph, and J. Kubiatowicz. Approximate Object Location and Spam Filtering. *ACM Middleware, 2003.*

# Secure Multi-attribute Procurement Auction

Koutarou Suzuki[1] and Makoto Yokoo[2]

[1] NTT Information Sharing Platform Laboratories, NTT Corporation,
1-1 Hikari-no-oka, Yokosuka, Kanagawa, 239-0847 Japan
suzuki.koutarou@lab.ntt.co.jp

[2] Faculty of Information Science and Electrical Engineering, Kyushu University,
6-10-1 Hakozaki, Higashi-ku, Fukuoka, 812-8581 Japan
lang.is.kyushu-u.ac.jp/~yokoo/
yokoo@is.kyushu-u.ac.jp

**Abstract.** In this paper, we develop a secure multi-attribute procurement auction, in which a sales item is defined by several attributes called qualities, the buyer is the auctioneer (e.g., a government), and the sellers are the bidders. We first present a Vickrey-type protocol that can be used for multi-attribute procurement auctions. Next, we show how this protocol can be executed securely.

**Keywords:** Procurement auction, Vickrey auction, security, privacy.

## 1 Introduction

Internet auctions have become an integral part of Electronic Commerce and a promising field for applying game-theory and information security technologies. Also, electronic bidding over the public network has become popular for procurement auctions. Since these auction procedures can be efficiently carried out, they have been introduced very rapidly and will be used more widely in the future.

Current research on auctions is focusing mostly on models in which price is the unique strategic dimension, with some notable exceptions [2]. However, in many situations, it is necessary to conduct negotiations on multiple attributes of a deal. For example, in the case of allocating a task, the attributes of a deal may include starting time, ending deadline, accuracy level, etc. A service can be characterized by its quality, supply time, and risk involved, in case the service is not supplied on time. Also, a product can be characterized by several attributes, such as size, weight, and supply date.

In this paper, we develop a secure multi-attribute procurement auction, in which a sales item is defined by several attributes called quality, the buyer is the auctioneer (e.g., a government), and the sellers are the bidders. Our goal is to develop a protocol in which acting honestly is a dominant strategy for sellers and that does not leak the true cost of the winner, which is highly classified information that the winner wants to keep private.

We first present a Vickrey-type protocol that can be used for multi-attribute procurement auctions. In this protocol, acting honestly is a dominant strategy

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 333–335, 2005.
© Springer-Verlag Berlin Heidelberg 2005

for sellers and the resulting allocation is Pareto efficient as shown in Section 2. Next, we show how this protocol can be executed securely, i.e., the protocol does not leak the true cost of the winner, which is highly classified information that the winner wants to keep private in Section 3.

## 2 Proposed Vickrey-Type Protocol

First, we describe the model of a multi-attribute procurement auction. This model is a special case of [4], in which multiple tasks are assigned.

- There exists a single buyer 0, a set of sellers/bidders $N = \{1, 2, \ldots, n\}$, and a task to be assigned to a seller/bidder.
- For the task, quality $q \in Q$ is defined. We assume there is a special quality $q_0 \in Q$, which represents the fact that the task is not performed at all.
- Each bidder $i$ privately observes his type $\theta_i$, which is drawn from set $\Theta$. The cost of bidder $i$ for performing the task when the achieved quality is $q$ is represented as $c(\theta_i, q)$. We assume $c$ is normalized by $c(\theta_i, q_0) = 0$.
- The gross utility of buyer 0 when the obtained quality is $q$ is represented as $V(q)$. We assume $V$ is normalized by $V(q_0) = 0$.
- The payment from the buyer to a winning seller/bidder $i$ is represented as $p_i$. We assume each participant's utility is quasi-linear, i.e., for winning seller $i$, his utility is represented as $p_i - c(\theta_i, q)$. Also, for the buyer, her (net) utility is $V(q) - p_i$.

Please note that although only one parameter $q$ is used to represent the quality of the task, it does not mean our model can handle only one-dimensional quality. We don't assume $q$ is one-dimensional. For example, $q$ can be a vector of multiple attributes.

The proposed Vickrey-type protocol is described as follows.

- Each bidder $i$ submits a pair $(q_i, b_i)$, which means that if he performs a task with quality $q_i$, the resulting social surplus is $b_i$. If the bidder acts honestly, he should choose $q_i = \arg\max_q V(q) - c(\theta_i, q)$ and $b_i = V(q_i) - c(\theta_i, q_i)$.
- The buyer 0 chooses $i^*$ so that $b_i$ is maximized, i.e., $i^* = \arg\max_i b_i$. The buyer 0 allocates the task to bidder $i^*$ with quality $q_{i^*}$.
- The payment $p_{i^*}$ to bidder $i^*$ is defined as: $p_{i^*} = V(q_{i^*}) - b_{2nd}$, where $b_{2nd} = \max_{j \neq i^*} b_j$.

We can consider this protocol to be a special case of the Vickrey-Clarke-Groves-based protocol presented in [4]. However, in the protocol described in [4], a bidder needs to fully expose his private information $\theta_i$. In this protocol, we can avoid the full exposure of types. By this modification, the protocol becomes easier to implement securely.

Please note that if all bidders act honestly, payment $p_{i^*}$ is equal to $V(q^*) - [V(q^*_{\sim i}) - c(\theta_{j^*}, q^*_{\sim i})]$, where $(q^*_{\sim i}, j^*) = \arg\max_{j \neq i^*, q} V(q) - c(\theta'_j, q)$, i.e., $(q^*_{\sim i}, j^*)$ is the second-best choice when the task is not allocated to bidder $i^*$. We can assume that the payment to bidder $i^*$ is equal to the increased amount of the social surplus except for $i^*$ caused by the participation of $i^*$.

For the proposed Vickrey-type protocol, the following theorems hold.

**Theorem 1.** *In the multi-attribute procurement auction protocol, for each bidder i, acting honestly, i.e., reporting* $q_i = \arg\max_q V(q) - c(\theta_i, q)$ *and* $b_i = V(q_i) - c(\theta_i, q_i)$*, is a dominant strategy.*

**Theorem 2.** *The multi-attribute procurement auction protocol is individually rational both for the sellers and the buyer.*

**Theorem 3.** *The multi-attribute procurement auction protocol is Pareto efficient in the dominant strategy equilibrium where each agent acts honestly.*

## 3 Secure Protocol

We propose two cryptographic protocols based on [1] and [3] that realize our procurement auction.

We can securely realize our procurement auction based on the M+1-st price auction in [1] using homomorphic encryption. In the bidding phase, bidder $i$ bids the encryption of his price $b_i$ and encryption $E(q_i)$ of his quality $q_i$. In the opening phase, winning bidder $i^* = \arg\max_i b_i$ and second highest price $b_{2nd} = \max_{j \neq i^*} b_j$ are computed by using the technique of [1]. Next, quality $q_{i^*}$ of the winning bidder $i^*$ is obtained by decrypting $E(q_i)$, and payment $p_{i^*} = V(q_{i^*}) - b_{2nd}$ is computed. The scheme is easy to make robust. However, the scheme is not efficient, i.e., its complexity is $O(np)$ where $n$ and $p$ are the number of bidders and prices, respectively.

We can also securely realize our procurement auction based on the secure auction in [3], where the auctioneer securely computes the circuit of auction using Yao's garbled circuit. We apply the secure auction circuit evaluation of [3] to the circuit of our procurement auction. The scheme is efficient, i.e., its complexity is $O(n \log(p))$. However, the scheme is difficult to make robust.

This suggests that, we can use the first protocol if strong security is needed, and the second protocol if $p$ is large.

## References

1. Masayuki Abe and Koutarou Suzuki. M+1-st price auction using homomorphic encryption. *Proceedings of Public Key Cryptography 2002*, 2002.
2. Yeon-Koo Che. Design cometition through multidimensional auctions. *RAND Journal of Economics*, 24(4):668–680, 1993.
3. Moni Naor, Benny Pinkas, and Reuben Sumner. Privacy preserving auctions and mechanism design. In *Proceedings of the First ACM Conference on Electronic Commerce (EC-99)*, pages 129–139, 1999.
4. Takayuki Suyama and Makoto Yokoo. Strategy/false-name proof protocols for combinatorial multi-attribute procurement auction. In *Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS2004)*, 2004.

# Audit File Reduction Using N-Gram Models*

Fernando Godínez[1], Dieter Hutter[2], and Raúl Monroy[3]

[1] Centre for Intelligent Systems, ITESM–Monterrey,
Eugenio Garza Sada 2501, Monterrey, 64849, Mexico
fgodinez@itesm.mx

[2] DFKI, Saarbrücken University, Stuhlsatzenhausweg 3,
D-66123 Saarbrücken, Germany
hutter@dfki.de

[3] Department of Computer Science, ITESM–Estado de México,
Carr. Lago de Guadalupe, Km. 3.5, Estado de México,
52926, Mexico
raulm@itesm.mx

While some accurate, current Intrusion Detection Systems (IDS's) get rapidly overwhelmed with contemporary information workload [1, 2]. This problem partly dwells in the number of repetitive spurious information that IDS's unnecessarily analyse. Using this observation, we propose a methodology which can be used to significantly remove such spurious information and thus alleviate intrusion detection.

Throughout our experiments we have considered host-based intrusion detection, using the 1998 DARPA repository [3]. The IDS is thus assumed to make an audit from a set of sessions, each of which is a sequence of system calls and corresponds to either of the following services: `telnet`, `ftp`, `smtp`, `finger` or `echo`.

The reduction methodology is twofold:

1 **(Training):** We identify and tag with a new label the sequences of system calls of most frequent occurrence, considering only intrusion-free sessions; and

2 **(Reduction):** We shrink an input session replacing every such a repetitive sequence with its corresponding new label.

Folding repetitive sequences significantly reduces the length of a given session: we obtained an average reduction factor of 4 (3.6, worst case scenario, and 4.8, best case one.) It also helps intrusion detection: for example, it is much faster to build an hidden Markov model-based misuse IDS; and it slightly increases the detection ratio but, more importantly, the false positive ratio is only 1% higher.

*Training.* To identify sequences of system calls of most frequent occurrence, we use n-gram theory [4]. *N-gram theory* comprises a collection of probabilistic

* This research is supported by three research grants CONACyT 33337-A, CONACyT-DLR J200.324/2003 and ITESM CCEM-0302-05.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 336–340, 2005.

© Springer-Verlag Berlin Heidelberg 2005

methods for estimating the probability that a sequence of symbols, i.e. system calls, will occur in a larger, unseen sequence, i.e. a session. Using such probabilities, we have selected a collection of sequences of system calls, henceforth called *n-grams*, that, when folded, are hoped to largely shrink an input session.

The training step consists of 4 steps: i) n-gram extraction; ii) n-gram priority assignment; iii) n-gram selection; and iv) n-gram overlapping avoidance. N-gram extraction consists of identifying all the n-grams arising throughout every session as well as counting their occurrences. Because it consists of one or more interleaved processes, each session is first manipulated so that it is turned into a sequence of orderly processes.

Priority assignment consists of associating with every possible n-gram an estimation as to how much will it reduce a given session, called *its priority.* This step poses two technical problems. First, some n-grams may not have turned up in any training session and yet they all must be assigned a priority. To overcome this problem, we use a *discounting strategy*, namely good-Turing, with which we can estimate an occurrence probability and then use it to compute a priority. Second, some n-grams yield a high reduction ratio regardless of the service, but others impact only on a specific service. For a service not to be neglected only because it has a less representative body in the training sessions, priority should account for both the estimated reduction factor per day, considering sessions of several services, and per service.

The priority of an n-gram, if appears in the training corpora, is given by:

$$Pr_t = \frac{n \times (f_t + 1)}{N_t} \tag{1}$$

where $n$ is the size of the n-gram, $N$ is the total number of system calls within a day considering either all services or a given one, $f$ is the frequency of occurrence of the associated n-gram, and where $t$ stands either for $d$, a day, or $s$, a given service. By contrast, the priority for an n-gram, if an probability of occurrence is estimated, is given by:

$$Pr_t = P_t \times n \tag{2}$$

In the third step, n-grams are first separated into two classes, depending on whether or not they occur in the training corpora, and then each class is manipulated as follows. First, each n-gram is put into 2 separate sets, in one the n-gram is labelled with $Pr_d$ and with $Pr_s$ in the other. Then, each set is arranged according to the n-grams priority in descending order. Then, using the 4 separate arrays, we select as many top n-grams as required in order to achieve a designated reduction factor. If sorting turns out a prohibitively expensive process, as is in the normal case of huge sessions, we suggest to depict histograms and then examine them seeking the n-grams with most promising reduction factor. In this case, it is helpful to consider n-grams whose frequency of occurrence is (nearly) a multiple of the number of different sessions. The rationale behind this selection criterion is that such n-grams are likely to have appeared along every session.

The fourth, final step is n-gram overlapping avoidance. N-grams tend to overlap with each other, they might intersect at some point. To avoid overlapping

```
mmap(2)||success  ○  close(2)||success  ○  open(2)_-_read||success  ○
mmap(2)||success  ○  mmap(2)||success   ○  munmap(2)||success       ○
mmap(2)||success  ○  close(2)||success  ○  open(2)_-_read||success  ○
mmap(2)||success  ○  mmap(2)||success   ○  munmap(2)||success       ○
mmap(2)||success  ○  close(2)||success  ○  open(2)_-_read||success  ○
mmap(2)||success  ○  mmap(2)||success   ○  munmap(2)||success       ○
mmap(2)||success  ○  close(2)||success  ○
```

**Fig. 1.** An example reduction n-gram of size 20 with $Pr_d = 0.1135$. ○ denotes the sequence constructor function

as well as using n-grams with a higher reduction ratio, we form a queue with the selected n-grams, ordering them by priority. Then, we create a window of a size equal to the largest n-gram in the queue. In production, this window is filled with an input session and then tested against the n-grams in the priority queue. By substituting n-grams with higher ratio we guarantee that, even if there is an overlapping, only the n-grams that provide maximum reduction are used. Notice that by substituting an n-gram with a new symbol we are avoiding further substitution on that segment resulting in overlapping elimination. We avoid substitution because the newly added symbol is not present in any n-gram used in the substitution.

We run this training methodology using 5 DARPA log files. We initially selected 200 n-grams from the 4 independent arrays. Our training experiments show that only 11 n-grams are really used out of the 100 n-grams selected from the occurrence-frequency, $Pr_d$ array; only 5 n-grams are used out of the 50 ones extracted from the occurrence-probability $Pr_d$ array; and only 3 n-grams were used from the 50 ones selected from the two $Pr_d$ arrays. Thus 89% of 93% of the selected n-grams are overlapping. Since these n-grams do not overlap, any subsequent reductions do not consider a priority. Our main result is a set of 19 reduction n-grams that, as discussed below, provide an average reduction rate of 74%. One such an n-gram is shown in Fig. 1. The n-gram reduction set is available upon request, by sending e-mail to the 1st author.

*Reduction.* When tested against the training sessions, the n-gram reduction set provided an average reduction of 74%. Then, we validated the n-gram set by making it reduce a collection of unseen sessions, taken from 5 different DARPA log files from the 1999 repository. The results obtained from the validation experiments are shown in table 1; they portray an average reduction of 70.5%. Given that the training log files and the validation ones are from a different year, we conclude that the n-grams are general enough to fold sessions from different users.

*Impact on Intrusion Detection.* Using our folding approach, we have reduced up to 75% the length of a typical session. This reduction allows us to build and use hidden Markov models (HMM's) with larger sequences than those used in current approaches [1, 2, 5]. HMM's take a large amount of time for training.

**Table 1.** Validation Results

| Log File | size(file) | size(reduced file) | Reduction factor | N-grams used |
|---|---|---|---|---|
| 1 | 776, 000 | 270, 000 | 65.3% | 7 |
| 2 | 1, 800, 000 | 486, 000 | 73% | 12 |
| 3 | 1, 150, 000 | 344, 000 | 70.1% | 5 |
| 4 | 801, 000 | 175, 000 | 78.2% | 9 |
| 5 | 1, 158, 000 | 392, 000 | 66.2% | 5 |
| Telnet | 209, 000 | 48, 000 | 77.1% | 5 |

Wagner and Soto, describe the disadvantages of using only short sequences as the detection base using HMM's [6]. We used entire sessions containing the attacks for both, our training data and detection testing. We used a single HMM for each attack.

All the attacks used throughout our experimentations are present in the 1998 and 1999 DARPA repositories. We obtained a detection ratio of 97%. False positive rate is 10.5%. This false positive rate is still high. By reducing the detection threshold, false positive ratio also lowers. Initially we used a 90% similarity measure, i.e., to be labelled as an attack, the tested sequence should be 90% similar to the training sequence. When increased to 95%, false positives were reduced to 6.5%. Detection ratio was also lowered to 92%.

By using reduced and a similarity measure of 90%, detection ratio increased to 98%, and false positive rate was 11.5%. By increasing the similarity measure to 95%, false positives lowered to 7% and the detection ratio also lowered to 94%. We tested the same attacks for reduced and non-reduced sessions. The difference in false positives was found in short attacks such as `eject`. Most of the false positives were normal sessions labelled as one of these short attacks.

From these results we can conclude that folded sessions do not have a negative impact on intrusion detection. Moreover, by using folded sessions the detection rate increases and the false positives only are 1% higher. When using folded sessions, we found a higher detection rate for variations of these same short attacks. The use of reduced sessions is very helpful when detecting attack variations.

## References

1. Warrender, C., Forrest, S., Pearlmutter, B.: Detecting intrusions using system calls: Alternative data models. In: IEEE Symposium on security and Privacy, IEEE Computer Society Press (1999)
2. Yeung, D.Y., Ding, Y.: Host-based intrusion detection using dynamic and static behavioral models. Pattern Recognition **Vol. 36** (2003) pp. 229–243
3. Lippman, R.P., Cunningham, R.K., Fried, D.J., Graf, I., Kendall, K.R., Webster, S.E., Zissman, M.A.: Results of the DARPA 1998 offline intrusion detection evaluation. slides presented at RAID 1999 Conference (1999)

4. Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Massachusets Institute of Technology, Cambridge, Massachusets 02142 (1999)
5. Qiao, Y., Xin, X., Bin, Y., Ge, S.: Anomaly intrusion detection method based on hmm. ELECTRONIC LETTERS **38** (2002) 663–664
6. Wagner, D., Soto, P.: Mimicry attacks on host based intrusion detection systems. In: Ninth ACM Conference on Computer and Communications Security, Washington, DC, USA, ACM (2002) 255–265

# Interactive Diffie-Hellman Assumptions with Applications to Password-Based Authentication

Michel Abdalla and David Pointcheval

Departement d'Informatique,
École normale supérieure,
45 Rue d'Ulm, 75230 Paris Cedex 05, France
{Michel.Abdalla, David.Pointcheval}@ens.fr
http://www.di.ens.fr/users/{mabdalla, pointche}

**Abstract.** Password-based authenticated key exchange are protocols that are designed to provide strong authentication for client-server applications, such as online banking, even when the users' secret keys are considered weak (e.g., a four-digit pin). In this paper, we address this problem in the three-party setting, in which the parties trying to authenticate each other and to establish a session key only share a password with a trusted server and not directly among themselves. This is the same setting used in the popular *Kerberos* network authentication system. More precisely, we introduce a new three-party password-based authenticated key exchange protocol. Our protocol is reasonably efficient and has a per-user computational cost that is comparable to that of the underlying two-party authenticated key exchange protocol. The proof of security is in the random oracle model and is based on new and apparently stronger variants of the decisional Diffie-Hellman problem which are of independent interest.

**Keywords:** Password-based authentication, Diffie-Hellman assumptions, multi-party protocols.

## 1 Introduction

**Motivation.** Key exchange protocols are cryptographic primitives that allow users communicating over an unreliable channel to establish secure sessions keys. They are widely used in practice and can be found in several different flavors. In this paper, we are interested in the setting in which the secret keys shared among the users are not uniformly distributed over a large space, but are rather drawn from a small set of values (e.g., a four-digit pin). This seems to be a more realistic scenario since, in practice, these keys are usually chosen by humans. Moreover, they also seem to be more convenient to use as they do not require the use of more specialized hardware for storing or generating secret keys.

Due to the low entropy of the secret keys, password-based protocols are always subject to password-guessing attacks. In these attacks, also known as dictionary

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 341–356, 2005.
© Springer-Verlag Berlin Heidelberg 2005

attacks, the adversary tries to impersonate a user by simply guessing the value of his password. Since these attacks cannot be completely ruled out, the goal of password-based protocol is to limit the adversary's capability to the online case only. In an online attack, whose success probability is still non-negligible, the adversary needs be present and interact with the system during his attempt to impersonate a user. In other words, the adversary has no means of verifying off-line whether or not a given password guess is correct. The idea of restricting the adversary to the online case only is that we can limit the damage caused by such attacks by using other means, such as limiting the number of failed login attempts or imposing a minimum time interval between failed attempts.

PASSWORD-BASED PROTOCOLS IN THE 3-PARTY MODEL. Due to their practical aspects, password-based key exchange protocols have been the subject of extensive work in the recent years. But despite the attention given to them, it was only recently [1] that the problem has been formally addressed in the three-party model, where the server is considered to be a trusted third party (TTP). This is the same scenario used in the popular 3-party *Kerberos* authentication system. The main advantage of these systems is that users are only required to remember a single password, the one they share with a trusted server, while still being able to establish secure sessions with many users. The main drawback is the need of the trusted server during the establishment of these secure sessions.

In [1], the authors put forth a formal model of security for 3-party password-based authenticated key exchange (PAKE) and present a natural and generic construction of a 3-party password-based authenticated key exchange from any secure 2-party one. There are three phases in their generic construction. In the first phase, a high-entropy session key is generated between the server and each of the two clients using an instance of the 2-party PAKE protocol for each client. In the second phase, a message authentication code (MAC) key is distributed by the server to each client using a 3-party key distribution protocol. In the final phase, both clients execute an authenticated version of the Diffie-Hellman key exchange protocol [13] using the MAC keys obtained in the previous phase.

EFFICIENT 3-PARTY PASSWORD-BASED PROTOCOLS. Though attractive and natural, the construction given in [1] is not particularly efficient. Not only does it require a large amount of computation by the server and the clients, but it also has a large number of rounds. In this paper, we show how to improve both measures when the underlying 2-party password-based key exchange protocol is based on the encrypted key exchange protocol of Bellovin and Merritt [7].

The main idea behind our protocol is quite simple. In order to protect legitimate users from learning each other's password via an off-line dictionary attack, the server randomizes all the values that it receives from one participant before re-encrypting them using the password of another participant. Starting from this idea, we can design a provably-secure protocol, based on the encrypted key exchange of Bellovin and Merritt [7]. The new protocol is quite simple and elegant and, yet, we can prove its security (see Section 4). Moreover, it is also rather efficient, specially when compared to the generic construction in [1]. In particular, the costs for each participant of the new 3-party protocol are comparable to those of a 2-party key exchange protocol. The main drawback of the new 3-party protocol is that it relies on stronger assumptions than those used by the generic construction in addition to being in the random oracle model.

New Diffie-Hellman assumptions. Despite the simplicity of the protocol, its proof of security does not follow directly from the standard Diffie-Hellman assumptions and requires the introduction of some new variants of these standard assumptions. We call them chosen-basis Diffie-Hellman assumptions due to the adversary's capability to choose some of the bases used in the definition of the problem. These assumptions are particularly interesting when considered in the context of password-based protocols and we do expect to find applications for them other than the ones in this paper. Despite being apparently stronger than the standard Diffie-Hellman assumptions, no separations or reductions between these problems are known. Hence, to gain more confidence in these assumptions, we also provide lower bounds for them in the generic group model of Shoup [23].

**Related Work.** Password-based authenticated key exchange has been quite extensively studied in recent years. While the majority of the work deals with different aspects of 2-party key exchange (e.g., [3, 8, 9, 14, 15, 17, 20]), only a few take into account the 3-party scenario (e.g., [1, 10, 16, 19, 24, 25, 26]). Moreover, to the best of our knowledge, with the exception of the generic construction in [1], none of the password-based schemes in the 3-party scenario enjoys provable security. Other protocols, such as the Needham and Schroeder protocol for authenticated key exchange [22] and the symmetric-key-based key distribution scheme of Bellare and Rogaway [5], do consider the 3-party setting, but not in the password-based scenario. As we mentioned above, the goal of the present work is to provide a more efficient and provably-secure alternative to the generic protocol of [1].

**Contributions.** We make two main contributions in this paper.

An efficient construction in Random Oracle model. We present a new construction of a 3-party password-based (implicitly) authenticated key exchange protocol, based on the encrypted key exchange protocols in [6, 21, 9]. The protocol is quite efficient, requiring only 2 exponentiations and a few multiplications from each of the parties involved in the protocol. This amounts to less than half of the computational cost for the server if the latter were to perform two separate key exchange protocols, as in the generic construction of [1]. The gain in efficiency, however, comes at the cost of stronger security assumptions. The security proof is in the Random Oracle model and makes use of new and stronger variations of the Decisional Diffie-Hellman assumption.

New Diffie-Hellman assumptions. The proof of security of our protocol makes use of new non-standard variations of the standard Diffie-Hellman assumptions. These assumptions are of independent interest as they deal with interesting relations between the computational and the decisional versions of the

Diffie-Hellman assumption. We call them chosen-basis decisional Diffie-Hellman assumptions, given the adversary's capability to choose some of the bases used in the definition of the problem. Despite being apparently stronger than the standard Diffie-Hellman assumptions, no separations or reductions between these problems are known. Lower bounds in the generic group model are also provided for these new assumptions.

**Organization.** In Section 2, we recall the formal model of security for 3-party password-based authenticated key exchange. Next, in Section 3, we recall the definitions of the standard Diffie-Hellman assumptions and introduce some new variants of these assumptions, on which the security of our protocol is based. We also present some relations between these assumptions. Section 4 then presents our 3-party password-based key exchange protocol, called 3PAKE, along with its security claims. Some important remarks are also presented in Section 4.

## 2 Definitions

We now recall the formal security model for 3-party password-authenticated key exchange protocols introduced in [1], which in turn builds upon those of Bellare and Rogaway [4, 5] and that of Bellare, Pointcheval, and Rogaway [3].

Protocol participants. The distributed system we consider is made up of three disjoint sets: $\mathcal{S}$, the set of trusted servers; $\mathcal{C}$, the set of honest clients; and $\mathcal{E}$, the set of malicious clients. We also denote the set of all clients by $\mathcal{U}$. That is, $\mathcal{U} = \mathcal{C} \cup \mathcal{E}$. As in [1], we also assume $\mathcal{S}$ to contain only a single trusted server.

Long-lived keys. Each participant $U \in \mathcal{U}$ holds a password $pw_U$. The server $S$ holds a vector $\mathsf{pw}_S = \langle pw_U \rangle_{U \in \mathcal{U}}$ with an entry for each client.

Execution of the protocol. The interaction between an adversary $\mathcal{A}$ and the protocol participants occurs only via oracle queries, which model the adversary capabilities in a real attack. While in a concurrent model, several instances may be active at any given time, only one active user instance is allowed for a given intended partner and password in a non-concurrent model. Let $U^i$ denote the instance $i$ of a participant $U$ and let $b$ be a bit chosen uniformly at random. These queries are as follows:

- *Execute*$(U_1^{i_1}, S^j, U_2^{i_2})$: This query models passive attacks in which the attacker eavesdrops on honest executions among client instances $U_1^{i_1}$ and $U_2^{i_2}$ and the server instance $S^j$. The output of this query consists of the messages that were exchanged during the honest execution of the protocol.
- *Reveal*$(U^i)$: This query models the misuse of session keys by clients. It returns to the adversary the session key of client instance $U^i$, if the latter is defined.
- *SendClient*$(U^i, m)$: This query models an active attack. It outputs the message that client instance $U^i$ would generate upon receipt of message $m$.

- *SendServer*$(S^j, m)$: This query models an active attack against a server. It outputs the message that server instance $S^j$ would generate upon receipt of message $m$.
- *Test*$(U^i)$: This query is used to measure the semantic security of the session key of client instance $U^i$, if the latter is defined. If the key is not defined, it returns $\perp$. Otherwise, it returns either the session key held by client instance $U^i$ if $b = 0$ or a random of key of the same size if $b = 1$.

Notation. Following [1], which in turn follows [4, 5], an instance $U^i$ is said to be *opened* if a query *Reveal*$(U^i)$ has been made by the adversary. We say an instance $U^i$ is *unopened* if it is not *opened*. We say an instance $U^i$ has *accepted* if it goes into an accept mode after receiving the last expected protocol message.

Partnering. The definition of partnering uses the notion of session identifications (*sid*), which in our case is the partial transcript of the conversation between the clients and the server before the acceptance. More specifically, two instances $U_1^i$ and $U_2^j$ are said to be partners if the following conditions are met: (1) Both $U_1^i$ and $U_2^j$ accept; (2) Both $U_1^i$ and $U_2^j$ share the same *sid*; (3) The partner identification for $U_1^i$ is $U_2^j$ and vice-versa; and (4) No instance other than $U_1^i$ and $U_2^j$ accepts with a partner identification equal to $U_1^i$ or $U_2^j$.

Freshness. An instance $U^i$ is considered *fresh* if that it has *accepted*, both $U^i$ and its partner (as defined by the partner function) are *unopened* and they are both instances of honest clients.

AKE semantic security. Consider an execution of the key exchange protocol $P$ by the adversary $\mathcal{A}$, in which the latter is given access to the *Execute*, *SendClient*, *SendServer*, and *Test* oracles and asks at most one *Test* query to a *fresh* instance of an honest client. Let $b'$ be his output. Such an adversary is said to win the experiment defining the semantic security if $b' = b$, where $b$ is the hidden bit used by the *Test* oracle. Let Succ denote the event in which the adversary wins this game.

The *advantage* of $\mathcal{A}$ in violating the AKE semantic security of the protocol $P$ and the *advantage function* of the protocol $P$, when passwords are drawn from a dictionary $\mathcal{D}$, are defined, respectively, as follows:

$$\mathbf{Adv}^{\mathrm{ake}}_{P,\mathcal{D}}(\mathcal{A}) = 2 \cdot \Pr[\,\textsc{Succ}\,] - 1 \quad \text{and} \quad \mathbf{Adv}^{\mathrm{ake}}_{P,\mathcal{D}}(t, R) = \max_{\mathcal{A}}\{\, \mathbf{Adv}^{\mathrm{ake}}_{P,\mathcal{D}}(\mathcal{A})\,\},$$

where maximum is over all $\mathcal{A}$ with time-complexity at most $t$ and using resources at most $R$ (such as the number of oracle queries). The definition of time-complexity is the usual one, which includes the maximum of all execution times in the experiments defining the security plus the code size. The probability rescaling was added to make the advantage of an adversary that simply guesses the bit $b$ equal to 0.

A 3-party password-based key exchange protocol $P$ is said to be semantically secure if the advantage $\mathbf{Adv}^{\mathrm{ake}}_{P,\mathcal{D}}$ is only negligibly larger than $kn/|\mathcal{D}|$, where $n$ is number of active sessions and $k$ is a constant. Note that $k = 1$ is the best one can hope for since an adversary that simply guesses the password in each of the active sessions has an advantage of $n/|\mathcal{D}|$.

## 3 Diffie-Hellman Assumptions

In this section, we recall the definitions of standard Diffie-Hellman assumptions and introduce some new variants, which we use in the security proof of our protocol. We also present some relations between these assumptions.

Henceforth, we assume a finite cyclic group $G$ of prime order $p$ generated by an element $g$. We also call the tuple $\mathbb{G} = (G, g, p)$ a represented group.

**Computational Diffie-Hellman Assumption:** CDH. The CDH assumption in a represented group $\mathbb{G}$ states that given $g^u$ and $g^v$, where $u, v$ were drawn at random from $\mathsf{Z}_p$, it is hard to compute $g^{uv}$. This can be defined more precisely by considering an Experiment $\mathbf{Exp}^{\mathrm{cdh}}_{\mathbb{G}}(\mathcal{A})$, in which we select two values $u$ and $v$ in $\mathsf{Z}_p$, compute $U = g^u$, and $V = g^v$, and then give both $U$ and $V$ to $\mathcal{A}$. Let $Z$ be the output of $\mathcal{A}$. Then, the Experiment $\mathbf{Exp}^{\mathrm{cdh}}_{\mathbb{G}}(\mathcal{A})$ outputs 1 if $Z = g^{uv}$ and 0 otherwise. We define the *advantage* of $\mathcal{A}$ in violating the CDH assumption as $\mathbf{Adv}^{\mathrm{cdh}}_{\mathbb{G}}(\mathcal{A}) = \Pr[\,\mathbf{Exp}^{\mathrm{cdh}}_{\mathbb{G}}(\mathcal{A}) = 1\,]$ and the *advantage function* of the group, $\mathbf{Adv}^{\mathrm{cdh}}_{\mathbb{G}}(t)$, as the maximum value of $\mathbf{Adv}^{\mathrm{cdh}}_{\mathbb{G}}(\mathcal{A})$ over all $\mathcal{A}$ with time-complexity at most $t$.

**Decisional Diffie-Hellman Assumption:** DDH. Roughly, the DDH assumption states that the distributions $(g^u, g^v, g^{uv})$ and $(g^u, g^v, g^w)$ are computationally indistinguishable when $u, v, w$ are drawn at random from $\mathsf{Z}_p$. As before, we can define the DDH assumption more formally by defining two experiments, $\mathbf{Exp}^{\mathrm{ddh\text{-}real}}_{\mathbb{G}}(\mathcal{A})$ and $\mathbf{Exp}^{\mathrm{ddh\text{-}rand}}_{\mathbb{G}}(\mathcal{A})$. In both experiments, we compute two values $U = g^u$ and $V = g^v$ as before. But in addition to that, we also provide a third input, which is $g^{uv}$ in $\mathbf{Exp}^{\mathrm{ddh\text{-}real}}_{\mathbb{G}}(\mathcal{A})$ and $g^z$ for a random $z$ in $\mathbf{Exp}^{\mathrm{ddh\text{-}rand}}_{\mathbb{G}}(\mathcal{A})$. The goal of the adversary is to guess a bit indicating the experiment he thinks he is in. We define the *advantage* of $\mathcal{A}$ in violating the DDH assumption, $\mathbf{Adv}^{\mathrm{ddh}}_{\mathbb{G}}(\mathcal{A})$, as $\Pr[\,\mathbf{Exp}^{\mathrm{ddh\text{-}real}}_{\mathbb{G}}(\mathcal{A}) = 1\,] - \Pr[\,\mathbf{Exp}^{\mathrm{ddh\text{-}rand}}_{\mathbb{G}}(\mathcal{A}) = 1\,]$. The *advantage function* of the group, $\mathbf{Adv}^{\mathrm{ddh}}_{\mathbb{G}}(t)$, is then defined in a similar manner.

**Chosen-Basis Decisional Diffie-Hellman Assumptions.** The security of our protocol relies on two new variations of the DDH assumption, which we call *Chosen-basis Decisional Diffie-Hellman* assumptions 1 and 2, where 1 and 2 denote the number of values outputted by the adversary at the end of the first phase. So, let us start by motivating the first of these, the CDDH1 assumption. A similar argument can be used to justify our second assumption, CDDH2, and hence we only provide its formal definition.

The CDDH1 assumption considers an adversary running in two stages. In a $\mathsf{find}$ stage, the adversary is given three values $U = g^u$, $V = g^v$, and $X = g^x$, where $u$, $v$, and $x$ are random elements in $\mathsf{Z}_p$. The adversary should then select an element $Y$ in $G$. Using $Y$, we then consider two games. In the first game ($b = 0$), we pick a random bit $b_0$ and set another bit $b_1 = b_0$ to the same value.

We then choose two secret random values $r_0$ and $r_1$, we compute two pairs of values $(X_0, K_0)$ and $(X_1, K_1)$ using bits $r_{b_0}$ and $r_{b_1}$ as in Definition 1 below and the value $Y' = Y^{r_0}$, and we give them to the adversary. In other words, in this game, we compute both pairs using the same exponent, which may or may not be the same used in the computation of $Y'$ from $Y$, the value previously chosen by the adversary. The second game ($b = 1$) is similar to the first one except that $b_1$ is set to $1 - b_0$ and hence the pairs $(X_0, K_0)$ and $(X_1, K_1)$ are computed using different exponents. The adversary wins if he guesses correctly the bit $b = b_0 \oplus b_1$.

To understand the subtlety of the assumption, let us consider the different strategies the adversary may take. First, if the adversary chooses $Y = g^y$ knowing its discrete log $y$, then he can compute $\mathrm{CDH}(X/U, Y)$ as well as $g^{r_0}$. He can also verify that each key $K_i$ is in fact $X_i^y$. Hence, the keys $K_i$ do not leak any additional information. Let $g_0 = X/U$ and $g_1 = X/V$. Then $X_i = g_i^{r_{b_i}}$. Thus, the adversary in this case needs to be able to tell whether the same exponent is used in $X_i$ knowing only $g^{r_0}$. We believe this is not easy.

Now let us consider the case in which the adversary chooses $Y$ as a function of the inputs that he was given at the $\mathsf{find}$ stage (hence not knowing $y$). In this case, the adversary should not be able to compute the CDH value and hence the values $K_i$ are not of much help either. Consider the case where he chooses $Y = X/U$. Then, using $Y'$, the adversary can easily know the value of $b_0$ by checking whether $X_0 = Y'$. However, that does not seem to be of much help since he now needs to tell whether $X_0 = g_0^{r_{b_0}}$ was computed using the same exponent as $X_1 = g_1^{r_{b_1}}$. Knowing $b_0$ does not seem of any help. We now proceed with the formal definitions.

**Definition 1** (CDDH1)**.** *Let $\mathbb{G} = (G, g, p)$ be a represented group and let $\mathcal{A}$ be an adversary. Consider the following experiment, defined for $b = 0, 1$, where $U$, $V$, and $X$ are elements in $G$ and $r_0$ and $r_1$ are elements in $\mathsf{Z}_p$.*

```
**Experiment** $\mathbf{Exp}^{\mathrm{cddh1}}_{\mathbb{G},b}(\mathcal{A}, U, V, X, r_0, r_1)$
    $(Y, s) \overset{R}{\leftarrow} \mathcal{A}(\mathsf{find}, U, V, X)$
    $b_0 \overset{R}{\leftarrow} \{0, 1\}$ ; $b_1 = b \oplus b_0$
    $X_0 \leftarrow (X/U)^{r_{b_0}}$ ;   $K_0 \leftarrow \mathrm{CDH}(X/U, Y)^{r_{b_0}}$
    $X_1 \leftarrow (X/V)^{r_{b_1}}$ ;   $K_1 \leftarrow \mathrm{CDH}(X/V, Y)^{r_{b_1}}$
    $Y' \leftarrow Y^{r_0}$
    $d \leftarrow \mathcal{A}(\mathsf{guess}, s, X_0, K_0, X_1, K_1, Y')$
    **return** $d$
```

*Now define the advantage of $\mathcal{A}$ in violating the chosen-basis decisional Diffie-Hellman 1 assumption with respect to $(U, V, X, r_0, r_1)$, the advantage of $\mathcal{A}$, and the advantage function of the group, respectively, as follows:*

$$\mathbf{Adv}^{\mathrm{cddh1}}_{\mathbb{G}}(\mathcal{A}, U, V, X, r_0, r_1) = 2 \cdot \Pr[\,\mathbf{Exp}^{\mathrm{cddh1}}_{\mathbb{G},b}(\mathcal{A}, U, V, X, r_0, r_1) = b\,] - 1$$

$$\mathbf{Adv}^{\mathrm{cddh1}}_{\mathbb{G}}(\mathcal{A}) = \mathbf{E}_{U,V,X,r_0,r_1}\left[\mathbf{Adv}^{\mathrm{cddh1}}_{\mathbb{G}}(\mathcal{A}, U, V, X, r_0, r_1)\right]$$

$$\mathbf{Adv}^{\mathrm{cddh1}}_{\mathbb{G}}(t) = \max_{\mathcal{A}}\{\,\mathbf{Adv}^{\mathrm{cddh1}}_{\mathbb{G}}(\mathcal{A})\,\},$$

*where the maximum is over all $\mathcal{A}$ with time-complexity at most $t$.* ♢

**Definition 2** (CDDH2). *Let $\mathbb{G} = (G, g, p)$ be a represented group and let $\mathcal{A}$ be an adversary. Consider the following experiment, defined for $b = 0, 1$, where $U$ and $V$ are elements in $G$ and $r_0$ and $r_1$ are elements in $\mathsf{Z}_p$.*

```
Experiment $\mathbf{Exp}^{\mathrm{cddh2}}_{\mathbb{G},b}(\mathcal{A}, U, V, r_0, r_1)$
    $(X, Y, s) \overset{R}{\leftarrow} \mathcal{A}(\mathsf{find}, U, V)$
    $b_0 \overset{R}{\leftarrow} \{0, 1\}$ ; $b_1 = b \oplus b_0$
    $X_0 \leftarrow (X/U)^{r_{b_0}}$ ;   $X_1 \leftarrow (X/V)^{r_{b_1}}$ ;   $Y' \leftarrow Y^{r_0}$
    $d \leftarrow \mathcal{A}(\mathsf{guess}, s, X_0, X_1, Y')$
    return $d$
```

*We define the advantage of $\mathcal{A}$ in violating the chosen-basis decisional Diffie-Hellman 2 assumption with respect to $(U, V, r_0, r_1)$, $\mathbf{Adv}^{\mathrm{cddh2}}_{\mathbb{G},\mathcal{A},U,V,r_0,r_1}$, the advantage of $\mathcal{A}$, $\mathbf{Adv}^{\mathrm{cddh2}}_{\mathbb{G}}(\mathcal{A})$, and the advantage function of the group, $\mathbf{Adv}^{\mathrm{cddh2}}_{\mathbb{G}}(t)$, as in Definition 1.* ♢

**Password-Based Chosen-Basis Decisional Diffie-Hellman Assumptions.** The actual proof of security of our protocol uses password-related versions of the chosen-basis decisional Diffie-Hellman assumptions, which we call *password-based chosen-basis decisional Diffie-Hellman* assumptions 1 and 2.

**Definition 3** (PCDDH1). *Let $\mathbb{G} = (G, g, p)$ be a represented group and let $\mathcal{A}$ be an adversary. Consider the following experiment, defined for $b = 0, 1$, where $\mathcal{P}$ is a random function from $\{1, \ldots, n\}$ into $G$, $X$ is an element in $G$, $k$ is a password in $\{1, \ldots, n\}$, and $r_0$ and $r_1$ are elements in $\mathsf{Z}_p$.*

```
Experiment $\mathbf{Exp}^{\mathrm{pcddh1}}_{\mathbb{G},n,b}(\mathcal{A}, \mathcal{P}, X, k, r_0, r_1)$
    $(Y, s) \overset{R}{\leftarrow} \mathcal{A}^{\mathcal{P}}(\mathsf{find}, X)$
    $U \leftarrow \mathcal{P}(k)$ ; $X' \leftarrow (X/U)^{r_b}$ ;   $K \leftarrow \mathrm{CDH}(X/U, Y)^{r_b}$ ;   $Y' \leftarrow Y^{r_0}$
    $d \leftarrow \mathcal{A}(\mathsf{guess}, s, X', Y', K, k)$
    return $d$
```

*We define the advantage of $\mathcal{A}$ in violating the password-based chosen-basis decisional Diffie-Hellman 1 assumption with respect to $(\mathcal{P}, X, k, r_0, r_1)$, $\mathbf{Adv}^{\mathrm{pcddh1}}_{\mathbb{G},n}$ $(\mathcal{A}, \mathcal{P}, X, k, r_0, r_1)$, the advantage of $\mathcal{A}$, $\mathbf{Adv}^{\mathrm{pcddh1}}_{\mathbb{G},n}(\mathcal{A}, \mathcal{P})$, and the advantage function of the group, $\mathbf{Adv}^{\mathrm{pcddh1}}_{\mathbb{G},n}(t, \mathcal{P})$, as in Definition 1.* ♢

**Definition 4** (PCDDH2). *Let $\mathbb{G} = (G, g, p)$ be a represented group and let $\mathcal{A}$ be an adversary. Consider the following experiment, defined for $b = 0, 1$, where $\mathcal{P}$ is a random function from $\{1, \ldots, n\}$ into $G$, $k$ is a password in $\{1, \ldots, n\}$, and $r_0$ and $r_1$ are elements in $\mathsf{Z}_p$.*

```
Experiment $\mathbf{Exp}^{\mathrm{pcddh2}}_{\mathbb{G},n,b}(\mathcal{A}, \mathcal{P}, k, r_0, r_1)$
    $(X, Y, s) \overset{R}{\leftarrow} \mathcal{A}^{\mathcal{P}}(\mathsf{find})$
    $U \leftarrow \mathcal{P}(k)$ ;   $X' \leftarrow (X/U)^{r_b}$ ;   $Y' \leftarrow Y^{r_0}$
    $d \leftarrow \mathcal{A}^{\mathcal{P}}(\mathsf{guess}, s, X', Y', k)$
    return $d$
```

*We define the advantage of $\mathcal{A}$ in violating the password-based chosen-basis decisional Diffie-Hellman 2 assumption with respect to $(\mathcal{P}, k, r_0, r_1)$, $\mathbf{Adv}^{\text{pcddh2}}_{\mathbb{G},n}(\mathcal{A}, \mathcal{P}, k, r_0, r_1)$, the advantage of $\mathcal{A}$, $\mathbf{Adv}^{\text{pcddh2}}_{\mathbb{G},n}(\mathcal{A}, \mathcal{P})$, and the advantage function of the group, $\mathbf{Adv}^{\text{pcddh2}}_{\mathbb{G},n}(t, \mathcal{P})$, as in Definition 1.* ♢

**Relations Between the** PCDDH1 **and** CDDH1 **Problems.** The following two lemmas, whose proofs can be found in the full version of this paper [2], present relations between the PCDDH1 and CDDH1 problems. The first result is meaningful for small $n$ (polynomially bounded in the asymptotic framework). The second one considers larger dictionaries.

**Lemma 1.** *Let $\mathbb{G} = (G, g, p)$ be a represented group and let $n$ be an integer. If there exists a distinguisher $\mathcal{A}$ such that $\mathbf{Adv}^{\text{pcddh1}}_{\mathbb{G},n}(\mathcal{A}) \geq \frac{2}{n} + \epsilon$, then there exists a distinguisher $\mathcal{B}$ and a subset $S$ of $G^3 \times \mathsf{Z}_p^2$ of probability greater than $\epsilon/8n^2$ such that for any $(U, V, X, r_0, r_1) \in S$, $\mathbf{Adv}^{\text{cddh1}}_{\mathbb{G},n}(\mathcal{B}, U, V, X, r_0, r_1) \geq \frac{\epsilon^2}{8}$.*

**Lemma 2.** *Let $\mathbb{G} = (G, g, p)$ be a represented group and let $n$ be an integer. If there exists a distinguisher $\mathcal{A}$ such that $\mathbf{Adv}^{\text{pcddh1}}_{\mathbb{G},n}(\mathcal{A}) \geq \epsilon \geq \frac{16}{n}$, then there exists a distinguisher $\mathcal{B}$ and a subset $S$ of $G^3 \times \mathsf{Z}_p^2$ of probability greater than $\epsilon^3/2^{10}$ such that for any $(U, V, X, r_0, r_1) \in S$, $\mathbf{Adv}^{\text{cddh1}}_{\mathbb{G},n}(\mathcal{B}, U, V, X, r_0, r_1) \geq \frac{\epsilon^2}{8}$.*

**Relations Between the** PCDDH2 **and** CDDH2 **Problems.** The following two lemmas, whose proofs can be easily derived from the proofs of the previous two lemmas, present relations between the PCDDH2 and CDDH2 problems. While the first result is meaningful for small values of $n$, the second one considers larger values.

**Lemma 3.** *Let $\mathbb{G} = (G, g, p)$ be a represented group and let $n$ be an integer. If there exists a distinguisher $\mathcal{A}$ such that $\mathbf{Adv}^{\text{pcddh2}}_{\mathbb{G},n}(\mathcal{A}) \geq \frac{2}{n} + \epsilon$, then there exists a distinguisher $\mathcal{B}$ and a subset $S$ of $G^2 \times \mathsf{Z}_p^2$ of probability greater than $\epsilon/8n^2$ such that for any $(U, V, r_0, r_1) \in S$ $\mathbf{Adv}^{\text{cddh2}}_{\mathbb{G},n}(\mathcal{B}, U, V, r_0, r_1) \geq \frac{\epsilon^2}{8}$.*

**Lemma 4.** *Let $\mathbb{G} = (G, g, p)$ be a represented group and let $n$ be an integer. If there exists a distinguisher $\mathcal{A}$ such that $\mathbf{Adv}^{\text{pcddh1}}_{\mathbb{G},n}(\mathcal{A}) \geq \epsilon \geq \frac{16}{n}$, then there exists a distinguisher $\mathcal{B}$ and a subset $S$ of $G^2 \times \mathsf{Z}_p^2$ of probability greater than $\epsilon^3/2^{10}$ such that for any $(U, V, r_0, r_1) \in S$ $\mathbf{Adv}^{\text{cddh1}}_{\mathbb{G},n}(\mathcal{B}, U, V, r_0, r_1) \geq \frac{\epsilon^2}{8}$.*

**Distinguishers.** In all of the above relations, we show that if there exists an adversary against the password version of the chosen-basis decisional problem that is capable of doing better than just guessing the password, then we can construct a distinguisher for underlying chosen-basis decisional problem, whose success probability is non-negligible over a non-negligible subset of the probability space. Even though these results provide enough evidence of the hardness of breaking the original password-based problem, one may want a more concrete result that works for the most of the probability space. The next lemma, whose proof can be found in the full version of this paper [2], proves just that. More precisely, it shows that if a good distinguisher exists for a non-negligible portion of the probability space, then the same distinguisher is a good distinguisher either for the entire probability space or for at least half of it.

**Lemma 5 (Amplification Lemma).** *Let $E^b(x)$ be an experiment for $b \in \{0,1\}$ and $x \in S$. Let $D$ be a distinguisher between two experiments $E^0(x)$ and $E^1(x)$ with advantage $\epsilon$ for $x \in S'$, where $S' \subset S$ is of measure $\mu = |S'|/|S|$:*

$$\Pr_x[x \in S'] = \mu; \qquad \Pr_{b,x}[E^b(D,x) = b \mid x \in S'] \geq \frac{1}{2} + \frac{\epsilon}{2}.$$

*Then either $D$ is a good distinguisher on the whole set $S$:*

$$\Pr_{b,x}[E^b(D,x) = b] \geq \frac{1}{2} + \frac{\mu\epsilon}{4},$$

*or $D$ is a good distinguisher for $S'$ and $S \backslash S'$, one of which is a subset of measure greater than or equal to one half:*

$$\begin{aligned} \Pr_x[x \in S'] = \mu \quad & \Pr_{b,x}[E^b(D,x) = b \mid x \in S'] \geq \frac{1}{2} + \frac{\epsilon}{2}; \\ \Pr_x[x \in S \backslash S'] = 1 - \mu \quad & \Pr_{b,x}[E^b(D,x) = b \mid x \in S \backslash S'] \leq \frac{1}{2} - \frac{\mu\epsilon}{4}. \end{aligned}$$

**Lower Bounds.** The following lemma, whose proof can be found in the full version of this paper [2], gives an upper bound on the advantage of any adversary for the CDDH1 or CDDH2 problem in the generic model of Shoup [23]. From it, it follows that any generic algorithm capable of solving the CDDH1 or CDDH2 problem with success probability bounded away from 1/2 has to perform at least $\Omega(p^{1/2})$ group operations. Please refer to [23] for a description of the model.

**Lemma 6.** *Let $p$ be a prime number and let $\sigma$ represent a random injective mapping of $\mathsf{Z}_p$ into a set $S$ of bit strings of cardinality at least $p$. Let $\mathcal{A}$ be a distinguisher for the* CDDH1 *or the* CDDH2 *problem in the generic setting making at most $m$ queries to the group oracle associated with $\sigma$. Then, the advantage of $\mathcal{A}$ is at most $O(m^2/p)$.*

## 4 Our 3-Party Password-Based Protocol

In this section, we introduce our new protocol, a non-concurrent 3-party password-based authenticated key exchange protocol called 3PAKE, whose security proof is in the random oracle model. It assumes that the clients willing to establish a common secret session key share passwords with a common server. Even though the proof of security assumes a non-concurrent model, we outline in Section 4 ways in which one can modify our protocol to make it concurrent.

**Description.** Our 3-party password-based protocol, 3PAKE, is based the on password-based key exchange protocols in [6, 9, 21], which in turn are based on the encrypted key exchange of Bellovin and Merritt [7]. The description of 3PAKE is given in Figure 1, where $(G, g, p)$ is the represented group; $\ell_r$ and $\ell_k$ are security parameters; and $G_1 : \mathcal{D} \rightarrow G$, $G_2 : \{0,1\}^{\ell_r} \times \mathcal{D} \times G \rightarrow G$, and $H : \mathcal{U}^3 \times \{0,1\}^{\ell_r} \times G^4 \rightarrow \{0,1\}^{\ell_k}$ are random oracles.

The protocol consists of two rounds of message. First, each client chooses an ephemeral public key by choosing a random element in $\mathsf{Z}_p$ and raising $g$ to the that power, encrypts it using the output of the hash function $G_1$ with his password as the input, and sends it to the server. Upon receiving a message from each client, the server decrypts these messages to recover each client's ephemeral public key, chooses a random index $r \in \mathsf{Z}_p$ and a random element $R \in \{0,1\}^{\ell_r}$, exponentiates each of the ephemeral public keys to the $r$-th power, and re-encrypts them using the output of the hash function $G_2$, with $R$ and the appropriate first-round message and password as input.

In the second round of messages, the server sends to each client the encrypted value of the randomized ephemeral public key of their partner along with the messages that the server exchanged with that partner, which are omitted in Figure 1 for clarity. Upon receiving a message from the server, each client recovers the randomized ephemeral public key of his partner, computes the Diffie-Hellman key $K$, and the session key $SK$ via a hash function $H$ using as input $K$ and the transcript of the conversation among the clients and the server. The session identification is defined to be the transcript $T = (R, X^\star, Y^\star, \overline{X}^\star, \overline{Y}^\star)$ of the conversation among the server and clients, along with their identity strings.

$$\begin{array}{ccccc}
 & & \text{Public information: } G, g, p, \ell_r, \ell_k, G_1, G_2, H & & \\
\text{Client } A & & \text{Server } S & & \text{Client } B \\
pw_A \in \mathcal{D} & & pw_A, pw_B \in \mathcal{D} & & pw_B \in \mathcal{D} \\
x \xleftarrow{R} \mathsf{Z}_p \;;\; X \leftarrow g^x & & r \xleftarrow{R} \mathsf{Z}_p \;;\; R \xleftarrow{R} \{0,1\}^{\ell_r} & & y \xleftarrow{R} \mathsf{Z}_p \;;\; Y \leftarrow g^y \\
\mathsf{pw}_{A,1} \leftarrow G_1(pw_A) & & & & \mathsf{pw}_{B,1} \leftarrow G_1(pw_B) \\
X^\star \leftarrow X \cdot \mathsf{pw}_{A,1} & & & & Y^\star \leftarrow Y \cdot \mathsf{pw}_{B,1} \\
 & \xrightarrow{X^\star} & & \xleftarrow{Y^\star} & \\
 & & \mathsf{pw}_{A,1} \leftarrow G_1(pw_A) & & \\
 & & \mathsf{pw}_{B,1} \leftarrow G_1(pw_B) & & \\
 & & X \leftarrow X^\star / \mathsf{pw}_{A,1} & & \\
 & & Y \leftarrow Y^\star / \mathsf{pw}_{B,1} & & \\
 & & \overline{X} \leftarrow X^r & & \\
 & & \overline{Y} \leftarrow Y^r & & \\
 & & \mathsf{pw}_{A,2} \leftarrow G_2(R, pw_A, X^\star) & & \\
 & & \mathsf{pw}_{B,2} \leftarrow G_2(R, pw_B, Y^\star) & & \\
 & & \overline{Y}^\star \leftarrow \overline{Y} \cdot \mathsf{pw}_{A,2} & & \\
 & & \overline{X}^\star \leftarrow \overline{X} \cdot \mathsf{pw}_{B,2} & & \\
 & \xleftarrow{R, \overline{Y}^\star} & & \xrightarrow{R, \overline{X}^\star} & \\
\mathsf{pw}_{A,2} \leftarrow G_2(R, pw_A, X^\star) & & & & \mathsf{pw}_{B,2} \leftarrow G_2(R, pw_B, Y^\star) \\
\overline{Y} \leftarrow \overline{Y}^\star / \mathsf{pw}_{A,2} \;;\; K \leftarrow \overline{Y}^x & & & & \overline{X} \leftarrow \overline{X}^\star / \mathsf{pw}_{B,2} \;;\; K \leftarrow \overline{X}^y \\
T \leftarrow R, X^\star, Y^\star, \overline{X}^\star, \overline{Y}^\star & & & & T \leftarrow R, X^\star, Y^\star, \overline{X}^\star, \overline{Y}^\star \\
SK \leftarrow H(A, B, S, T, K) & & & & SK \leftarrow H(A, B, S, T, K)
\end{array}$$

**Fig. 1.** 3PAKE: A provably-secure 3-party password-based authenticated key exchange protocol

Correctness. In an honest execution of the protocol in Figure 1, we have $\overline{Y} = Y^r = g^{yr}$ and $\overline{X} = X^r = g^{xr}$. Hence, $K = \overline{Y}^x = \overline{X}^y = g^{xyr}$.

Efficiency. 3PAKE is quite efficient, not requiring much computational power from the server. Note that the amount of computation performed by the server in this case is comparable to that of each user. That is at least half the amount of computation that it would be required if the server were to perform a separate 2-party password-based encrypted key exchange with each user.

Rationale for the scheme. As pointed out in the introduction, the random value $r$ is used by the server to hide the password of one user with respect to other users. For this same reason, it is also crucial that the server rejects any value $X^\star$ or $Y^\star$ whose underlying value $X$ or $Y$ is equal to 1. This is omitted in Figure 1 for clarity reasons only.

The reason for using two different masks $\mathsf{pw}_{A,1}$ and $\mathsf{pw}_{A,2}$ in each session, on the other hand, is a little more intricate and is related to our proof technique. More precisely, in our proof of security, we embed instances of the CDDH1 and CDDH2 problems in $\mathsf{pw}_{A,1}$ and $\mathsf{pw}_{A,2}$ and we hope to get an answer for these problems from the list of queries that the adversary makes to the $G_1$ and $G_2$ oracles. Unfortunately, this does not appear to be possible when the values of $\mathsf{pw}_{A,1}$ and $\mathsf{pw}_{A,2}$ are fixed for all sessions since a powerful adversary could be able to learn the values of $\mathsf{pw}_{A,1}$ and $\mathsf{pw}_{A,2}$ and break the semantic security of the scheme without querying the oracles for $G_1$ and $G_2$.

To see how, let us assume two fixed but random values for $\mathsf{pw}_{A,1}$ and $\mathsf{pw}_{A,2}$ and that we are dealing with an adversary that knows the password of a legitimate but malicious user. Let us also assume that the adversary is capable of breaking the computational Diffie-Hellman inversion (CDHI) problem, in which the goal is to compute $g^y$ from $g$, $g^x$, and $g^{xy}$. Since in the security model, the adversary is allowed to intercept and replay messages, he can play the role of the partner of $A$ and ask a given query $(A, g^x \cdot \mathsf{pw}_{A,1})$ twice to the server. From the answers to these queries, the adversary would be able to compute two sets of values $(g^x \cdot \mathsf{pw}_{A,1}, g^y, g^{xr}, g^{yr} \cdot \mathsf{pw}_{A,2})$ and $(g^x \cdot \mathsf{pw}_{A,1}, g^y, g^{xr'}, g^{yr'} \cdot \mathsf{pw}_{A,2})$ based on different values $r$ and $r'$. By dividing the last two terms of each set, the adversary can compute $g^{(r'-r)x}$ and $g^{(r'-r)y}$. Moreover, since the adversary plays the role of the partner of $A$ and knows $y$, he can also compute $g^{r'-r}$. Hence, the adversary can learn the values of $g$, $g^{r'-r}$, and $g^{(r'-r)x}$ as well as $g^x \cdot \mathsf{pw}_{A,1}$. By solving the CDHI problem, he can also learn the value of $g^x$ from $g$, $g^{r'-r}$, and $g^{(r'-r)x}$. Thus, he can recover $\mathsf{pw}_{A,1}$ without querying the oracle $G_1$ on various inputs $pw$. Moreover, since such adversary is capable of computing $g^r$ from $g$, $g^x$, and $g^{rx}$, and hence capable of computing $g^{ry}$, he can also learn the value of $\mathsf{pw}_{A,2}$ without querying the oracle $G_2$.

**Security.** As the following theorem states, 3PAKE is a secure non-concurrent 3-party password-based key exchange protocol as long as the CDH, DDH, PCDDH1, and PCDDH2 problems are hard in $\mathbb{G}$. As shown in Section 3, the latter two problems are hard as long as CDDH1 and CDDH2 are hard in $\mathbb{G}$. Please note that the proof of security assumes $\mathcal{D}$ to be a uniformly distributed dictionary.

**Theorem 1.** *Let $\mathbb{G} = (G, g, p)$ be a represent group of prime order $p$ and let $\mathcal{D}$ be a uniformly distributed dictionary of size $|\mathcal{D}|$. Let* 3PAKE *describe the encrypted key exchange protocol associated with these primitives as defined in Figure 1. Then, for any numbers $t$, $q_{\text{server}}$, $q_{\text{start}}$, $q_{\text{exe}}$, $q_{G_1}$, $q_{G_2}$, and $q_H$,*

$$\mathbf{Adv}^{\text{ake}}_{\mathsf{3PAKE},\mathbb{G},\mathcal{D}}(t, q_{\text{server}}, q_{\text{start}}, q_{\text{exe}}, q_{G_1}, q_{G_2}, q_H) \ \leq$$

$$\frac{2\ q_{\text{start}}}{|\mathcal{D}|} + \frac{q_{G_1}^2 + q_{G_2}^2 + (q_{\text{exe}} + q_{\text{start}})^2}{p} + 4\ q_{\text{exe}}\ \mathbf{Adv}^{\text{ddh}}_{\mathbb{G}}(t) +$$
$$2 \cdot q_{\text{server}} \cdot \max\{\, 2 \cdot \mathbf{Adv}^{\text{pcddh1}}_{\mathbb{G},|\mathcal{D}|}(q_{\text{start}} \cdot t)\ ,\ \mathbf{Adv}^{\text{pcddh2}}_{\mathbb{G},|\mathcal{D}|}(t)\,\} +$$
$$2\ q_{G_1}^2\ q_{G_2}^2\ q_H^2\ \mathbf{Adv}^{\text{cdh}}_{\mathbb{G}}(t + 3\tau_e) + 2\ \frac{q_{G_1} + q_{G_2}}{p} + 4\ \frac{q_H}{p}\ ,$$

*where $q_H$, $q_{G_1}$, and $q_{G_2}$ represent the number of queries to the $H$, $G_1$ and $G_2$ oracles, respectively; $q_{\text{exe}}$ represents the number of queries to the Execute oracle; $q_{\text{start}}$ represents the number of queries to the SendClient oracle used to initiate an client oracle instance; $q_{\text{server}}$ represents the number of queries to the SendServer oracle; and $\tau_e$ denotes the exponentiation computational time in $\mathbb{G}$.*

**Proof Idea.** Here we only present a brief sketch of the proof. We refer the reader to the full version of this paper [2] for the full proof of security. The proof for 3PAKE defines a sequence of hybrid experiments, starting with the real attack and ending in an experiment in which the adversary has no advantage. Each experiment addresses a different security aspect.

Experiments 1 through 5 show that the adversary gains no information from passive attacks. They do so by showing that keys generated in these sessions can be safely replaced by random ones as long as the DDH assumption holds in $\mathbb{G}$.

In Experiment 6, we change the simulation of the random oracle $H$ in all those situations for which the adversary may ask a valid test query. Such a change implies that the output of the test query is random and hence the advantage of the adversary in this case is 0. However, the difference between this experiment and previous still cannot be computed since it depends on the event AskH that the adversary asks certain queries to the random oracle $H$. Our goal at this point shifts to computing the probability of the event AskH.

In experiments 7 through 9, we deal with active attacks against the server. First, in Experiment 7, we show that the output values $\overline{X}^\star$ and $\overline{Y}^\star$ associated with honest users can be computed using random values and independently of each other as long as the PCDDH1 and PCDDH2 assumptions hold in $\mathbb{G}$. More precisely, we show how to upper-bound the difference in the probability of the event AskH using the PCDDH1 and PCDDH2 assumptions. Then, in the next two experiments, we show that for those cases in which we replaced $\overline{X}^\star$ and $\overline{Y}^\star$

with random values, the password is no longer used and that the Diffie-Hellman keys $K$ used to compute the session keys for these users are indistinguishable from random.

Finally, in Experiment 10, we consider active attacks against a user. More precisely, we show that we can answer all *SendClient* queries with respect to honest users using random values, without using the password of these users, and without changing the probability of the event AskH. Moreover, at this moment, we also show how to bound the probability of the event AskH based on the hardness of the CDH problem in $\mathbb{G}$ and on the probability that the adversary successfully guesses the password of an honest user during an active attack against that user.

**Concluding Remarks.** First, the main reason for assuming an underlying group $G$ of prime order $p$ is to ensure that the exponentiation of an element in the group other than the unit yields a generator. For the same reason, it is crucial for the server to check whether the elements to which it applies the randomization step are different from the unit element. Both these assumptions are implicitly made in several parts of the proof and they are essential for the security of our protocol.

Second, the proof of security for 3PAKE assumes a non-concurrent model, in which only one instance of each player can exist at a time. One can argue that such proof is not worth much as it rules out most interesting attack scenarios or makes the scheme too restrictive to be used in practice. To address the first of these concerns, we argue that, even though the non-concurrent scenario rules out a significant class of attacks, it still allows many interesting ones. For example, the identity-misbinding attacks in [12, 18] still work in the non-concurrent scenario. To address the second concern, we point out that several applications found in practice do not require concurrency. And even when they do require concurrent sessions, it is usually between different pairs of users. A simple modification is enough to make our protocol work in the latter case, by including the users' identification in the input of the $G_1$ and $G_2$ hash functions.

Third, if full concurrency is required, then one could modify 3PAKE to make it work in this new scenario by adding two extra flows at the beginning of the protocol going from the server to each of the two users. Such flows would include nonces in the input of the $G_1$ and $G_2$ hash functions. Each user would also have to add its own nonce to the input of the $G_1$ and $G_2$ hash functions, and send it to the server along with $X^\star$ or $Y^\star$. Moreover, the protocol's efficiency would remain almost the same, except for the number of rounds, but would still be significantly better than the round complexity of the generic construction in [1].

Finally, some of the problems that we found in our proof may be avoidable in the "ideal-cipher model," in which the encryption function is considered to be a truly random permutation. The reason for that is that non-linear properties of the ideal cipher model naturally remove the algebraic properties existent in the "one-time pad" version of the encryption function. Nonetheless, we opted to rely only on a single idealized model, the random oracle model, which is already a strong assumption as other papers have shown (e.g., [11]).

## Acknowledgements

This work has been supported in part by the European Commission through the IST Program under Contract IST-2002-507932 ECRYPT.

## References

1. M. Abdalla, P.-A. Fouque, and D. Pointcheval. Password-based authenticated key exchange in the three-party setting. In *PKC 2005*, Springer-Verlag LNCS 3386, 2005.
2. M. Abdalla and D. Pointcheval. Interactive Diffie-Hellman assumptions with applications to password-based authentication. Full version of current paper. Available from authors' web pages.
3. M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In *EUROCRYPT 2000*, Springer-Verlag LNCS 1807, 2000.
4. M. Bellare and P. Rogaway. Entity authentication and key distribution. In *CRYPTO'93*, Springer-Verlag LNCS 773, 1994.
5. M. Bellare and P. Rogaway. Provably secure session key distribution — the three party case. In *28th ACM STOC*. ACM Press, 1996.
6. M. Bellare and P. Rogaway. The AuthA protocol for password-based authenticated key exchange. Contributions to IEEE P1363, 2000.
7. S. M. Bellovin and M. Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *1992 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, 1992.
8. V. Boyko, P. MacKenzie, and S. Patel. Provably secure password-authenticated key exchange using Diffie-Hellman. In *EUROCRYPT 2000*, Springer-Verlag LNCS 1807, 2000.
9. E. Bresson, O. Chevassut, and D. Pointcheval. New security results on encrypted key exchange. In *PKC 2004*, Springer-Verlag LNCS 2947, 2004.
10. J. W. Byun, I. R. Jeong, D. H. Lee, and C.-S. Park. Password-authenticated key exchange between clients with different passwords. In *ICICS 02*, Springer-Verlag LNCS 2513, 2002.
11. R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. In *30th ACM STOC*. ACM Press, 1998.
12. R. Canetti and H. Krawczyk. Security analysis of IKE's signature-based key-exchange protocol. In *CRYPTO 2002*, Springer-Verlag LNCS 2442, 2002.
13. W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22:644–654, 1978.
14. R. Gennaro and Y. Lindell. A framework for password-based authenticated key exchange. In *EUROCRYPT 2003*, Springer-Verlag LNCS 2656, 2003.
15. O. Goldreich and Y. Lindell. Session-key generation using human passwords only. In *CRYPTO 2001*, Springer-Verlag LNCS 2139, 2001.
16. L. Gong. Optimal authentication protocols resistant to password guessing attacks. In *CSFW'95*, pages 24–29, 1995. IEEE Computer Society.
17. S. Halevi and H. Krawczyk. Public-key cryptography and password protocols. In *ACM Transactions on Information and System Security*, pages 524–543. 1999.
18. H. Krawczyk. SIGMA: The "SIGn-and-MAc" approach to authenticated Diffie-Hellman and its use in the IKE protocols. In *CRYPTO 2003*, Springer-Verlag LNCS 2729, 2003.

19. C.-L. Lin, H.-M. Sun, and T. Hwang. Three-party encrypted key exchange: Attacks and a solution. *ACM SIGOPS Operating Systems Review*, 34(4):12–20, Oct. 2000.
20. P. MacKenzie, S. Patel, and R. Swaminathan. Password-authenticated key exchange based on RSA. In *ASIACRYPT 2000*, Springer-Verlag LNCS 1976, 2000.
21. P. MacKenzie. The PAK suite: Protocols for password-authenticated key exchange. Contributions to IEEE P1363.2, 2002.
22. R. Needham and M. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(21):993–999, Dec. 1978.
23. V. Shoup. Lower bounds for discrete logarithms and related problems. In *EUROCRYPT'97*, Springer-Verlag LNCS 1233, 1997.
24. M. Steiner, G. Tsudik, and M. Waidner. Refinement and extension of encrypted key exchange. *ACM SIGOPS Operating Systems Review*, 29(3):22–30, July 1995.
25. S. Wang, J. Wang, and M. Xu. Weaknesses of a password-authenticated key exchange protocol between clients with different passwords. In *ACNS 04*, Springer-Verlag LNCS 3089, 2004.
26. H.-T. Yeh, H.-M. Sun, and T. Hwang. Efficient three-party authentication and key agreement protocols resistant to password guessing attacks. *Journal of Information Science and Engineering*, 19(6):1059–1070, Nov. 2003.

# Secure Biometric Authentication for Weak Computational Devices

Mikhail J. Atallah[1], Keith B. Frikken[1],
Michael T. Goodrich[2], and Roberto Tamassia[3]

[1] Department of Computer Sciences, Purdue University
{mja, kbf}@cs.purdue.edu
[2] Department of Computer Science, Univ. of California, Irvine
goodrich@ieee.org
[3] Department of Computer Science, Brown University
rt@cs.brown.edu

**Abstract.** This paper presents computationally "lightweight" schemes for performing biometric authentication that carry out the comparison stage without revealing any information that can later be used to impersonate the user (or reveal personal biometric information). Unlike some previous computationally expensive schemes — which make use of slower cryptographic primitives — this paper presents methods that are particularly suited to financial institutions that authenticate users with biometric smartcards, sensors, and other computationally limited devices. In our schemes, the client and server need only perform cryptographic hash computations on the feature vectors, and do not perform any expensive digital signatures or public-key encryption operations. In fact, the schemes we present have properties that make them appealing even in a framework of powerful devices capable of public-key signatures and encryptions. Our schemes make it computationally infeasible for an attacker to impersonate a user even if the attacker completely compromises the information stored at the server, including all the server's secret keys. Likewise, our schemes make it computationally infeasible for an attacker to impersonate a user even if the attacker completely compromises the information stored at the client device (but not the biometric itself, which is assumed to remain attached to the user and is not stored on the client device in any form).

**Keywords:** biometrics, authentication, smart cards, cryptographic hash functions.

[1] Portions of this work were supported by Grants IIS-0325345, IIS-0219560, IIS-0312357, and IIS-0242421 from the National Science Foundation, Contract N00014-02-1-0364 from the Office of Naval Research, by sponsors of the Center for Education and Research in Information Assurance and Security, and by Purdue Discovery Park's e-enterprise Center.

[2] This work was supported in part by Grants CCR-0312760, CCR-0311720, CCR-0225642, and CCR-0098068 from the National Science Foundation.

[3] This work was supported in part by Grants CCF-0311510, IIS-0324846 and CNS-0303577 from the National Science Foundation.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 357–371, 2005.
© Springer-Verlag Berlin Heidelberg 2005

# 1 Introduction

Biometric-based identification starts with a physical measurement for capturing a user's biometric data, followed by the extraction of features from the measurement, and finally a comparison of the feature vector to some previously-stored reference vector. While biometric-based identification holds the promise of providing unforgeable authentication (because the biometric is physically attached to the user), it has a number of practical disadvantages. For example, the storage of reference vectors presents a serious privacy concern, since they usually contain sensitive information that many would prefer to keep private. Even from a security standpoint, biometric information must be stored and transmitted electronically, and, as the old adage goes, a user only gets nine chances to change her fingerprint password (and only one chance to change a retinal password). Thus, we would like to protect the privacy of biometric reference vectors.

One of the major difficulties in biometric information is that, even when it comes from the same individual, it is variable from one measurement to the next. This means that standard encryption of the reference vector is not sufficient to achieve the desired properties. For, even when the reference vector is stored in encrypted form, it appears as though the comparison step (comparing a recently-read biometric image to the reference vector) needs to be done in the clear. That is, standard techniques of comparing one-way hashes (or encryptions) of a stored password and an entered password cannot be used in the context of biometric authentication, as two very similar readings will produce very different hash (or encrypted) values. Unfortunately, this cleartext comparison of biometric data exposes sensitive information to capture by an adversary who obtains one of the two in-the-clear comparands, e.g., through spy-ware at the client or at the server. Moreover, in addition to this comparison-step vulnerability, encrypting the reference vector is obviously not sufficient to protect biometric data from an adversary who learns the decryption key, as could be the case with a dishonest insider at a financial institution. We next review previous work in overcoming these difficulties.

## 1.1 Related Work

There is a vast literature on biometric authentication, and we briefly focus here on the work most relevant to our paper. There are two broad approaches: The one where the comparison is done at the remote server, and the one where the comparison is done at the client end (the portable device where the biometric measurement is done). Most of the recent work has focused on the second category, ever since the landmark paper of Davida et al. [9] proposed that comparisons be done at the client end (although their scheme is also useful in the first case, of remote comparison at the server end). Many other papers (e.g., [3, 4, 13], to mention a few) build on the *wallet with observer* paradigm introduced in Chaum et al. [6] and much-used in the digital cash literature; it implies that there is a tamper-proof device available at the client's end where the comparison is made. The "approximate equality" in biometric comparisons is a challenge faced by any biometric scheme, and many ways have been proposed for overcoming that difficulty while preserving the required security properties. These methods include the use of error-correcting codes [8, 9, 10, 16], fuzzy commitments and fuzzy vaults [7, 15, 16] (for encrypting the private key on the smartcard using fingerprint information), fuzzy extractors [11], secret extraction codes [20] and the use of secure multi-party computation protocols [17]. Some methods carry out the comparisons "in the clear" (after decryption), whereas others manage to avoid it. They all rely, in varying degrees, to one (or more) of the following assumptions: That the portable device is tamper-resistant ("wallet with observer" based papers), that the portable device is powerful enough to carry out relatively expensive cryptographic computations (public-key encryption, homomorphic encryption), and sequences of these for in carrying out complex multi-step protocols. See [5, 12] for a review and a general discussion of the pitfalls and perils of biometric authentication and identification (and how to avoid them), and [18] for a rather skeptical view of biometrics.

In spite of some drawbacks and practicality issues, these schemes have shown the theoretical (and, for some, practical) existence of secure and private biometric authentication.

### 1.2 Motivation for Our Approach

Just like the tiny (and weak) embedded microprocessors that are now pervasive in cars, machinery, and manufacturing plants, so will biometrically-enabled electro-mechanical devices follow a similar path to pervasiveness (this is already starting to happen due to security concerns). Not only inexpensive smartcards, but also small battery-operated sensors, embedded processors, and all kinds of other computationally weak and memory-limited devices may be called upon to carry out biometric authentication. Our work is based on the premise that biometric authentication will eventually be used in a such a pervasive manner, that it will be done on weak clients and servers, ones that can compute cryptographic hashes but not the more expensive cryptographic primitives and protocols; in a battery-powered device, this may be more for energy-consumption reasons than because the processor is slow. This paper explores the use of such inexpensive primitives, and shows that much can be achieved with them.

Our solutions have other desirable characteristics, such as not relying on physical tamper-resistance alone. We believe that relying entirely on tamper-resistance is a case of "putting too many eggs in one basket", just as would be a complete reliance on the assumed security of a remote online server — in either case there is a "single point of failure". See [1, 2] on the hazards of putting too much faith in tamper-resistance. It is desirable that a system's failure requires the compromise of *both* the client and remote server.

### 1.3 Lightweight Biometric Authentication

As stated above, this paper explores the use of lightweight computational primitives and simple protocols for carrying out secure biometric authentication. As in the previous schemes mentioned above, our security requirement is that an attacker should not learn the cleartext biometric data and should not be able to impersonate users by replaying encrypted (or otherwise disguised) data. Indeed, we would like a scheme to be resilient against insider attacks; that is, a (dishonest) insider should be unable to use data stored at the server to impersonate a user (even to the server). We also want our solutions to be simple and practical enough to be easily deployed on weak computational devices, especially at the client, which could be a small smartcard biometric reader. Even so, we do not want to rely on tamper-resistant hardware to store the reference vector at the client. Ideally, we desire solutions that make it infeasible for an attacker to impersonate a user even if the attacker steals the user's client device (e.g., a smartcard) and completely compromises its contents.

Even though the main rationale for this kind of investigation is that it makes possible the use of inexpensive portable units that are computationally weak (due to a slow processor, limited battery life, or both), it is always useful to provide such faster schemes even when powerful units are involved.

### 1.4 Our Contributions

The framework of this paper is one where biometric measurement and feature extraction are done in a unit we henceforth refer to as the *reader*, which we sometimes refer to informally as the "smartcard," although this physical implementation of the reader is just one of many possibilities. It is assumed that the client has physical possession of the reader and, of course, the biometric itself. The alignment and comparison of the resulting measured feature information to the reference feature information is carried out at the *comparison unit*. Both the reference feature information and the comparison unit are assumed to be located at the *server* (at which authentication of the client is desired). Since authentication for financial transactions is a common application of biometric identification, we sometimes refer to the server informally as the "bank."

We present schemes for biometric authentication that can resist several possible attacks. In particular, we allow for the possibility of an attacker gaining access to the communication channel between reader and comparison unit, and/or somehow learning the reference information stored at the comparison unit (reference data is write-protected but could be read by insiders, spyware, etc.). We also allow for the possibility of an attacker stealing the reader from the client and learning the data stored on the reader. Such an attack will, of course, deny authentication service to the client, but it will not allow the attacker to impersonate the user, unless the attacker also obtains a cleartext biometric measurement from the user or the stored reference information at the server. To further resist even these two latter coordinated multiple attacks, the reader could have its data protected with tamper-resistant hardware, but we feel such coordinated multiple attacks (e.g., of simultaneously compromising the reader and the server) should be rare. Even so, tamper-resistant hardware protecting the memory at the reader could allow us to resist even such coordinated attacks.

Given such a rich mix of attacks that we wish to resist, it is desirable that the authentication protocol between reader and comparator not compromise the security or privacy of biometric information. We also require that compromise of the reference data in the comparator does not enable impersonation of the user. These requirements pose a challenging problem because biometrics present the peculiar difficulty that the comparisons are necessarily inexact; they are for approximate equality. We give solutions that satisfy the following properties:

1. The protocols use cryptographic hash computations but not encryption. All the other operations used are inexpensive (no multiplication).
2. Information obtained by an eavesdropper during one round of authentication is useless for the next round, i.e., no replay attacks are possible.
3. User information obtained from the comparison unit by an adversary (e.g., through a corrupt insider or spyware) cannot be used to impersonate that user with that server or in any other context.
4. If a card is stolen and all its contents compromised, then the thief cannot impersonate the user.

Our solutions are based on a *decoupling* of information between the physical biometric, the reader, and the server, so that their communication and storage are protected and private, but the three of them can nevertheless perform robust biometric authentication. Moreover, each authentication in our scheme automatically sets up the parameters for the next authentication to also be performed securely and privately. Our scheme has the property that one smartcard is needed for each bank; this can be viewed as a drawback or as a feature, depending on the application at hand — a real bank is unlikely to trust a universal smartcard and will insist on its own, probably as an added security feature for its existing ATM card infrastructure. On the other hand, a universal card design for our framework of weak computational clients and servers (i.e., a card that works with many banks, as many of the above-mentioned earlier papers achieve) would be interesting and a worthwhile subject of further research. For now, our scheme should be viewed as a biometric supplement to, say, an ATM card's PIN; the PIN problem is trivial because the authentication test is of exact equality — our goal is to handle biometric data with the same efficiency and results as if a PIN had been used. Our scheme is not a competitor for the powerful PKI-like designs in the previous literature, but rather another point on a tradeoff between cost and performance.

We are not aware of any previous work that meets the above-mentioned security requirements using only lightweight primitives and protocols. The alignment stage, which precedes the comparison stage of biometric matching, need not involve any cryptographic computations even when security is a concern (cf. [17], which implemented a secure version of [14]). It is carrying out the (Hamming or more general) distance-computation in a secure manner that involves the expensive cryptographic primitives (in [17], homomorphic encryption). It is therefore on this "bottleneck" of the distance comparison that we henceforth focus, except that we do not restrict ourselves to Hamming distance and also consider other metrics.

# 2 Security Definition for Biometric Authentication

## 2.1 Adversary Model

An adversary is defined by the resources that it has. We now list these resources, and of course an adversary may have any combination of these resources:

1. *Smartcard (SCU and SCC):* An adversary may obtain an uncracked version of the client's smartcard (SCU) or a cracked version of the smartcard (SCC). An adversary with SCU does not see the values on the smartcard, but can probe with various fingerprints. An adversary with SCC is also able to obtain all information on the smartcard. We consider an adversary that cracks the smartcard and then gives it back to the user as outside of our attack model.
2. *Fingerprint (FP):* An adversary may obtain someone's fingerprint, by dusting for the print or by some other more extreme measure.
3. *Eavesdrop (ESD, ECC, and ECU):* An adversary can eavesdrop on various components of the system. These include: i) The server's database (ESD) which contains all information that the server stores about the client, ii) the communication channel (ECC) which has all information sent between the client and server, and iii) the comparison unit (ECU) which has all information from ESD, ECC, and the result of the comparison.
4. *Malicious (MCC):* An adversary may not only be able to eavesdrop on the communication channel but could also change values. We consider adversaries that can change the comparison unit or the server's database as outside of our attack model.

## 2.2 Security Definitions

We look at the confidentiality, integrity, and availability of the system. The confidentiality requirements of the system are that an adversary should not be able to learn information about the fingerprint. The integrity of the system requires that an adversary cannot impersonate a client. The availability of the system requires that an adversary cannot make a client unable to login (i.e., "denial of service"). We now formally define the security requirements for the notions above.

**Confidentiality:**
We present three oracles that are considered secure in our paper, and we prove confidentiality by showing an adversary is equivalent to one of these oracles; in other words if given such an oracle you could emulate the adversary's information. We assume that the oracle has a copy of the ideal fingerprint $\bar{f}$.

1. Suppose the adversary has an oracle $A : \{0,1\}^{|\bar{f}|} \rightarrow \{0,1\}$, where $A(f)$ is true iff $\bar{f}$ and $f$ are close. In other words, the adversary can try an arbitrary number of fingerprints and learn whether or not they are close to each other. We consider a protocol that allows such adversaries to be strongly secure.
2. Suppose the adversary has an oracle $B : \emptyset \rightarrow \{0,1\}^{\log|\bar{f}|}$, where $B()$ returns the distance between several readings of a fingerprint (the actual fingerprints

are unknown to the adversary). In other words, the adversary sees the distance between several readings of a fingerprint. We consider a protocol that allows such adversaries to be strongly secure.

3. Suppose the adversary has an oracle $C : \{0,1\}^{|f|} \rightarrow \{0,1\}^{\log |\bar{f}|}$, where $C(f)$ returns the distance between $\bar{f}$ and $f$. In other words, the adversary can try many fingerprints and will learn the distance from the ideal. Clearly, this adversary is stronger than the above mentioned adversaries. A protocol with such an adversary has acceptable security only in cases where the attack is detectable by the client; we call this weakly secure.

**Integrity:**

To ensure integrity we show that there is a check in place (either by the server or by the client) that an adversary with the specific resources cannot pass without having to invert a one-way function or guess a fingerprint. Of course if the adversary can weakly guess the fingerprint, then we say that the adversary can weakly impersonate the client.

**Availability:**

The types of denial of service attacks that we consider are those where the adversary can prevent the parties from communicating or can make the parties have inconsistent information which would make them unable to successfully authenticate.

### 2.3 Summary of Security Properties for Our Schemes

Before we define the security of our system, we discuss the security (in the terms outlined above) of an "ideal" implementation that uses a trusted oracle. Such a system would require that the client use his fingerprint along with the smartcard and that all communication with the oracle take place through a secure communication channel. The trusted oracle would authenticate the user if and only if both the fingerprint and the smartcard were present. Clearly, we cannot do better than such an implementation.

Table 1 is a summary of an adversary's power with various resources (in our protocol); there are three categories of security: Strong, Weak, and No. Where the first two are defined in the previous section, and "No" means that the system does not protect this resource against this type of adversary. Furthermore, we highlight the entries that are different from an "ideal" system. To avoid cluttering this exposition we do not enumerate all values in the table below, but rather for entries not in the table the adversary has capabilities equal to the maximum over all entries that it dominates.

Thus, in many ways, the smartcard is the lynchpin of the system. While it is desirable to have a protocol that requires both the biometric and the smartcard, having the smartcard be the lynchpin is preferable to having the biometric be the lynchpin. The reason for this is that a biometric can be stolen without the theft being detected, however there is a physical trace when a smartcard is stolen (i.e., it is not there). The only exception to this is when the adversary

**Table 1.** Security of our Protocols

| **Resources** | **Confidentiality** | **Integrity** | **Availability** |
|---|---|---|---|
| FP | No | Strong | Strong |
| SCC and ESD | **No** | **No** | No |
| SCU and FP | No | No | No |
| MCC and ESD | Strong | **No** | No |
| SCU and ESD and MCC | **No** | **No** | No |
| MCC | Strong | Strong | No |
| SCU | Strong | Strong | No |
| SCU and ECU | **Weak** | **Weak** | No |

has malicious control of the communication channel and can eavesdrop on the server's database, and in this case it can impersonate the client (but cannot learn the fingerprint).

## 3 Some False Starts

In this section, we outline some preliminary protocols for biometric authentication that should be viewed as "warmups" for the better solutions given later in the paper. The purpose of giving preliminary protocols first is twofold: (*i*) to demonstrate the difficulty of this problem, and (*ii*) to provide insight into the protocol given later.

Initially, we give preliminary solutions for binary vectors and for the Hamming distance, however these preliminary solutions are extended to arbitrary vectors and other distance functions. The primary question that needs to be addressed is:

> "How does the bank compute the Hamming distance between two binary vectors without learning information about the vectors themselves?"

We assume that the server stores some information about some binary vector $f_0$ (the *reference vector*), and that the client sends the server some information about some other vector $f_1$ (the recently measured *biometric vector*). Furthermore, the server authenticates the client if $dist(f_0, f_1)$, the Hamming distance between $f_0$ and $f_1$, is below some threshold, $\epsilon$. In addition to our security goal of being able to tolerate a number of possible attacks, there are two requirements for such a protocol:

- *Correctness:* the server should correctly compute $dist(f_0, f_1)$.
- *Privacy:* the protocol should reveal nothing about $f_0$ and $f_1$ other than the Hamming distance between the two vectors.

We now give various example protocols that attempt to achieve these goals, but nevertheless fail at some point:

1. Suppose the server stores $f_0$ and the client sends $f_1$ in the clear or encrypted for the server. This amounts to the naive (but common) solution

mentioned above in the introduction. Clearly, this protocol satisfies the correctness property, but it does not satisfy the privacy requirement. In our architecture, this is vulnerable to insider attacks at the server and it reveals actual biometric data to the server.

2. Suppose, instead of storing $f_0$, the server stores $h(f_0||r)$, the result of a cryptographic one-way hash of $f_0$ and a random nonce, $r$. The client would then need to compute $f_1||r$ and apply $h$ to this string, sending the result, $h(f_1||r)$, to the server. This solution improves upon the previous protocol in that it protects the client's privacy. Indeed, the one-way property of the hash function, $h$, makes it computationally infeasible for the server to reconstruct $f_0$ given only $h(f_0||r)$. Unfortunately, this solution does not preserve the correctness of biometric authentication, since cryptographic hashing does not preserve the distance between objects. This scheme will work only for the case when $f_0 = f_1$, which is unlikely given the noise that is inherent in biometric measurements.
3. Suppose, then, that the server instead stores $f_0 \oplus r$ and the client sends $f_1 \oplus r$, for some random vector $r$ known only to the client. This solution satisfies the correctness property for biometric authentication, because $dist(f_0 \oplus r, f_1 \oplus r) = dist(f_0, f_1)$ for the Hamming distance metric. This solution might at first seem to satisfy the privacy requirement, because it hides the number of 0's and 1's in the vectors $f_0$ and $f_1$. However, the server learns the positions where there is a difference between these vectors, which leaks information to the server with each authentication. This leakage is problematic, for after several authentication attempts the server will know statistics about the locations that differ frequently. Depending on the means of how feature vectors are extracted from the biometric, this leakage could reveal identifying characteristics of the client's biometric information. Thus, although it seems to be secure, this solution nonetheless violates the privacy constraint.
4. Suppose, therefore, that the scheme uses a more sophisticated obfuscating technique, requiring the server to store $\Pi(f_0 \oplus r)$, for some random vector $r$ and some fixed random permutation over the indices of biometric vector, $\Pi$, known only to the client. The client can authenticate in this case by sending $\Pi(f_1 \oplus r)$. This solution satisfies the correctness property, because $dist(\Pi(f_0 \oplus r), \Pi(f_1 \oplus r)) = dist(f_0, f_1)$, for the Hamming distance metric. Moreover, by using a random permutation, the server does not learn the places in $f_0$ and $f_1$ where differences occur (just the places where the permuted vectors differ). Thus, for a single authentication round the server learns only the Hamming distance between $f_0$ and $f_1$. Unfortunately, this scheme nevertheless still leaks information with each authentication, since the server learns the places in the permuted vectors where they differ. Over time, because the same $\Pi$ is used each time, this could allow the server to determine identifying information in the biometric.

This final scheme is clearly the most promising of the above false starts, in that it satisfies the correctness and privacy goals for a single authentication round. Our scheme for secure biometric authentication, in fact, is based on taking this final false start as a starting point. The main challenge in making this scheme secure even for an arbitrarily long sequence of authentications is that we need a secure way of getting the server and client to agree on future permutations and random nonces (without again violating the correctness and privacy constraints).

# 4 Our Schemes for Secure Biometric Authentication

In this section, we give our protocols for secure biometric authentication. We begin with a protocol for the case of Boolean vectors where the relevant distance between two such vectors is the Hamming distance. We later extend this to vectors of arbitrary numbers and distance metrics that depend on differences between the corresponding components (this is a broad class that contains the Euclidean distance $L_2$, as well as $L_1$). We use $H(\cdot)$ to denote a keyed hash, where the key is a secret known to the client and server but not to others. An additional challenge in using such a function is that we now must prevent someone who accidentally (or maliciously) learns the client information at the server's end from using that information to impersonate the client to the server. Likewise, we must maintain the property that someone who learns the client's information on the reader should not be able to use this information (and possibly previously eavesdropped sessions) to impersonate the client.

## 4.1 Boolean Biometric Vectors

The server (in the database and the comparison unit) and the client (in the smartcard) store a small collection of values, which are recomputed after each round. Also, there are $q$ copies of this information at the server and on the card, where $q$ is the number of fingerprint mismatches before a person must re-register with the server. In what follows, $f_i$ and $f_{i+1}$ are Boolean vectors derived from biometric readings at the client's end, $\Pi_i$ and $\Pi_{i+1}$ denote random permutations generated by and known to the client but not the server, and $r_i, r_{i+1}, s_i, s_{i+1}, s_{i+2}$ are random Boolean vectors generated by the client, some of which may end up being revealed to the server.

Before a round, the server and client store the following values:

- The server has: $s_i \oplus \Pi_i(f_i \oplus r_i)$, $H(s_i)$, $H(s_i, H(s_{i+1}))$.
- The client has: $\Pi_i$, $r_i$, $s_i$, $s_{i+1}$.

A round of authentication must not only convince the server that the client has a vector $f_{i+1}$ that is "close" (in the Hamming distance sense) to $f_i$, but must also refresh the above information. A round consists of the following steps:

1. The client uses the smartcard to read a new biometric $f_{i+1}$ and to generate random Boolean vectors $r_{i+1}$ and $s_{i+2}$ and a random permutation $\Pi_{i+1}$.
2. The smartcard connects to the terminal and sends to the server the following values: $\Pi_i(f_{i+1} \oplus r_i)$, $s_i$, and "transaction information" $T$ that consists of

a nonce as well as some other information related to this particular access request (e.g., date and time, etc).

3. The server computes the hash of the just-received $s_i$ and checks that it is equal to the previously-stored $H(s_i)$. If this check does not match it aborts the protocol. If it does match, then the server computes the XOR of $s_i$ with the previously-stored $s_i \oplus \Pi_i(f_i \oplus r_i)$ and obtains $\Pi_i(f_i \oplus r_i)$. Then the server computes the Hamming distance between the just-computed $\Pi_i(f_i \oplus r_i)$ and the received $\Pi_i(f_{i+1} \oplus r_i)$.
   - If the outcome is a match, then the server sends $H(T)$ to the client.
   - If it is not a match, then the server aborts but throws away this set of information in order to prevent replay attacks; if the server does not have any more authentication parts, then it locks the account and requires the client to re-register.
4. The smartcard checks that the value sent back from the server matches $H(T)$ (recall that $H$ is a keyed hash). If the message does not match, the smartcard sends an error to the server. Otherwise, the smartcard sends the server the following information: $s_{i+1} \oplus \Pi_{i+1}(f_{i+1} \oplus r_{i+1})$, $H(s_{i+1}, H(s_{i+2}))$, and $H(s_{i+1})$. It also wipes from its memory the reading of fingerprint $f_{i+1}$ and of previous random values $r_i$ and $s_i$, so it is left with $\Pi_{i+1}$, $r_{i+1}$, $s_{i+1}$, $s_{i+2}$.
5. When the server receives this message it verifies that $H(s_i, H(s_{i+1}))$ matches the previous value that it has for this quantity and then updates its stored values to: $s_{i+1} \oplus \Pi_{i+1}(f_{i+1} \oplus r_{i+1})$, $H(s_{i+1}, H(s_{i+2}))$, and $H(s_{i+1})$.

### 4.2 Arbitrary Biometric Vectors

Suppose the biometric vectors $f_i$ and $f_{i+1}$ now contain arbitrary (rather than binary) values, and the proximity decision is based on a distance function that depends on $|f_i - f_{i+1}|$.

Modify the description of the Boolean protocol as follows:

- Each of $r_i, r_{i+1}$ is now a vector of arbitrary numerical values rather than Boolean values (but $s_i, s_{i+1}, s_{i+2}$ are still Boolean).
- Every $f_j \oplus x$ gets replaced in the protocol's description by $f_j + x$, e.g., $f_i \oplus r_i$ becomes $f_i + r_i$. (The length of $s_i$ must of course now be the same as the number of bits in the binary representation of $f_i + r_i$, but we refrain from belaboring this straightforward issue.)

The above requires communication $O((\log \Sigma)n)$, where $\Sigma$ is the size of the alphabet and $n$ is the number of items. This reveals slightly more than the distance, in that it reveals the component-wise differences. This information leakage is minimal especially since the values are permuted. In the case where the function is $\sum_{i=1}^{n} |f_i - f_{i+1}|$, we could use a unary encoding for each value and reduce it to a Hamming distance computation, for which the protocols of the previous section can then be used. This does not reveal the component-wise differences, but it requires $O(\Sigma n)$ communication.

## 5 Security of the Protocols

In this section, we define the information and abilities of the adversaries, and then prove the confidentiality, integrity and availability constraints.

**Resources:**

The following table summarizes the information of various adversaries. Generally, an adversary with multiple resources gets all of the information of each resource. There are cases where this is not the case, e.g., consider an adversary with SCU and ECC; the adversary could not see readings of the client's fingerprint, because the client no longer has the smartcard to attempt a login.

**Table 2.** Information of Various Adversaries

| Adversary | Information |
|---|---|
| FP | f |
| SCU | Ability to probe small number of fingerprints |
| SCC | SCU and $r_i, s_i, \Pi_i, k$ |
| ESD | $k$ and several sets of $H(s_i), H(s_i, H(s_{i+1})), s_i \oplus \Pi_i(f \oplus r_i)$ |
| ECC | Several sets of $s_i, \Pi_i(f \oplus r_i), H(s_{i+1}), H(s_{i+2})$ |
| ECU | ESD and ECC and distances of several readings |
| MCC | ECC and can change values |

**Confidentiality:**

Before we prove the confidentiality requirements we need the following lemma (the proof is omitted sue to space constraints):

**Lemma 1.** *The pair of values* $(\Pi(f \oplus r))$ *and* $(\Pi(f' \oplus r))$ *reveals nothing other than the distance between each pair of vectors.*

**Theorem 1.** *The only cases where an adversary learns the fingerprint are in: i) FP, ii) SCC and ESD, iii) SCU and ESD and MCC, and iv) any superset of these cases. In the case of SCU and ECU, the adversary weakly learns the fingerprint.*

**Proof:** First when the adversary has the fingerprint the case is clearly true. Suppose that the adversary has ECU and MCC, the adversary sees several pairs of $\Pi(f \oplus r)$ and $\Pi(f' \oplus r)$ and by Lemma 1, this only reveals a set of distances, which is equivalent to oracle B and thus is secure. Thus any attack must involve an adversary with the smartcard in some form. Clearly, any adversary with the smartcard cannot eavesdrop on communication when the client is logging into the system.

Suppose that the adversary has SCC and MCC. The adversary has no information about the fingerprint in any of its information, since nothing is on the smartcard and a client cannot login without the smartcard, and thus the fingerprint is protected. However, if the adversary has SCC and ESD, they can trivially learn the fingerprint from knowing $\Pi_i, r_i, s_i, s_i \oplus \Pi_i(f \oplus r_i)$.

Any adversary with SCU can only probe various fingerprints, as no other information is given. Suppose that the adversary has SCU and ECU. In this case the adversary can probe various fingerprints and can learn the distance, which is equivalent to oracle $C$ and thus is weakly secure. Consider an adversary with SCU and ESD. In this case they can probe using the SCU, but this is just oracle $A$. If the adversary has SCU and MCC, they can learn $s$, $\Pi$, and $r$ values by stopping the traffic and trying various fingerprints, however they cannot use this to glean the fingerprint as the client cannot login once the smartcard is stolen. Finally, if the adversary has SCU and MCC and ESD, then they can learn the values and then learn the fingerprint. □

**Integrity and Availability:**

**Theorem 2.** *The only cases where an adversary can impersonate a client are in: i) SCU+FP, ii) SCC and ESD, iii) MCC and ESD, and iv) any superset of these cases. In the case of SCU and ECU, the adversary weakly impersonate the client. The only cases where an adversary can attack the availability of the client are in: i) SCU, ii) MCC, and iii) any superset of these cases.*

**Proof:** The proof of this claim will be in the full version of the paper.

## 6 Storage-Computation Tradeoff

In this section, we introduce a protocol that allows $q$ fingerprint mismatches before requiring the client to re-register with the server, with only $O(1)$ storage, but that requires $O(q)$ hashes to authenticate. This utilizes similar ideas as SKEY [19]; in what follows $H^j(x)$ denotes the value of $x$ hashed $j$ times. We do not prove the security of this system due to space limitations. After the setup the following is the state of the system:

- Server has: $\bigoplus_{j=0}^{q-1} H^j(s_i) \oplus \Pi_i(f_i \oplus r_i)$, $H^q(s_i)$, and $H(H^q(s_i), H^q(s_{i+1}))$.
- Client has: $\Pi_i$, $r_i$, $s_i$, and $s_{i+1}$.

After $t$ fingerprint mismatches the server has: $\bigoplus_{j=0}^{q-t-1} H^j(s_i) \oplus \Pi_i(f_i \oplus r_i)$, $H^{q-t}(s_i)$, and $H(H^q(s_i), H^q(s_{i+1}))$.

The authentication and information-updating round is as follows for the $t$th attempt to authenticate the client:

1. The client uses the smartcard to read a new biometric $f_{i+1}$ and to generate random Boolean vectors $r_{i+1}$ and $s_{i+2}$ and a random permutation $\Pi_{i+1}$.
2. The smartcard connects to the terminal and sends to the server the following values: $\bigoplus_{j=0}^{q-t-1} H^j(s_i) \oplus \Pi_i(f_{i+1} \oplus r_i)$ and $H^{q-t}(s_i)$.
3. The server computes the hash of the just-received $H^{q-t}(s_i)$ and checks that it is equal to the previously-stored $H^{q-t+1}(s_i)$. If this check does not match it aborts the protocol. If it does match, then the server computes the XOR of $H^{q-t}(s_i)$ with the previously-stored $\bigoplus_{j=0}^{q-t} H^j(s_i) \oplus \Pi_i(f_i \oplus r_i)$ and obtains $\bigoplus_{j=0}^{q-t-1} H^j(s_i) \oplus \Pi_i(f_i \oplus r_i)$. It then computes the Hamming distance

between the just-computed $\bigoplus_{j=0}^{q-t-1} H^j(s_i) \oplus \Pi_i(f_i \oplus r_i)$ and the received $\bigoplus_{j=0}^{q-t-1} H^j(s_i) \oplus \Pi_i(f_{i+1} \oplus r_i)$.

   – If the outcome is a match, then the server sends $H(T)$ (recall that $H$ is a keyed hash) to the client.
   – If it is not a match, then the server updates its values to the following: $\bigoplus_{j=0}^{q-t-1} H^j(s_i) \oplus \Pi_i(f_i \oplus r_i)$, $H^{q-t}(s_i)$, and $H(H^q(s_i), H^q(s_{i+1}))$. If $t = q$, then the server locks the account and requires the client to re-register.

4. If the smartcard checks that the value sent back from the server matches $H(T)$, then the smartcard sends the server the following information: $\bigoplus_{j=0}^{q-1} H^j(H^{q-t}(s_{i+1})) \oplus \Pi_{i+1}(f_{i+1} \oplus r_{i+1})$, as well as $H^q(s_{i+1})$, and also $H(H^q(s_{i+1}), H^q(s_{i+2}))$. If it does not match, then it sends an error to the server and aborts. In either case, it wipes from its memory the reading of fingerprint $f_{i+1}$ and those previously stored values that are no longer relevant.
5. When the server receives this message it verifies that $H(H^q(s_i), H^q(s_{i+1}))$ matches the previous value that it has for this quantity and then updates its stored values to: $\bigoplus_{j=0}^{q-1} H^j(H^{q-t}(s_{i+1})) \oplus \Pi_{i+1}(f_{i+1} \oplus r_{i+1})$, $H^q(s_{i+1})$, and $H(H^q(s_{i+1}), H^q(s_{i+2}))$

## 7 Conclusions and Future Work

In this paper, a lightweight scheme was introduced for biometric authentication that could be used by weak computational devices. Unlike other protocols for this problem, our solution does not require complex cryptographic primitives, but instead relies on cryptographic hashes. Our protocols are secure in that the client's fingerprint is protected, it is "hard" to impersonate a client to the comparison unit, and adversaries with malicious access to the communication channel cannot steal a client's identity (i.e., be able to impersonate the client to the comparison unit after the transaction). To be more precise, an adversary would need the smartcard and either the fingerprint or the server's database to impersonate the client. One problem with our protocol is that for every successful authentication, the database must update its entry to a new value (to prevent replay attacks), and thus we present the following open problem: is it possible for the server to have a static database and have a secure authentication mechanism that requires only cryptographic hash functions?

## References

1. R. Anderson and M. Kuhn. Low cost attacks on tamper resistant devices. In *International Workshop on Security Protocols*, pages 125–136, 1997.
2. R. J. Anderson and M. Kuhn. Tamper resistance - a cautionary note. In *Proceedings of the 2nd USENIX Workshop on Electronic Commerce*, pages 1–11, 1996.
3. G. Bleumer. Biometric yet privacy protecting person authentication. In *Proceedings of 1998 Information Hiding Workshop (IHW 98)*, pages 101–112. Springer-Verlag, 1998.

4. G. Bleumer. Offine personal credentials. Technical Report TR 98.4.1, AT&T, 1998.
5. R. M. Bolle, J. H. Connell, and N. K. Ratha. Biometric perils and patches. *Pattern Recognition*, 35(12):2727–2738, 2002.
6. D. Chaum and T. P. Pedersen. Wallet databases with observers. In *Crypto '92, LNCS 740*, pages 89–105. Springer Verlag, 1993.
7. T. C. Clancy, N. Kiyavashr, and D. Lin. Secure smartcard-based fingerprint authentication. In *Proceedings of the 2003 ACM Workshop on Biometrics Methods and Applications*, pages 45–52, 2003.
8. G. Davida and Y. Frankel. Perfectly secure authorization and passive identification for an error tolerant biometric system. In *Proceedings of 7th Conference on Cryptography and Coding, LNCS 1746*, pages 104–113, 1999.
9. G. I. Davida, Y. Frankel, and B. J. Matt. On enabling secure applications through off-line biometric identification. In *Proceedings of 1998 IEEE Symposium on Security and Privacy*, pages 148–157, May 1998.
10. G. I. Davida, Y. Frankel, and B. J. Matt. On the relation of error correction and cryptography to an off-line biometric based identification scheme. In *Proceedings of WCC99, Workshop on Coding and Cryptography*, 1999.
11. Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *EUROCRYPT 2004, LNCS 3027*, pages 523–540. Springer Verlag, 2004.
12. G. Hachez, F. Koeune, and J.-J. Quisquater. Biometrics, access control, smart cards: A not so simple combination. In *Proc. of the Fourth Working Conference on Smart Card Research and Advanced Applications (CARDIS 2000)*, pages 273–288. Kluwer Academic Publishers, September 2000.
13. R. Impagliazzo and S. M. More. Anonymous credentials with biometrically-enforced non-transferability. In *Proceedings of the Second ACM Workshop on Privacy in the Electronic Society (WPES '03)*, pages 60–71, October 2003.
14. A. Jain, L. Hong, and R. Bolle. On-line fingerprint verification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):302–314, 1997.
15. A. Juels and M. Sudan. A fuzzy vault scheme. In *Proceedings of the 2002 IEEE International Symposium on Information Theory*, pages 408–413, 2002.
16. A. Juels and M. Wattenberg. A fuzzy commitment scheme. In *Proceedings of the 6th ACM conference on Computer and communications security*, pages 28–36. ACM Press, 1999.
17. F. Kerschbaum, M. J. Atallah, D. Mraihi, and J. R. Rice. Private fingerprint verification without local storage. In *International Conference on Biometric Authentication (ICBA)*, July 2004.
18. B. Schneier. Biometrics: Truths and fictions
`http://www.schneier.com/crypto-gram-9808.html#biometrics`.
19. B. Schneier. *Applied cryptography: protocols, algorithms, and source code in C.* John Wiley & Sons, Inc., 2nd edition, 1995.
20. P. Tuyls and J. Goseling. Capacity and examples of template-protecting biometric authentication systems. In *ECCV Workshop on Biometric Authentication*, volume 3087 of *Lecture Notes in Computer Science*, pages 158 – 170, 2004.

# Panel Summary: Incentives, Markets and Information Security

Allan Friedman

PhD Mailboxes, Kennedy School of Government, Harvard University,
79 JFK St, Cambridge MA 02138, USA
Allan_friedman@ksgphd.harvard.edu

Economics and information security should be naturally related: the former deals with the value and distribution of scarce resources, while the latter focuses on protecting and controlling valued resources. Indeed, the observation that information security should be informed by economic theory is not new. Anderson [1] and others have explicitly highlighted the relationship, which can be seen as a natural progression from the economics of crime literature that dates back to the 1960s [2].

The discipline of economics has a set of established methods for analyzing incentives, useful for mapping questions of possession and valuation of resources into tractable, analytic frameworks. The field also has a rich tradition of "mechanism design" or how systems can be structured such that self-interested agents can be induced to behave in socially optimal fashions. Economics also offers a framework for analyzing difficult trade-offs by focusing on the underlying value. Rather than looking at the ability to prevent any sort of system subversion, benefit-cost comparison tools such as return-on-investment and externality identification allow us to examine the overall impact of action or inaction.

This panel was assembled to present a range of issues in information security that can benefit from the application of economic analysis and discuss how engineering and economics are both needed to address a wide range of pressing policy issues. It is by no means a complete description of this nascent interdisciplinary field. Further information can be found in [6] or [3].

**Panel Presentations.** Bazelel Gavish presented a straightforward economic analysis of a commonly discussed information security issue: spam. Gavish argues the problem stems from the low marginal cost to send messages in a digital environment, and proposes fee-based system that gives a credit to each recipient, claimable from the sender. While the general idea has been discussed before [5], this approach involved both end parties and the service providers. Gavish advocated a dynamic pricing scheme, and highlighted important areas of research for implementation.

Paul Syverson shifted the focus from mechanisms to institutions, arguing the "identity theft is about neither identity nor theft." Syverson highlighted flaws in the current state of consumer authentication, where information that has a very high value in specific contexts (a social security number can open a line of credit or obtain a new password) is undervalued by some actors, leading to arbitrage and fraud. This also introduced the concept of a security externality, where poor protection or overuse of identifying and authenticating information can raise fraud rates for other parties.

A.S. Patrick and M. Yung (Eds.): FC 2005, LNCS 3570, pp. 372 – 373, 2005.
© Springer-Verlag Berlin Heidelberg 2005

Sven Dietrich demonstrated that a single security issue like distributed denial of service (DDOS) attacks presents the opportunity for multiple levels of analysis that stem from unique features of information systems. The nature of the attack stems from the decentralized environment, where the coordination costs of a bot-net are less than the damage inflicted on the target. Networks of subverted machines also raise the question of who should bear responsibility for the damage caused, since the software manufacturer, the machine owner and local ISP could all have theoretically prevented the machine from causing damage. Dietrich even explained the networks of subverted machines were traded in illicit marketplaces, raising questions of trust and quality guarantees. While no single approach can solve the problem of DDOS attacks, each layer of analysis opens an opportunity to raise the costs, reduce the damages and mitigate harms of this critical issue.

Finally, Richard Clayton took a step back, acknowledging the importance of economics in the field of security, but tempering this enthusiasm with several observations. Using the example of email payments, he illustrated that proposed economic solutions might fall flat from simple economic or technical realities. Further- more, economics is a nice tool, but good numbers are needed to judge efficacy. It is one thing to build internally consistent models but to further extend the field, these models should be consistent with empirical data. Clayton summed up by urging people to learn more about economics, but suggesting that it was "perhaps not yet time to change depart- ments."

**Future Directions.** There is little doubt that information security can be improved by better integrating economic and security theory. Acquiring better data is critical to applying theory in a policy context for both private firms and public decision-makers. Some problems may be small and tractable enough to resolve with well-informed models. The common debate about public disclosure of other's security flaws, for example, has been the focus of much attention, and it is conceivable that a consensus might be reached. Economics also serves as a useful lever with which to break apart major security issues into segments without competing incentives. Similar to Clark et al's "tussle space" theory [4], this would allow computer scientists to better address open problems without worrying about unnecessary conflicts of motivation. Finally, all involved must acknowledge that economics itself isn't the magic bullet: information security solutions, especially those at the user level, should incorporate critical findings in behavioral psychology and usability.

## References

1. Anderson, R.: Why Information Security is Hard - An Economic Perspective. Proceedings In Proc. 17th Annual Computer Security Applications Conference (2001)
2. Becker, G. S. Crime and Punishment: An Economic Approach. The Journal of Political Economy. 76:8 (1968) 169:217
3. Camp, L.J., Lewis, S.(ed.): Economics of Information Security. Kluwer Academic Publishers, Boston MA (2004)
4. D. D. Clark, J. Wroclawski, K. Sollins, R. Braden, Tussle in Cyberspace: Defining Tomorrow's Internet, Proc. ACM SIGCOMM (2002)
5. Cranor, L.F., LaMacchia, B.A.: Spam! Communications of the ACM. 41:8 (1998)
6. Workshop on the Economics of Information Security: Past Workshops http://infosecon.net/workshop/past.php (2005)

# Author Index