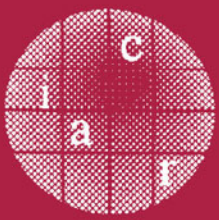
Feng Bao
Robert Deng
Jianying Zhou (Eds.)

LNCS 2947

Public Key Cryptography – PKC 2004

7th International Workshop
on Theory and Practice in Public Key Cryptography
Singapore, March 2004, Proceedings



Springer

Lecture Notes in Computer Science 2947

Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

Springer
Berlin
Heidelberg
New York
Hong Kong
London
Milan
Paris
Tokyo

Feng Bao Robert Deng
Jianying Zhou (Eds.)

Public Key Cryptography – PKC 2004

7th International Workshop
on Theory and Practice in Public Key Cryptography
Singapore, March 1-4, 2004
Proceedings

Springer

eBook ISBN: 3-540-24632-0
Print ISBN: 3-540-21018-0

Visit Springer's eBookstore at: http://ebooks.kluweronline.com
and the Springer Global Website Online at: http://www.springeronline.com

Preface

PKC 2004 was the 7th International Workshop on Practice and Theory in Public Key Cryptography and was sponsored by IACR, the International Association for Cryptologic Research (www.iacr.org). This year the workshop was organized in cooperation with the Institute for Infocomm Research (I^2R), Singapore.

There were 106 paper submissions from 19 countries to PKC 2004. That is the highest submission number in PKC history. Due to the large number of submissions and the high quality of the submitted papers, not all the papers that contained new ideas were accepted. Of the 106 submissions, 32 were selected for the proceedings. Each paper was sent to at least 3 members of the Program Committee for comments. The revised versions of the accepted papers were not checked for correctness of their scientific aspects and the authors bear the full responsibility for the contents of their papers. Some authors will write final versions of their papers for publication in refereed journals.

I am very grateful to the members of the Program Committee for their hard work in the difficult task of selecting fewer than 1 in 3 of the submitted papers, as well as the following external referees who helped the Program Committee: Nuttapong Attrapadung, Roberto Maria Avanzi, Gildas Avoine, Joonsang Baek, Qingjun Cai, Jae Choon Cha, Chien-Ning Chen, Liqun Chen, Xiaofeng Chen, Koji Chida, Nicolas T. Courtois, Yang Cui, Jean-François Dhem, Louis Goubin, Louis Granboulan, Rob Granger, Jens Groth, Yumiko Hanaoka, Darrel Hankerson, Chao-Chih Hsu, Tetsutaro Kobayashi, Yuichi Komano, Hidenori Kuwakado, Tanja Lange, Peter Leadbitter, Byoungcheon Lee, Chun-Ko Lee, Henry C.J. Lee, John Malone Lee, Yong Li, Benoît Libert, Hsi-Chung Lin, Yi Lu, Jean Monnerat, Anderson C.A. Nascimento, C. Andrew Neff, Akira Otsuka, Daniel Page, Kenny Paterson, Kun Peng, David Pointcheval, Taiichi Saitoh, Junji Shikata, Igor Shparlinksi, Martijn Stam, Ron Steinfeld, Koutarou Suzuki, Shigenori Uchiyama, Frederik Vercauteren, Guilin Wang, Benne de Weger, Guohua Xiong, Go Yamamoto, Shoko Yonezawa, Rui Zhang, and Huafei Zhu. (I apologize for any possible omission.) The Program Committee appreciates their efforts.

Thanks to Patricia Loh for the secretarial work and to Ying Qiu for maintaining the WWW page of the conference. Finally, I would like to thank everyone who submitted to PKC 2004, and IACR for its sponsorship.

December 2003 Feng Bao

This page intentionally left blank

PKC 2004

7th International Workshop on Practice and Theory in Public Key Cryptography

Singapore
March 1–4, 2004

Sponsored and organized by

Institute for Infocomm Research, Singapore

In co-operation with

International Association for Cryptologic Research

General Chair

Robert Deng ..... Institute for Infocomm Research, Singapore

Program Chair

Feng Bao ..... Institute for Infocomm Research, Singapore

Publication Chair

Jianying Zhou ..... Institute for Infocomm Research, Singapore

Program Committee

Masayuki Abe ..... NTT Laboratories, Japan
Feng Bao ..... Institute for Infocomm Research, Singapore
Colin Boyd ..... Queensland University of Technology, Australia
Robert Deng ..... Institute for Infocomm Research, Singapore
Yvo Desmedt ..... Florida State University, USA
Marc Fischlin ..... Fraunhofer Institute for Secure Telecooperation, Germany
Eiichiro Fujisaki ..... NTT Laboratories, Japan
Goichiro Hanaoka ..... University of Tokyo, Japan
Marc Joye ..... Gemplus, France
Kwangjo Kim ..... Information and Communications University, Korea
Arjen Lenstra ..... Citibank, USA and Tech. Uni. Eindhoven, Netherlands
Wenbo Mao ..... Hewlett-Packard Labs, UK
Alfred Menezes ..... University of Waterloo, Canada
Phong Nguyen ..... CNRS/École Normale Supérieure, France
Dingyi Pei ..... Chinese Academy of Sciences, China
Claus Schnorr ..... Frankfurt University, Germany

Nigel Smart ..... University of Bristol, UK
Renji Tao ..... Chinese Academy of Sciences, China
Serge Vaudenay ..... Swiss Federal Institute of Technology, Switzerland
Sung-Ming Yen ..... National Central University, Taiwan
Moti Yung ..... Columbia University, USA
Yuliang Zheng ..... University of North Carolina, USA
Jianying Zhou ..... Institute for Infocomm Research, Singapore

Table of Contents

This page intentionally left blank

A Generalized Wiener Attack on RSA

Johannes Blömer and Alexander May

Faculty of Computer Science, Electrical Engineering and Mathematics
University of Paderborn
33102 Paderborn, Germany
{bloemer,alexx}@uni-paderborn.de

Abstract. We present an extension of Wiener's attack on small RSA secret decryption exponents [10]. Wiener showed that every RSA public key tuple (N, e) with $e \in \mathbb{Z}^*_{\phi(N)}$ that satisfies $ed - 1 = 0 \bmod \phi(N)$ for some $d < \frac{1}{3}N^{\frac{1}{4}}$ yields the factorization of $N = pq$. Our new method finds p and q in polynomial time for every (N, e) satisfying $ex + y = 0 \bmod \phi(N)$ with

$$x < \frac{1}{3}N^{\frac{1}{4}} \quad \text{and} \quad |y| = \mathcal{O}(N^{-\frac{3}{4}}ex).$$

In other words, the generalization works for all secret keys $d = -xy^{-1}$, where x, y are suitably small. We show that the number of these weak keys is at least $N^{\frac{3}{4}-\epsilon}$ and that the number increases with decreasing prime difference $p - q$. As an application of our new attack, we present the cryptanalysis of an RSA-type scheme presented by Yen, Kim, Lim and Moon [11,12]. Our results point out again the warning for crypto-designers to be careful when using the RSA key generation process with special parameters.

Keywords: RSA, weak keys, Wiener attack, continued fractions

1 Introduction

Let $N = pq$ be an RSA-modulus, where p and q are primes of equal bit-size (wlog $p > q$). Let e be the public exponent and d be the secret exponent satisfying $ed = 1 \bmod \phi(N)$, where $\phi(N)$ is the Euler totient function. We denote by $\mathbb{Z}^*_{\phi(N)}$ the multiplicative group of invertible integers modulo $\phi(N)$. An RSA public key is a tuple $(N, e) \in \mathbb{Z} \times \mathbb{Z}^*_{\phi(N)}$.

In order to study the security of RSA, many people focus on the difficulty to factor the modulus N without taking into account additional information that may be encoded in the public exponent e. Hence, it is tempting for crypto-designers to construct RSA-type schemes with special public exponents that yield a good performance in encryption/decryption. For example, one might be tempted to use small decryption exponents d in order to speed up the decryption process. Another fast RSA-variant that makes use of special RSA-keys was proposed by Yen, Kim, Lim and Moon [11,12] in 2001. This YKLM-scheme is

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 1–13, 2004.

designed to counteract the fault-based attack on CRT-RSA of Boneh, DeMillo and Lipton [2].

In 1990, Wiener [10] observed that information encoded in the public exponent e might help to factor N. More precisely, he showed that every public exponent $e \in \mathbb{Z}^*_{\phi(N)}$ that corresponds to a secret exponent d with $d \leq \frac{1}{3}N^{\frac{1}{4}}$ yields the factorization of the modulus in time polynomial in $\log(N)$. In 1999, Boneh and Durfee [3] used Coppersmith's method for finding small roots of modular polynomial equations [4] to improve the bound to $d \leq N^{0.292}$.

Although the YKLM-scheme uses a special key generation algorithm in order to provide good performance, the secret keys d are not chosen to be small. Therefore, the Wiener attack as well as the Boneh-Durfee attack cannot directly be applied to this RSA-variant. However, in this work we present an extension of Wiener's approach that leeds to a much larger class of secret keys d which are insecure. Furthermore, we show that the keys which are generated in the YKLM-scheme belong to this larger class, for all reasonable parameter choices of the scheme. As a result, we obtain that the public keys (N, e) in the YKLM-scheme yield the factorization of N in polynomial time.

Let us put the cryptanalytic approaches above into a more general framework by defining the notion of *weak keys:* The results so far show that there are classes of public keys (N, e) where every element in the class yields the factorization of N. One may view the auxiliary input e as a hint how to factor N: Without having e we assume that factoring N is hard, but with the help of e it becomes feasible. In the case of the Wiener attack the class consists of all public key tuples (N, e) where $ed - 1 = 0 \bmod \phi(N)$ with $d < \frac{1}{3}N^{\frac{1}{4}}$.

We call such a class *weak* and the elements (N, e) of the weak class are called *weak keys.* To be more precisely: We define the size of a class of public key tuples by the number of elements (N, e) in the class for every fixed N. Let C be a class of public key tuples (N, e), then

$$\text{size}_C(N) = |\{e \in \mathbb{Z}^*_{\phi(N)} \mid (N, e) \in C\}|.$$

C is called *weak* if

1. The size of C is polynomial in N, i.e. $\text{size}_C(N) = \Omega(N^\gamma)$ for some $\gamma > 0$.
2. There exists a probabilistic algorithm A that on every input $(N, e) \in C$ outputs the factorization of N in time polynomial in $\log(N)$.

Note that the size of a weak class is a function in N which denotes the number of elements that can be factored by the corresponding algorithm A. For example, the size of the class in the Wiener attack is at least $N^{\frac{1}{4}-\epsilon}$. Here the ϵ-term comes from the fact that only those d with $\gcd(d, \phi(N)) = 1$ define legitimate RSA-keys.

Let us give another (trivial) example of a weak class of public keys. Every tuple (N, e) with $e = kq$, $1 < k < p$ is a weak key, since the computation $\gcd(N, e) = q$ yields the factorization. These are $p > N^{\frac{1}{2}}$ many weak keys. Howgrave-Graham [6] observed that even the knowledge of $e = kq + r$ for some

unknown $r \leq N^{\frac{1}{4}}$ suffices to find the factorization of N. This implies the existence of a weak class with size $N^{\frac{3}{4}}$.

We think that it is a very natural question to study how many of the possible choices of the public keys are indeed weak keys that should not be used in the design of crypto-systems. For the Wiener attack and the Boneh-Durfee attack it is easy for a crypto-designer to see that a key is weak by inspecting the most significant bits of d. For the extension of Wiener's attack that we describe in this paper the weakness of the keys is not obvious. One can understand our new result as a warning for crypto-designers to be careful when using keys with a special structure.

There is also an imminent danger from weak keys in the case of untrusted servers that create public/secret key pairs: Crépeau and Slakmon [5] showed how to use weak keys in order to construct malicious RSA systems by encoding information into the public exponent e. Our new class of weak keys is well-suited for the use in such systems and leads to a large variety of new malicious keys.

In order to describe our new attack, let us first consider the normal RSA-case, where $p - q = \Omega(\sqrt{N})$. Note that for randomly chosen primes of the same bitsize, the probability that p, q agree in the c most significant bits is roughly $2^{-(c-1)}$. Hence, we have $p - q = \Omega(\sqrt{N})$ with overwhelming probability.

For the case $p - q = \Omega(\sqrt{N})$, we introduce a variant of Wiener's attack that works for all public keys (N, e) where $ex + y = k\phi(N)$, $k \in \mathbb{N}$ with

$$0 < x \leq \frac{1}{3}N^{\frac{1}{4}} \quad \text{and} \quad |y| = \mathcal{O}(N^{-\frac{3}{4}}ex).$$

Notice that our bounds exclude trivial solutions where $ex + y = 0$, since $|y| < ex$.

The new method works as follows: As in Wiener's approach, we use the continued fraction algorithm to recover the unknown values x and k. Afterwards, we show that a factorization method due to Coppersmith [4] can be applied: Given half of the most significant bits of p, one can find the factorization of N.

Let us compare the new result to Wiener's attack. Our weak keys have the structure that $e^{-1} = d = -\frac{x}{y} \bmod \phi(N)$, i.e. Wiener's algorithm is the special case where $x = d$ and $y = -1$. One should observe that for x of size roughly $N^{\frac{1}{4}}$ as in Wiener's attack, the parameter e must be of size at least $N^{\frac{3}{4}}$ in order to satisfy a relation of the form $ex + y = 0 \bmod \phi(N)$. Thus, $|y|$ can be chosen of size at least x. If e is roughly N, which is normally the case for small d, $|y|$ can even be chosen of size $N^{\frac{1}{4}}x$ in the attack.

One should expect that for fixed N the number of public keys (N, e) for which our approach applies is roughly the number of tuples (x, y) within the given bounds. This number can be upper bounded by $x \cdot N^{\frac{1}{4}}x \leq N^{\frac{3}{4}}$. In fact, we are able to show that the number of weak keys (N, e) for which our algorithm works is also lower bounded by $\Omega(N^{\frac{3}{4}-\epsilon})$.

It is important to notice that in contrast to the approaches of Wiener and Boneh-Durfee, the secret keys in our attack are not small itself but have a "small decomposition" in x and y. So they might look innocuous to crypto-designers

and may be tempting to use in the design of cryto-systems with good encryption/decryption performance.

As an example, we show that the public keys (N, e) constructed in the YKLM-scheme can be attacked by our generalization of Wiener's method. Namely, we can express the secret exponent d in terms of small x and y, which breaks the crypto-system for all reasonable parameter choices.

In 2002, de Weger [9] observed that Wiener's attack can be improved when the prime difference $p-q$ is significantly less than $\sqrt{N}$. de Weger's method also applies to our extension of Wiener's attack. Interestingly, we are able to show that for prime difference $p - q = N^{\frac{1}{4}+\gamma}$, $0 < \gamma \leq \frac{1}{4}$ there are at least $N^{1-\gamma-\epsilon}$ weak RSA-keys (N, e).

It is important to notice that for prime difference $p-q = \mathcal{O}(N^{\frac{1}{4}})$ an algorithm of Fermat finds the factorization in polynomial time. Thus, our attack has a nice interpolation property towards Fermat's algorithm: As $p-q$ decreases, the number of weak public keys increases. For γ approaching zero almost all keys are weak, corresponding to the fact that N can be easily factored without any hint that is encoded in e.

As a by-product, we get a simple probabilistic factorization algorithm with expected running time $\mathcal{O}(N^{\gamma+\epsilon})$ comparable to Fermat-Factorization: For a fixed N, choose random $e < N$ and apply our algorithm to each choice (N, e) until (N, e) is a weak key that yields the factorization.

Notice that the interpolation property above seems to imply that one cannot improve our approach significantly. On the other hand, there might be different techniques – for example lattice reduction techniques for higher dimensional lattices – that lead to larger classes of weak keys for the prime difference $p-q = \Omega(\sqrt{N})$. But at the moment this is an open question.

The paper is organized as follows: In Section 2, we present our extension of Wiener's attack. As an application of this method, we present the crytanalysis of the YKLM-scheme in Section 3. In Section 4, we apply the methods of de Weger to our generalized Wiener attack. We conclude the paper by showing in Section 5 that the number of weak RSA-keys (N, e) in our approach is $\Omega(N^{\frac{3}{4}-\epsilon})$.

2 The Generalized Wiener Attack

Throughout this work we consider RSA-moduli $N = pq$, where p and q are of the same bit-size (wlog $p > q$). This implies the inequalities

$$p - q \leq N^{\frac{1}{2}} \quad \text{and} \quad 2N^{\frac{1}{2}} \leq p + q \leq 3N^{\frac{1}{2}}.$$

Furthermore, we have $\phi(N) = N + 1 - (p + q) > \frac{N}{2}$.
Our attack makes use of a well-known result due to Coppersmith [4]:

Theorem 1 (Coppersmith) *Let $N = pq$ be an RSA-modulus, where p and q are of the same bit-size. Suppose we are given an approximation of p with additive error at most $N^{\frac{1}{4}}$. Then N can be factored in time polynomial in* $\log N$.

We are now able to state our main theorem. Here we consider the normal RSA-case where $p - q = \Omega(\sqrt{N})$.

Theorem 2 *Let $c \leq 1$ and let (N, e) be an RSA public key tuple with $N = pq$ and $p - q \geq cN^{\frac{1}{2}}$. Suppose that $e \in Z^*_{\phi(N)}$ satisfies an equation $ex + y = k\phi(N)$ with*

$$0 < x \leq \frac{1}{3}N^{\frac{1}{4}} \quad \text{and} \quad |y| \leq cN^{-\frac{3}{4}}ex.$$

Then N can be factored in polynomial time.

One should notice that the conditions of Theorem 2 imply that $ex + y \neq 0$, thereby excluding trivial congruences: Since $c \leq 1$, we see that $|y| < ex$. This in turn implies $k > 0$.

Roadmap for the proof of Theorem 2

- We show that the unknown parameters x, k can be found among the convergents of the continued fraction expansion of $\frac{e}{N}$.
- From x and k, we compute an approximation of $p + q$.
- From an approximation of $p + q$, we compute an approximation of $p - q$.
- Combining both approximations gives us an approximation of p, which leads to the factorization of N by using Coppersmith's Theorem.

We want to argue that in the following proof we can assume wlog that $N \geq (\frac{8}{c})^4$. This condition is equivalent to $c \geq 8N^{-\frac{1}{4}}$. If this inequality does not hold then $p - q = \mathcal{O}(N^{\frac{1}{4}})$ and Fermat's factorization algorithm yields the factorization of N in polynomial time.

Proof: Let us start with the RSA key equation

$$ex + y = k(N - p - q + 1). \tag{1}$$

Dividing by Nx gives us

$$\frac{e}{N} - \frac{k}{x} = -\frac{k(p + q - 1) + y}{Nx}.$$

We want to argue that we can assume wlog that $\gcd(k, x) = 1$. Notice that every integer that divides both k and x must also divide y by equation (1). Thus, we can divide equation (1) by $\gcd(k, x)$ which gives us an equation $ex' + y' = 0 \bmod \phi(N)$ with even smaller parameters x' and y'. Hence we can assume that $\frac{k}{x}$ is a fraction in its lowest terms.

By a well-known theorem (see e.g. Theorem 184 in [7]), the fraction $\frac{k}{x}$ appears among the convergents of $\frac{e}{N}$ if the condition $|\frac{e}{N} - \frac{k}{x}| < \frac{1}{2x^2}$ is satisfied. Thus it remains to show that $|k(p + q - 1) + y| < \frac{N}{2x}$. Let us first find a bound for the parameter k. We know that $k = \frac{ex+y}{\phi(N)}$ and $|y| \leq cN^{-\frac{3}{4}}ex$. Since our precondition $N \geq (\frac{8}{c})^4$ implies $N \geq 2^{12}$, we conclude that $|y| \leq \frac{1}{4}ex$. Therefore, we obtain

$$\frac{3}{4}\frac{ex}{\phi(N)} \leq k \leq \frac{5}{4}\frac{ex}{\phi(N)}. \tag{2}$$

Now we are able to estimate

$$k(p+q-1)+y \leq \frac{15}{4}\frac{ex}{\phi(N)} \cdot N^{\frac{1}{2}} + cN^{-\frac{3}{4}}ex \leq \frac{15}{4}xN^{\frac{1}{2}} + xN^{\frac{1}{4}} \leq 4xN^{\frac{1}{2}},$$

where the last inequality holds for $N \geq 2^{12}$.

Therefore, we have to satisfy the condition $4xN^{\frac{1}{2}} < \frac{N}{2x}$ which is equivalent to $x < \frac{1}{\sqrt{8}}N^{\frac{1}{4}}$. This condition holds by our upper bound $x \leq \frac{1}{3}N^{\frac{1}{4}}$.

Hence, the fraction $\frac{k}{x}$ must be among the convergents of the continued fraction expansion of $\frac{e}{N}$. Since there are only $\mathcal{O}(\log N)$ many convergents, we can apply the following process to each candidate for k and x until our algorithm succeeds.

We have to show that the correct k and x yield the factorization of N. Let us write equation (1) as

$$N + 1 - \frac{ex}{k} = p + q + \frac{y}{k}.$$

Since every parameter on the left hand side is now known to us, we can compute an approximation of $p+q$ up to some unknown error term $\frac{y}{k}$, that can be bounded by $|\frac{y}{k}| \leq \frac{4}{3}cN^{\frac{1}{4}}$ using inequality (2).

Our goal is to find an approximation of p up to some error of size $N^{\frac{1}{4}}$ in order to apply Coppersmith's theorem. Therefore, we transform our approximation of $p+q$ into an approximation of $p-q$ using the relation

$$p - q = \sqrt{(p-q)^2} = \sqrt{(p+q)^2 - 4N}.$$

Let s be our approximation of $p+q$ with additive error at most $\frac{4}{3}cN^{\frac{1}{4}}$. We will show that $t = \sqrt{s^2 - 4N}$ is an approximation of $p-q$ with an additive error that can be bounded by $9N^{\frac{1}{4}}$. Thus, the term $\frac{1}{2}(s+t)$ is an approximation of p with error at most

$$\begin{aligned}\left|\frac{1}{2}(s+t) - p\right| &= \frac{1}{2}|s - p - q + t - p + q| \\ &\leq \frac{1}{2}|s-(p+q)| + \frac{1}{2}|t-(p-q)| \\ &\leq \frac{2}{3}cN^{\frac{1}{4}} + \frac{9}{2}N^{\frac{1}{4}} \leq 6N^{\frac{1}{4}}\end{aligned}$$

Define $\tilde{p} = \frac{1}{2}(s+t)$. Then one out of the six values $\tilde{p} + (2k+1)N^{\frac{1}{4}}$, $k = -3,-2,-1,0,1,2$ is an approximation of p up to an error of at most $N^{\frac{1}{4}}$ in absolute value. We can apply Coppersmith's algorithm to all these values. The correct term will then lead to the factorization of N in polynomial time.

It remains to show that $t = \sqrt{s^2 - 4N}$ is indeed an approximation of $p-q$ up to some error term that can be bounded by $9N^{\frac{1}{4}}$. Let us first show that t is well-defined, i.e. $s^2 - 4N \geq 0$. Observe that $s = p + q + \frac{y}{k}$ satisfies

$$s^2 - 4N = (p-q)^2 + 2\frac{y}{k}(p+q) + \left(\frac{y}{k}\right)^2.$$

Therefore, it suffices to show that $|2\frac{y}{k}(p+q)| \leq (p-q)^2$. Using $|\frac{y}{k}| \leq \frac{4}{3}cN^{\frac{1}{4}}$, we obtain $|2\frac{y}{k}(p+q)| \leq 8cN^{\frac{3}{4}}$. From our precondition $N \geq (\frac{8}{c})^4$, we see that $8 \leq cN^{\frac{1}{4}}$. This immediately implies $8cN^{\frac{3}{4}} \leq c^2N \leq (p-q)^2$ as desired.

Since $N \geq 2^{12}$, we know that the error term $\frac{y}{k}$ for $p+q$ can be bounded in absolute value by $\frac{4}{3}cN^{\frac{1}{4}} \leq \frac{1}{2}N^{\frac{1}{2}} \leq \frac{1}{4}(p+q)$. This implies the inequality

$$s \leq \frac{5}{4}(p+q). \tag{3}$$

We observe that

$$t-(p-q) = \sqrt{s^2-4N} - (p-q) = \frac{(s-(p+q))(s+(p+q))}{\sqrt{s^2-4N}+(p-q)}.$$

Using the inequalities (3), $s-(p+q) \leq \frac{4}{3}cN^{\frac{1}{4}}$ and $p-q \geq cN^{\frac{1}{2}}$ finally leads us to the desired bound

$$t-(p-q) \leq \frac{\frac{4}{3}cN^{\frac{1}{4}} \cdot \frac{27}{4}N^{\frac{1}{2}}}{(p-q)} \leq 9N^{\frac{1}{4}}.$$

Let us briefly summarize the whole factorization algorithm.

Algorithm Generalized Wiener Attack

INPUT: (N,e), where $N=pq$ and $ex+y=0 \bmod \phi(N)$ for some unknown $0 < x \leq \frac{1}{3}N^{\frac{1}{4}}$ and $|y| \leq cN^{-\frac{3}{4}}ex$.

1. Compute the continued fraction expansion of $\frac{e}{N}$.
2. For every convergent $\frac{k}{x}$ of the expansion:
   (a) Compute $s = N+1-\frac{ex}{k}$, $t=\sqrt{s^2-4N}$ and $\tilde{p}=\frac{1}{2}(s+t)$.
   (b) Apply Coppersmith's algorithm to the candidates $\tilde{p}+(2k+1)N^{\frac{1}{4}}$ for $k=-3,-2,\ldots,2$: If Coppersmith's algorithm outputs the factorization of N, then stop.

OUTPUT: p,q

Since every step in Algorithm Generalized Wiener-Attack can be done in polynomial time and the number of convergents is bounded by $\mathcal{O}(\log N)$, this concludes the proof of Theorem 2. □

3 Cryptanalysis of the YKLM-scheme

In 2001, Yen, Kim, Lim and Moon [11,12] presented an RSA-type scheme that was designed to counteract the Bellcore-attack (see [2]). Unfortunately, they

need a specialized RSA key generation process in order to make their scheme efficient. Their public key e satisfies a relation with some small parameters that will be described in this section. The efficiency of the YKLM-scheme relies on the fact that these parameters are indeed much smaller than the modulus N. It was raised as an open question by the authors if one can use random public keys e as well in their scheme, thereby maintaining the same performance.

We show that the public keys constructed in the YKLM-scheme satisfy the conditions of Theorem 2, i.e. for every public exponent e we have $ex + y = 0 \bmod \phi(N)$ with small x and y.

Let us first reconsider the modified key generation algorithm in the YKLM-scheme.

RSA Key Generation in the YKLM-scheme

Modulus : Choose randomly two primes p and q of the same bit-size and compute the product $N = pq$.
Small parameters : Fix a bound B, where $B \ll N$. Choose randomly e_r and r in $\{1,\ldots,B\}$ such that $\gcd(e_r, \phi(N)) = 1$. Compute $d_r = e_r^{-1} \bmod \phi(N)$.
Secret exponent : Compute $d = d_r + r$. If $\gcd(d, \phi(N)) \neq 1$, choose different parameters e_r and r.
Public exponent : Compute $e = d^{-1} \bmod \phi(N)$.
Public parameters : Publish the tuple (N, e).

The authors pointed out that instead of the public key tuple (N, e) one could even publish the parameters e_r and r as well, but the following observation shows that the parameters e_r and r immediately yield the factorization of N.

Consider the public key equation

$$ed - 1 = 0 \bmod \phi(N)$$

The secret key d has a decomposition into the unknown part d_r and the known parameter r

$$e(d_r + r) - 1 = 0 \bmod \phi(N).$$

Multiplication with e_r removes the unknown parameter d_r

$$e(1 + e_r r) - e_r = 0 \bmod \phi(N).$$

Since every parameter on the left hand side is known, we can compute a multiple $k\phi(N)$ of the Euler function

$$e(1 + e_r r) - e_r = k\phi(N) \quad \text{for some } k \in \mathbb{N}. \tag{4}$$

Since $e < \phi(N)$, we have that $k < (1 + e_r r)$. Therefore, the bit-length of k is polynomial in the bit-length of N. It is a well-known result that such a multiple $k\phi(N)$ yields the factorization of N in probabilistic polynomial time in the bit-length of N (see for example [8]).

Certainly, there is no need to publish the small parameters e_r and r in the YKLM-scheme. On the other hand, we see that by equation (4) one can apply Theorem 2 by setting $x = 1 + e_r r$ and $y = -e_r$. This gives us the following corollary from Theorem 2.

Corollary 3 *Let $c \leq 1$ and let (N, e) be a public key tuple constructed by the key generation process in the YKLM-scheme with $p-q \geq cN^{\frac{1}{2}}$. Furthermore, let e_r and r satisfy the conditions*

$$1 + e_r r \leq \frac{1}{3} N^{\frac{1}{4}} \quad \textit{and} \quad e_r \leq \frac{1}{2} cN^{\frac{1}{4}}$$

Then N can be factored in time polynomial in $\log(N)$.

Proof: In order to be able to apply Theorem 2, it remains to show that $\frac{1}{2}cN^{\frac{1}{4}} \leq cN^{-\frac{3}{4}}e(1 + e_r r)$. Using equation (4), we conclude that

$$cN^{-\frac{3}{4}}e(1 + e_r r) > cN^{-\frac{3}{4}}\phi(N) > \frac{1}{2}cN^{\frac{1}{4}},$$

which proves the claim. □

Since the efficiency of the YKLM-scheme relies on the fact that e_r and r are very small compared to N, Corollary 3 breaks the YKLM-scheme for all reasonable parameter choices.

4 Generalizing to Arbitrary Prime Differences $p - q$

de Weger [10] observed that Wiener's attack can be improved when $p-q$ is significantly smaller than $\sqrt{N}$. He showed that $N' = N - \lfloor 2\sqrt{N} \rfloor$ is an approximation of $\phi(N)$ with error at most $\frac{(p-q)^2}{\sqrt{N}}$. Thus, using the continued fraction expansion $\frac{e}{N'}$ instead of $\frac{e}{N}$ leads to an improvement in Wiener's algorithm. Namely, de Weger proved that for prime differences $p - q$ of size $N^{\frac{1}{4}+\gamma}$, $0 \leq \gamma \leq \frac{1}{4}$ one can achieve a bound of $d \leq N^{\frac{1}{2}-\gamma}$ in Wiener's algorithm.

The same trick applies to our generalized version of Wiener's attack (Section 2) as well. This gives us the following more general result.

Theorem 4 *Given an RSA public key tuple (N, e), where $N = pq$. Suppose that e satisfies an equation $ex + y = 0 \bmod \phi(N)$ with*

$$0 < x \leq \frac{1}{3}\sqrt{\frac{\phi(N)}{e}}\frac{N^{\frac{3}{4}}}{p-q} \quad \textit{and} \quad |y| \leq \frac{p-q}{\phi(N)N^{\frac{1}{4}}} \cdot ex.$$

Then N can be factored in time polynomial in $\log N$.

Proof. The proof is similar to the proof of Theorem 2. One mainly substitutes N by $N' = N - \lfloor 2\sqrt{N} \rfloor$ and works through the arithmetic. Therefore we omit the proof.

Instead we give the factorization algorithm.

Algorithm Generalized Wiener Attack II

INPUT: (N, e), where $N = pq$ and $ex + y = 0 \bmod \phi(N)$ for some unknown $0 < x \leq \frac{1}{3}\sqrt{\frac{\phi(N)}{e}}\frac{N^{\frac{3}{4}}}{p-q}$ and $|y| \leq \frac{p-q}{\phi(N)N^{\frac{1}{4}}} \cdot ex$.

1. Set $N' = N - \lfloor 2\sqrt{N} \rfloor$ and compute the continued fraction expansion of $\frac{e}{N'}$.
2. For every convergent $\frac{k}{x}$ of the expansion:
   (a) Compute $s = N + 1 - \frac{ex}{k}$, $t = \sqrt{s^2 - 4N}$ and $\tilde{p} = \frac{1}{2}(s + t)$.
   (b) Apply Coppersmith's algorithm to the candidates $\tilde{p} + (2k + 1)N^{\frac{1}{4}}$ for $k = -3, -2, \ldots, 2$: If Coppersmith's algorithm outputs the factorization of N, then stop.

OUTPUT: p, q

5 There Are $N^{\frac{3}{4}-\epsilon}$ Weak RSA-keys

In Section 4, we showed that every public key tuple (N, e) that satisfies a relation $ex + y = 0 \bmod \phi(N)$, with

$$0 < x \leq \frac{1}{3}\sqrt{\frac{\phi(N)}{e}}\frac{N^{\frac{3}{4}}}{p-q} \quad \text{and} \quad |y| \leq \frac{p-q}{\phi(N)N^{\frac{1}{4}}} \cdot ex. \tag{5}$$

yields the factorization of N in polynomial time. Those tuples (N, e) are *weak keys* that should not be used in the design of a crypto-system. Let us formalize the notion of weak keys.

Definition 5 *Let C be a class of RSA public keys (N, e). The size of the class C is defined by*

$$size_C(N) = |\{e \in \mathbb{Z}^*_{\phi(N)} \mid (N, e) \in C\}|.$$

C is called weak *if:*

1. *$size_C(N) = \Omega(N^\gamma)$ for some $\gamma > 0$.*
2. *There exists a probabilistic algorithm A that on every input $(N, e) \in C$ outputs the factorization of N in time polynomial in* $\log(N)$.

The elements of a weak class are called weak keys.

Our variant of Wiener's attack in Section 4 defines a weak class C. The question we will study in this chapter is, how large this weak class is.

What bounds can we expect for $size_C(N)$? As a first estimate we can sum over all tuples (x, y) within the bounds given by the inequalities in (5). This gives us an upper bound on the size of C. Therefore, we have at most

$$\text{size}_C(N) \leq \left(\frac{1}{3}\sqrt{\frac{\phi(N)}{e}}\frac{N^{\frac{3}{4}}}{p-q}\right)^2 \cdot \frac{e}{\phi(N)}\frac{p-q}{N^{\frac{1}{4}}} = \mathcal{O}\left(\frac{N^{\frac{5}{4}}}{p-q}\right) \tag{6}$$

weak keys. This is an upper bound on $\text{size}_C(N)$ since:

- Different tuples (x, y) might define the same public exponent e.
- Some of the tuples (x, y) do not even define a legitimate public key e, e.g. a key $e \in Z^*_{\phi(N)}$.

Instead of an upper bound on $\text{size}_C(N)$, we are interested in a lower bound. Namely, we want to know the minimal number of public exponents $e \in \mathbb{Z}^*_{\phi(N)}$ that yield the factorization for some fixed modulus N. In this section we will prove a lower bound for $\text{size}_C(N)$.

As the result we obtain that our lower bound almost perfectly matches the upper bound: If $p - q = \Omega(N^{\frac{1}{4}+\gamma})$, $\gamma > 0$, we obtain a lower bound of

$$\text{size}_C(N) = \Omega\left(\frac{N^{\frac{5}{4}-\epsilon}}{p-q}\right).$$

Let us have a closer look at this result. In the common RSA case, we have $p - q = \mathcal{O}(N^{\frac{1}{2}})$ which implies a bound of

$$\text{size}_C(N) = \Omega\left(N^{\frac{3}{4}-\epsilon}\right)$$

weak RSA key tuples (N, e).

On the other hand, we know that Fermat's factorization algorithm yields the factorization of N in polynomial time if $p - q = \mathcal{O}(N^{\frac{1}{4}})$. But the number of weak keys for $p - q = N^{\frac{1}{4}+\gamma}$, $0 < \gamma \leq \frac{1}{4}$ is $\Omega(N^{1-\gamma-\epsilon})$. That means that the number of weak keys scales almost perfectly with the prime difference $p - q$. As $p - q$ decreases, the number of weak key tuples increases and as γ approaches zero almost all keys are weak. This corresponds to the fact that for $\gamma = 0$, all tuples (N, e) are weak because one can find the factorization of N in polynomial time with Fermat's algorithm.

We will now prove the lower bound result, where we use the following main lemma.

Lemma 6 *Let $f(N, e)$, $g(N, e)$ be functions such that $f^2(N, e)g(N, e) < \phi(N)$, $f(N, e) \geq 2$ and $g(N, e) \leq f(N, e)$. The number of public keys $e \in \mathbb{Z}^*_{\phi(N)}$, $e \geq \frac{\phi(N)}{4}$ that satisfy an equation $ex + y = 0 \bmod \phi(N)$ for $x \leq f(N, e)$ and $|y| \leq g(N, e)x$ is at least*

$$\frac{f^2(N, e)g(N, e)}{8 \log \log^2(N^2)} - \mathcal{O}(f^2(N, e)N^{\epsilon}),$$

where $\epsilon > 0$ is arbitrarily small for N suitably large.

Using Lemma 6, we can immediately prove our lower bound theorem.

Theorem 7 *Let $p - q = N^{\frac{1}{4}+\gamma}$ with $0 < \gamma \leq \frac{1}{4}$. Further, let C be the weak class that is given by the public key tuples (N, e) defined in Theorem 4 with the additional restriction that $e \in Z^*_{\phi(N)}$, $e \geq \frac{\phi(N)}{4}$. Then*

$$size_C(N) = \Omega\left(\frac{N^{1-\gamma}}{\log\log^2(N^2)}\right).$$

Proof: Using the bounds of (5), we define

$$f(N,e) = \frac{1}{3}\sqrt{\frac{\phi(N)}{e}}\frac{N^{\frac{3}{4}}}{p-q} \quad \text{and} \quad g(N,e) = \frac{e}{\phi(N)}\frac{p-q}{N^{\frac{1}{4}}}.$$

It can be easily checked that these settings fulfill the requirements of Lemma 6:

$$f^2(N,e)g(N,e) < \phi(N),\ f(N,e) \geq 2 \text{ and } g(N,e) \leq f(N,e).$$

Hence, we can apply Lemma 6. Since $g(N,e) = \Omega(N^{\gamma})$, the term

$$\frac{f^2(N,e)g(N,e)}{8\log\log^2(N^2)}$$

dominates the error term $\mathcal{O}(f^2(N,e)N^{\epsilon})$.
Using $f^2(N,e)g(N,e) = \Omega(\frac{N^{\frac{5}{4}}}{p-q})$ and $p - q = N^{\frac{1}{4}+\gamma}$ proves the claim. □

We obtain the following corollary.

Corollary 8 *Let C be the weak class that is given by the public key tuples (N, e) defined in Theorem 2 with the additional restriction that $e \in \mathbb{Z}^*_{\phi(N)}$, $e \geq \frac{\phi(N)}{4}$. Then*

$$size_C(N) = \Omega\left(\frac{N^{\frac{3}{4}}}{\log\log^2(N^2)}\right).$$

It remains to prove Lemma 6. Since the proof is technical, we describe just the rough idea and leave the details to the full version of the paper.

As denoted before, different tuples (x, y) might define the same public exponent e and some tuples (x, y) do not define a legitimate key $e \in Z^*_{\phi(N)}$. Therefore, we define a suitably large subclass T of all tuples (x, y) within the given bounds $x \leq f(N,e)$ and $|y| \leq g(N,e)x$ such that different tuples define different legitimate keys e.

References

1. T. M. Apostol, *Introduction to analytic number theory,* Springer-Verlag, 1980.
2. D. Boneh, R. DeMillo, R. Lipton, "On the importance of checking cryptographic protocols for faults", in *Proceedings of Eurocrypt'97,* Lecture Notes in Computer Science, Vol. 1233, Springer Verlag, 1997.

3. D. Boneh, G. Durfee, "Cryptanalysis of RSA with private key d less than $N^{0.292}$", *IEEE Trans. on Information Theory,* Vol. 46(4), 2000
4. D. Coppersmith, "Small solutions to polynomial equations and low exponent vulnerabilities", *Journal of Cryptology,* Vol. 10(4), pp. 223–260, 1997.
5. C. Crépeau, A. Slakmon, "Simple Backdoors for RSA Key Generation", in *Topics in Cryptology - CT-RSA 2003,* Lecture Notes in Computer Science Vol. 2612, pp. 403–416, Springer-Verlag, 2003
6. N. Howgrave-Graham, "Approximate integer common divisors", *Cryptography and Lattices,* Lecture Notes in Computer Science, Vol. 2146, Springer-Verlag, 2001.
7. G. H. Hardy, E. M. Wright, *Introduction to the theory of numbers,* Oxford University Press, 1979.
8. N. Koblitz, *A course in number theory and cryptography,* Springer-Verlag, 1994
9. B. de Weger, "Cryptanalysis of RSA with small prime difference", *Applicable Algebra in Engineering, Communication and Computing,* Vol. 13(1), pp. 17–28, 2002.
10. M. Wiener, "Cryptanalysis of short RSA secret exponents", *IEEE Transactions on Information Theory,* Vol. 36, pp. 553–558, 1998.
11. S. -M. Yen, S. Kim, S. Lim, S. Moon, "Speedup with Residue Number System Immune against Hardware Fault Cryptanalysis", 4^{th} *International Conference on Information Security and Cryptology,* Lecture Notes in Computer Science, Vol. 2288, pp. 397–413, Springer-Verlag, 2001.
12. S. -M. Yen, S. Kim, S. Lim, S. Moon, "RSA Speedup with Chinese Remainder Theorem Immune against Hardware Fault Cryptanalysis", *IEEE Transactions on Computers,* Vol. 52(4), 2003

Cryptanalysis of a Public-Key Encryption Scheme Based on the Polynomial Reconstruction Problem

Jean-Sébastien Coron

Gemplus Card International
34 rue Guynemer
Issy-les-Moulineaux, F-92447, France
jean-sebastien.coron@gemplus.com

Abstract. We describe a cryptanalysis of a public-key encryption scheme based on the polynomial reconstruction problem, published at Eurocrypt 2003 by Augot and Finiasz. Given the public-key and a ciphertext, we recover the corresponding plaintext in polynomial time. Our technique is a variant of the Berlekamp-Welsh algorithm. We also describe a cryptanalysis of the reparation published by the authors on the IACR eprint archive, using a variant of the previous attack. Both attacks are practical as given the public-key and a ciphertext, one recovers the plaintext in a few minutes on a single PC.

Key-Words: Cryptanalysis, Augot and Finiasz cryptosystem, Polynomial Reconstruction Problem, Reed-Solomon codes.

1 Introduction

We describe a cryptanalysis of a public-key encryption scheme recently proposed by Augot and Finiasz [1]. The scheme is based on the polynomial reconstruction (PR) problem [10], which is the following:

Problem 1 (Polynomial Reconstruction). Given n, k, ω and $(x_i, y_i)_{i=1\ldots n}$, output any polynomial p such that $\deg p < k$ and $p(x_i) = y_i$ for at least $n - \omega$ values of i.

This problem has an equivalent formulation in terms of the decoding of Reed-Solomon error-correcting codes [11]. The problem can be solved in polynomial time when the number of errors ω is such that $\omega \leq (n-k)/2$, using the Berlekamp-Welsh algorithm [3]. This has been improved to $\omega \leq n - \sqrt{kn}$ by Guruswami and Sudan [7].

When the number of errors is larger, no polynomial time algorithm is known for the PR problem. Therefore, some cryptosystem have been constructed based on the hardness of the PR problem; for example, an oblivious polynomial evaluation scheme [10], and a semantically secure symmetric cipher [8].

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 14–27, 2004.

At Eurocrypt 2003, Augot and Finiasz proposed a new public-key encryption scheme based on the polynomial reconstruction problem [1]. A security level exponential in terms of the parameters was conjectured. However, we provide a complete cryptanalysis of the cryptosystem: given the public key *pk* and a ciphertext c, we recover the corresponding plaintext m in polynomial time. Therefore, the scheme is not one-way and cannot be used in any application. Our technique is a variant of the Berlekamp-Welsh algorithm [3] for solving the PR problem.

After the publication of our attack in the IACR eprint archive [5], a reparation of the cryptosystem was published by Augot, Finiasz and Loidreau in [2]. The reparation is based on the trace operator, and is resistant against the previous attack. However, we describe a new cryptanalysis of the repaired scheme. Given the public-key and a ciphertext, we can still recover the corresponding plaintext in polynomial time. Our technique is again a variant of the Berlekamp-Welsh algorithm. Both attacks work very well in practice, as for the proposed parameters, one recovers the plaintext in a few minutes on a single PC.

2 Augot and Finiasz' Cryptosystem

In this section, we recall the original cryptosystem proposed by Augot and Finiasz at Eurocrypt 2003. As in [1], we first recall some basic definitions of Reed-Solomon codes.

2.1 Reed-Solomon Codes

Let F_q be the finite field with q elements and let $x_1, \cdots, x_n$ be n distinct elements of F_q. We denote by ev the following map:

$$ev : \begin{cases} F_q[X] \to F_q^n \\ p(X) \to (p(x_1), \ldots, p(x_n)) \end{cases}$$

Definition 1. *The Reed-Solomon code of dimension k and length n over F_q is the following set of n-tuples (codewords):*

$$RS_k = \{ev(f); f \in F_q[X], \deg f < k\}$$

where $F_q[X]$ is the set of univariate polynomials with coefficients in F_q.

The *weight* of a word $c \in F_q^n$ is the number of non-zero coordinates in c. The *Hamming distance* between two words x and y is the weight of $x - y$. Formally, the problem of decoding Reed-Solomon code is the following:

Problem 2 (Reed-Solomon decoding). Given a Reed-Solomon code RS_k of length n, ω an integer and a word $y \in F_q^n$, find any codeword in RS_k at distance less than ω of y.

The smallest weight of non-zero codewords in RS_k is $n - k + 1$. Therefore, when $\omega \leq (n-k)/2$, the solution to Reed-Solomon decoding is guaranteed to be unique. It is easy to see that the Polynomial Reconstruction problem and the Reed-Solomon decoding problem are equivalent. Both problems can be solved in polynomial time when $w \leq (n-k)/2$, using the Berlekamp-Welsh algorithm [3].

2.2 Augot and Finiasz' Cryptosystem

In the following, we briefly review Augot and Finiasz public-key cryptosystem [1].

Parameters: q is the size of F_q, n is the length of the Reed-Solomon code, k its dimension, W is the weight of a large error, so that the PR problem for n, k, W is believed to be hard, i.e. we must have:

$$W > \frac{n-k}{2}$$

ω is the weight of a small error, for which the PR problem with $n-W$ coordinates is easy:

$$\omega \leq \frac{n-W-k}{2} \tag{1}$$

It is recommended in [1] to take $n = 1024$, $k = 900$, $\omega = 25$, $W = 74$ and $q = 2^{80}$.

Key Generation: Generate a unitary polynomial p of degree $k-1$, and a random n-dimensional vector E of weight W. Compute the codeword $c = ev(p)$ of RS_k. The public key is $z = c + E$, while the private key is (p, E).

Encryption: Let m a message of length $k-1$ over the alphabet F_q. The message m is seen as a polynomial $m(X) = m_0 + m_1 X + \ldots + m_{k-1}X^{k-2}$ of degree at most $k-2$. Generate a random $\alpha \in F_q$ and a random error e of weight ω. The ciphertext y is then:

$$y = ev(m) + \alpha \times (c + E) + e$$

Decryption: One considers only the positions where $E_i = 0$ and define the shortened code of length $n-W$, which is also a Reed-Solomon code of dimension k, which we denote $\overline{RS}_k$. Let $\overline{y}, \overline{ev}(m), \overline{c}, \overline{e}$ be the shortened $y, ev(m), c, e$. One must solve the equation:

$$\overline{y} = \overline{ev}(m) + \alpha \times \overline{c} + \overline{e}$$

We have $\overline{ev}(m) + \alpha \times \overline{c} \in \overline{RS}_k$, and from (1), the weight of the small error $\overline{e}$ is less than the error correction capacity of $\overline{RS}_k$; therefore, using the Berlekamp-Welsh algorithm, one can recover the unique polynomial r of degree $k-1$ such that:

$$ev(r) = \overline{ev}(m) + \alpha \times \overline{c}$$

which gives

$$r = m + \alpha \cdot p$$

Since $\deg(m) \leq k-2$ and p is a unitary polynomial of degree $k-1$, the field element α is the leading coefficient of r. Therefore one can recover m as:

$$m = r - \alpha \cdot p$$

3 Our Attack

The attack is a variant of the Berlekamp-Welsh algorithm for solving the PR problem (see [6]).

Let n, k, W, ω and q be the parameters of the system. Let (p, E) be the private key and $z = ev(p) + E$ be the public-key. Let m be the plaintext encoded as a polynomial of degree less than $k-2$. Let e be an error vector of weight ω, and α be a field element. Let

$$y = ev(m) + \alpha \times z + e \tag{2}$$

be the corresponding ciphertext.

Theorem 1. *Given the public-key z and the ciphertext y, one can recover the corresponding plaintext m in polynomial time.*

Proof. Let y_i, z_i and e_i be the components of the words y, z and e. Given y and z, one must solve the following set of equations:

$$\exists e, m, \alpha,\ y_i = m(x_i) + \alpha \cdot z_i + e_i \quad \text{for all } 1 \leq i \leq n \tag{3}$$

where the weight of e is less than ω. Note that from the definition of the cryptosystem, there is a unique solution.

Consider the following set of equations:

$$\exists V, m, \alpha,\ \begin{cases} \deg(V) \leq \omega,\ V \neq 0,\ \deg(m) \leq k-2 \\ \forall i,\ \ V(x_i) \cdot (y_i - \alpha \cdot z_i) = V(x_i) \cdot m(x_i) \end{cases} \tag{4}$$

Any solution V, m, α of (4) gives a solution to (3). Namely, the fact that $V \neq 0$ and $\deg V \leq \omega$ implies that V can be equal to zero at most ω times. Therefore, letting $e_i = y_i - m(x_i) - \alpha \cdot z_i$, the weight of e is less than ω.

Conversely, any solution to (3) gives a solution to (4). Namely, one can take $V(X) = \prod_{i \in B}(X - x_i)$ with $B = \{i | e_i \neq 0\}$. The problem of solving (3) can thus be reduced to finding V, m, α satisfying (4). Consider now the following set of equations:

$$\exists V, N, \lambda,\ \begin{cases} \deg(V) \leq \omega,\ V \neq 0,\ \deg(N) \leq k + \omega - 1 \\ \forall i,\ \ V(x_i) \cdot (y_i - \lambda \cdot z_i) = N(x_i) \end{cases} \tag{5}$$

The system (5) is a linearized version of (4), in which one has replaced the product $V(x_i) \cdot m(x_i)$ by $N(x_i)$. It is easy to see that any solution of (4) gives a solution to (5), as one can take $\lambda = \alpha$ and $N = m \cdot V$. However, the converse is not necessarily true.

For a given λ, the system (5) gives a linear system of n equations in the $k + 2 \cdot \omega + 1$ unknown, which are the coefficients of the polynomials V and N. More precisely, denoting:

$$V(X) = \sum_{i=0}^{\omega} v_i \cdot X^i, \quad N(X) = \sum_{i=0}^{k+\omega-1} n_i \cdot X^i$$

and Y the vector of coordinates:

$$Y = (v_0, \cdots, v_\omega, n_0, \cdots, n_{k+\omega-1})$$

one let $M(\lambda)$ be the matrix of the system:

$$M(\lambda)_{i,j} = \begin{cases} (y_i - \lambda \cdot z_i) \cdot (x_i)^j & \text{if } 0 \leq j \leq \omega \\ -(x_i)^{j-\omega-1} & \text{if } \omega < j < k + 2\omega + 1 \end{cases}$$

The matrix $M(\lambda)$ is a rectangular matrix with n lines and $k + 2\omega + 1$ columns; from (1) we have that $n > k + 2\omega + 1$. The coefficients of $M(\lambda)$ are a function of the public-key and the ciphertext only. The system (5) is then equivalent to:

$$\exists Y, \lambda, \quad M(\lambda).Y = 0, \ Y \neq 0 \tag{6}$$

We consider the matrix $M(\lambda)$ with $\lambda = 0$. Using Gaussian elimination, we compute the rank of the matrix $M(0)$. We distinguishe two cases: *rank* $M(0) = k + 2\omega + 1$, and *rank* $M(0) < k + 2\omega + 1$.

If *rank* $M(0) = k + 2\omega + 1$, then there exists a square sub-matrix of $M(0)$ of dimension $k + 2\omega + 1$ which is invertible. Without loss of generality, one can assume that the matrix obtained by taking the first $k + 2\omega + 1$ lines of $M(0)$ is invertible. Let $M'(\lambda)$ be the square matrix obtained by taking the first $k+2\omega+1$ lines of $M(\lambda)$. Any solution *Y, λ* of (6) satisfies:

$$M'(\lambda).Y = 0, \ Y \neq 0$$

which implies that the matrix $M'(\lambda)$ is non-invertible, *i.e.* $\det(M(\lambda)) = 0$. Then, the solution α in system (4) must be a root of the function:

$$f(\lambda) = \mathrm{Det}(M'(\lambda))$$

which is a polynomial of degree at most $\omega+1$. The polynomial f is not identically zero, because $M'(0)$ is invertible, which implies $f(0) \neq 0$. The polynomial f can easily be obtained from the public-key z and the ciphertext y by computing $f(\lambda) = \mathrm{Det}(M'(\lambda))$ for $\omega + 2$ distinct values of λ and then using Lagrange interpolation.

The factorization of a polynomial over a finite-field can be done in polynomial time (see for example [13]). Therefore, one obtains a list of at most $\omega + 1$ candidates, one of which being the solution α of (4), and equivalently, of (3). For the right candidate α, the vector $y - \alpha \times z$ is equal to $ev(m) + e$, where the weight of e is less than the error correcting capacity of the Reed-Solomon code. Therefore, using Berlekamp-Welsh algorithm, one recovers the plaintext m from $y - \alpha \times z$ in polynomial time.

More precisely, let α, m, e be the solution of (3). Given a solution *V, N, λ* of (5) with $\lambda = \alpha$, we have for all $1 \leq i \leq k + 2 \cdot \omega + 1$:

$$V(x_i) \cdot (m(x_i) + e_i) = N(x_i)$$

Since the error vector e has a weight at most ω, we have for at least $\omega + k + 1$ values of i:

$$V(x_i) \cdot m(x_i) = N(x_i)$$

N and $V \cdot m$ are therefore two polynomials of degree less than $\omega + k - 1$ which take the same value on at least $\omega + k + 1$ distinct points; consequently, the two polynomials must be equal. This means that one can recover m by performing a polynomial division:

$$m = \frac{N}{V}$$

Therefore, one can recover the plaintext in polynomial time.

Let us now consider the second case, *i.e.* *rank* $M(0) < k+2\omega+1$. Then there exists $Y \neq 0$ such that $M(0).Y = 0$. The vector Y gives the coefficients of two polynomials V and N such that for all $1 \leq i \leq n$:

$$V(x_i) \cdot y_i = N(x_i)$$

From (2) we have $y_i = m(x_i) + \alpha \cdot (p(x_i) + E_i) + e_i$, which gives for all i:

$$V(x_i) \cdot \big((m + \alpha \cdot p)(x_i) + \alpha \cdot E_i + e_i\big) = N(x_i)$$

The weight of E is at most W and the weight of e is at most ω. Moreover, from (1) we have $n \geq k + 2\omega + W$. Therefore, for at least $\omega + k$ values of i, we have:

$$V(x_i) \cdot (m + \alpha \cdot p)(x_i) = N(x_i)$$

As previously, $V \cdot (m + \alpha \cdot p)$ and N are two polynomials of degree less than $k+\omega-1$ which take the same value on at least $\omega+k$ distinct points; consequently, they must be equal, which gives:

$$m + \alpha \cdot p = \frac{N}{V}$$

Since the polynomial p is unitary and $\deg p = k - 1$ and $\deg m \leq k - 2$, this enables to recover α. Then, as previously, given α, we recover m in polynomial time[1]. □

4 The Repaired Cryptosystem

In this section, we describe the repaired cryptosystem published in [2]. The new cryptosystem is resistant against the previous attack. The reparation is based on working in the subfield of a given field, and using the trace operator. Following [2], we recall these notions in the next section.

[1] In this second case, we can also recover the private key (p, E). It has been shown in [9] that this second case happens with negligible probability.

4.1 Subfields and Trace Operator

We consider the finite field $\mathrm{GF}(q^u)$, where q is the power of a prime integer. The finite field $\mathrm{GF}(q)$ is a subfield of $\mathrm{GF}(q^u)$. The finite field $\mathrm{GF}(q^u)$ can be viewed as a u-dimensional vector space over $\mathrm{GF}(q)$. Let $\gamma_1, \ldots, \gamma_u$ be a basis of $\mathrm{GF}(q^u)$ over $\mathrm{GF}(q)$, then every element $\alpha \in \mathrm{GF}(q^u)$ can be uniquely written $\alpha = \sum_{i=1}^{u} \alpha_i \gamma_i$, where $\alpha_i \in \mathrm{GF}(q)$.

Definition 2. *The trace operator of $GF(q^u)$ into $GF(q)$ is defined by:*

$$\forall x \in GF(q^u),\ Tr(x) = x + x^q + \ldots + x^{q^{u-1}}$$

The trace operator is a $\mathrm{GF}(q)$-linear mapping (and not $\mathrm{GF}(q^u)$-linear) of $\mathrm{GF}(q^u)$ into $\mathrm{GF}(q)$. For any basis $\gamma_1, \ldots, \gamma_u$ of $\mathrm{GF}(q^u)$, there exists a unique dual basis $\gamma_1^*, \ldots, \gamma_u^*$ with respect to the Trace operator. The dual basis is such that:

$$\mathrm{Tr}(\gamma_i \gamma_j^*) = 1 \text{ if } i = j, \text{ and } 0 \text{ otherwise}$$

The dual basis can be efficiently computed.

We extend the trace operator to vectors:

$$\mathrm{Tr}(c_1, \ldots, c_n) = (\mathrm{Tr}(c_1), \ldots, \mathrm{Tr}(c_n))$$

and to polynomials: for any polynomial $p \in \mathrm{GF}(q^u)[X]$, $p(x) = \sum_{i=0}^{k} p_i x^i$, we define the polynomial $\mathrm{Tr}(p) \in \mathrm{GF}(q)[X]$ as:

$$\mathrm{Tr}(p)(x) = \sum_{i=0}^{k} \mathrm{Tr}(p_i) x^i$$

Let $x_1, \cdots, x_n$ be n distinct elements of $\mathrm{GF}(q) \in \mathrm{GF}(q^u)$. As in section 2.1 we denote by ev the following map:

$$ev : \begin{cases} \mathrm{GF}(q^u)[X] \to \mathrm{GF}(q^u)^n \\ p(X) \qquad\quad \to (p(x_1), \ldots, p(x_n)) \end{cases}$$

Proposition 1. *For all $p \in GF(q^u)[X]$, we have $Tr(ev(p)) = ev(Tr(p))$*

Proof. The j-th component of $\mathrm{Tr}(ev(p))$ is

$$\mathrm{Tr}(p(x_j)) = \mathrm{Tr}(\sum_{i=0}^{k} p_i \cdot (x_j)^i)$$

From the $\mathrm{GF}(q)$-linearity of the Trace operator and the fact that $x_j \in \mathrm{GF}(q)$, we obtain:

$$\mathrm{Tr}(p(x_j)) = \sum_{i=0}^{k} \mathrm{Tr}(p_i)(x_j)^i$$

which is the j-th component of $ev(\mathrm{Tr}(p))$. □

As in section 2.1, we define the Reed-Solomon code of dimension k and length n over $\mathrm{GF}(q^u)$ as the following set of n-tuples (codewords):

$$RS_k = \{ev(f); f \in \mathrm{GF}(q^u)[X], \deg f < k\}$$

4.2 The Repaired Cryptosystem

In this section, we recall the repaired cryptosystem [2].

Parameters: A finite field $\mathrm{GF}(q^u)$, an integer n as the length of the Reed-Solomon code, k its dimension, W is the weight of a large error, ω is the weight of a small error, for which the PR problem with $n - W$ coordinates is easy:

$$\omega \leq \frac{n - W - k}{2} \tag{7}$$

The authors of the repaired cryptosystem recommend in [2] to take $q = 2^{20}$, $u = 4$, $n = 2048$, $k = 1400$, $W = 546$ and $\omega = 49$. [2]

Key Generation: Generate a random polynomial p of degree $k-1$ over $\mathrm{GF}(q^u)$, such that the u coefficients $p_{k-1}, \ldots, p_{k-u}$ form a basis of $\mathrm{GF}(q^u)$over $\mathrm{GF}(q)$. Compute $c = ev(p) \in RS_k$. Generate a random n-dimensional vector E of weight W with coefficients in $\mathrm{GF}(q^u)$. The public-key is the vector $K = c + E$ over $\mathrm{GF}(q^u)$. The private key is (p, E).

Encryption: Let m a message of length $k - u$ over the alphabet $\mathrm{GF}(q)$. The message m is seen as a polynomial $m(X) = m_0 + m_1 X + \ldots + m_{k-1} X^{k-u-1}$ in $\mathrm{GF}(q)[X]$. Generate a random $\alpha \in \mathrm{GF}(q^u)$ and a random vector e of weight ω over $\mathrm{GF}(q)$. The ciphertext y is then:

$$y = ev(m) + \mathrm{Tr}(\alpha \cdot K) + e$$

Decryption: One considers only the positions where $E_i = 0$ and define the shortened code of length $n - W$, which is also a Reed-Solomon code of dimension k, which we denote $\overline{RS}_k$. Let $\overline{y}, \overline{c}, \overline{e}$ be the shortened y, c, e and let $\overline{ev}$ be the shortened map ev. One must solve the equation:

$$\overline{y} = \overline{ev}(m) + \mathrm{Tr}(\alpha \cdot \overline{c}) + \overline{e}$$

Using proposition 1, we have:

$$\mathrm{Tr}(\alpha \cdot \overline{c}) = \mathrm{Tr}(\alpha \cdot \overline{ev}(p)) = \mathrm{Tr}(\overline{ev}(\alpha p)) = \overline{ev}(\mathrm{Tr}(\alpha p))$$

[2] Actually, the authors of [2] forgot to clearly specify k, but they state that with these parameters, "a plaintext consists of $k - u$ elements in $\mathrm{GF}(2^{20})$, that is 27920 bits", from which we infer that $k = 27920/20 + 4 = 1400$

Thus $\overline{ev}(m) + \mathrm{Tr}(\alpha \cdot \bar{c}) = \overline{ev}(m + \mathrm{Tr}(\alpha p)) \in \overline{RS}_k$, and from (7), the weight of the small error $\bar{e}$ is less than the error correction capacity of $\overline{RS}_k$; therefore, using the Berlekamp-Welsh algorithm, one can recover the polynomial $q = m + \mathrm{Tr}(\alpha p)$.

Letting $q = \sum_{i=0}^{k-1} q_i x^i$, since $\deg(m) \leq k - u - 1$, we have $q_i = \mathrm{Tr}(\alpha p_i)$ for $i = k - u, \ldots, k - 1$. This gives the u coordinates of α in the dual basis of $p_{k-u}, \ldots, p_{k-1}$, from which we derive α. From α one recovers m as $m = q - \mathrm{Tr}(\alpha p)$.

5 The Attack against the Repaired Cryptosystem

In this section, we describe an attack that breaks the repaired cryptosystem. Given the public key and a ciphertext, we recover the plaintext in polynomial time. As the attack of section 3, it is a variant of the Berlekamp-Welsh algorithm, but as opposed to the previous attack, it is only a heuristic (but it works very well in practice).

Let $\mathrm{GF}(q^u)$, n, k, W, ω be the parameters of the system. Let (p, E) be the private key and $K = ev(p) + E$ be the public-key. Let m be the plaintext encoded as a polynomial of degree less than $k - u - 1$. Let e be an error vector of weight ω, and $\alpha \in \mathrm{GF}(q^u)$. Let

$$y = ev(m) + \mathrm{Tr}(\alpha \cdot K) + e$$

be the corresponding ciphertext.

Let $\gamma_1, \ldots, \gamma_u$ be a basis of $\mathrm{GF}(q^u)$ over $\mathrm{GF}(q)$. We write $\alpha = \sum_{t=1}^{u} \alpha_t \cdot \gamma_t$ where $\alpha_t \in \mathrm{GF}(q)$. We have

$$\mathrm{Tr}(\alpha \cdot K) = \sum_{t=1}^{u} \alpha_t \mathrm{Tr}(\gamma_t \cdot K)$$

For $t = 1, \ldots, u$, we define:

$$K_t = \mathrm{Tr}(\gamma_t \cdot K)$$

Note that the u vectors K_t are vectors over $\mathrm{GF}(q)$ which can be computed from the public-key K. Finally the ciphertext can be written as:

$$y = ev(m) + \sum_{t=1}^{u} \alpha_t \cdot K_t + e \tag{8}$$

Note that in equation (8), all computation is done in the subfield $\mathrm{GF}(q)$. Let $y_i, K_{t,i}$ and e_i be the components of the vectors y, K_t and e. Given y and K_t, one must solve the following set of equations:

$$\exists e, m, \alpha_1, \ldots, \alpha_u,\ y_i = m(x_i) + \sum_{t=1}^{u} \alpha_t \cdot K_{t,i} + e_i \ \text{ for all } 1 \leq i \leq n \tag{9}$$

where the weight of e is ω. Note that from the definition of the cryptosystem, there is a unique solution.

Let $V, R_1, \ldots, R_u$ be polynomials of degree at most ω, with $V \neq 0$. Let N be a polynomial of degree at most $\omega + k - u - 1$. Consider the following set of equations, where the unknown are the polynomials $V, R_1, \ldots, R_u$ and N:

$$\forall i \in [1, n], \quad V(x_i) \cdot y_i = N(x_i) + \sum_{t=1}^{u} K_{t,i} \cdot R_t(x_i) \tag{10}$$

It is clear that given a solution to system (9), one can obtain a solution to system (10) with $V \neq 0$. Namely, one can take $V(X) = \prod_{i \in B}(X - x_i)$ with $B = \{i | e_i \neq 0\}$, and $R_t = \alpha_t \cdot V$ for $t = 1, \ldots, u$, and $N = m \cdot V$. This shows that the system (10) has at least a non-zero solution.

The system (10) gives a homogeneous linear system of n equations in the $k + (u + 2) \cdot \omega + 1$ unknowns, which are the coefficients of the polynomials $V, R_1, \ldots, R_u$ and N. Let M be the matrix of the corresponding system. The matrix has $k + (u + 2) \cdot \omega + 1$ columns and n rows and can be computed from the ciphertext and the public-key. In the following, we assume that:

$$n \geq k + (u + 2) \cdot \omega \tag{11}$$

This inequality is valid for the proposed parameters. Since the system (10) has at least a non-zero solution, the matrix cannot be of maximum rank, therefore $rank\ M \leq k + (u + 2) \cdot \omega$.

In the following, we assume that $rank\ M = k + (u + 2) \cdot \omega$. This is the only assumption that we make for our cryptanalysis. It seems that in practice, this assumption is always satisfied. In this case, the kernel of M is a linear space of dimension 1. We have already seen that $V(X) = \prod_{i \in B}(X - x_i)$ with $B = \{i | e_i \neq 0\}$, and $R_t = \alpha_t \cdot V$ for $t = 1, \ldots, u$ and $N = m \cdot V$ is a solution to the system (10), and so $(V, R_1, \ldots, R_t, N)$ generates the kernel of M.

Therefore, if we compute by Gaussian elimination an element $(V', R'_1, \ldots, R'_u, N')$ in $\ker M$, we must have that $V' = \lambda \cdot V$, $R'_t = \lambda R_t$ for $t = 1, \ldots, u$ and $N' = \lambda \cdot N$ for some $\lambda \in \mathrm{GF}(q)$ with $\lambda \neq 0$. Therefore, we have $N' = \lambda \cdot N = \lambda \cdot m \cdot V = m \cdot V'$ and we can recover m by doing a polynomial division:

$$m = \frac{N'}{V'}$$

To summarize, assuming that $rank\ M = k + (u + 2) \cdot \omega$, we recover the plaintext from the public-key and the ciphertext in polynomial time.

6 Practical Experiments

In appendix, we illustrate the attack against the original Augot and Finiasz' cryptosystem for small parameters. We have also implemented our attack using

Shoup's NTL library [12]. The attack works well in practice. For the recommended parameters ($n = 1024$, $k = 900$, $\omega = 25$, $W = 74$, $q = 2^{80}$), it takes roughly 30 minutes on a single PC to recover the plaintext from the ciphertext and the public-key. We have also implemented our attack against the repaired cryptosystem, and for the recommended parameters, it takes roughly 8 minutes on a single PC to recover the plaintext from the ciphertext and the public-key.

7 Discussion

In this section, we try to see if it is possible to modify the parameters of the scheme in order to resist to the previous attack. The only condition on the parameters for the attack to work is inequality (11). Therefore, one may try to increase k, u or ω while keeping n constant. In the following, we show that this is not possible. Namely, we describe another attack on the repaired cryptosystem that recovers the private-key from the public-key. The attack does not work for the recommended parameters, but applies for large u.

The attack is the following. Let $K = ev(p) + E$ be the public-key with the n components K_i, where $\deg p = k - 1$ and the weight of E is W. The Berlekamp-Welsh algorithm for recovering p from K is the following: it looks for two polynomials V and N such that $\deg V = W$, $\deg N = k + W - 1$ and $V \neq 0$, such that:

$$\forall i \in [1, n], V(x_i) \cdot K_i = N(x_i)$$

This gives a homogeneous linear system of n equations in $k + 2 \cdot W + 1$ unknown. This system has a non-zero solution as we can take $V(X) = \prod_{i \in B}(X - x_i)$ with $B = \{i | E_i \neq 0\}$ and $N = p \cdot V$. Letting V, N be any non-zero solution, we have for at least $n - W$ values of i:

$$V(x_i) \cdot p(x_i) = N(x_i)$$

Therefore, if $n - W > k + W - 1$, or equivalently,

$$n \geq k + 2 \cdot W \tag{12}$$

the polynomials $V \cdot p$ and N must be equal, which enables to recover p as $p = N/V$.

As in the attack of section 5, from K we derive the u vectors K_t for $t = 1, \ldots, u$ such that:

$$K_t = \mathrm{Tr}(\gamma_t \cdot K)$$

where $\gamma_1, \ldots, \gamma_u$ is a basis of $\mathrm{GF}(q^u)$ over $\mathrm{GF}(q)$. Then we have:

$$K_t = \mathrm{Tr}(\gamma_t \cdot (ev(p) + E)) = ev(\mathrm{Tr}(\gamma_t \cdot p)) + \mathrm{Tr}(\gamma_t \cdot E)$$

Letting $p_t = \mathrm{Tr}(\gamma_t \cdot p)$ and $E_t = \mathrm{Tr}(\gamma_t \cdot E)$, we can write:

$$\forall t \in [1, u], K_t = ev(p_t) + E_t$$

Therefore, we obtain a set of u vectors K_t which are evaluation of a polynomial p_t plus some error E_t. Thus we obtain u instances of the polynomial reconstruction problem over $\mathrm{GF}(q)$.

The key observation is that the instances are not independent because the errors occur is the same positions in all vectors E_t. This enables to derive the following improved attack: we look for a polynomial $V \neq 0$, $\deg V \leq W$ and polynomials $N_1, \ldots, N_u$, $\deg N_t \leq k + W - 1$ such that:

$$\forall i \in [1, n], \begin{cases} V(x_i) \cdot K_{1,i} = N_1(x_i) \\ \cdots \\ V(x_i) \cdot K_{u,i} = N_u(x_i) \end{cases}$$

We can take the same polynomial V for each $t \in [1, u]$ because the errors are in the same positions for all E_t. This gives a system of $u \cdot n$ equations in the $u \cdot k + (u+1) \cdot W + 1$ unknowns. Let M be the corresponding matrix. It has $u \cdot n$ rows and $u \cdot k + (u+1) \cdot W + 1$ columns. We assume that:

$$u \cdot n \geq u \cdot k + (u+1) \cdot W \tag{13}$$

The system has a non-zero solution. Therefore, the matrix cannot be of maximum rank, therefore *rank* $M \leq u \cdot k + (u+1) \cdot W$. In the following, we assume that *rank* $M = u \cdot k + (u+1) \cdot W$. This makes our attack heuristic, but the heuristic works well in practice. In this case, as in section 5, the kernel of M is a linear space of dimension one, and given a solution $(V, N_1, \ldots, N_u)$, one can recover the polynomials p_t as $p_t = N_t / V$ and then recover the private key (p, E). A similar approach was already used in [4] for the decoding of interleaved Reed-Solomon codes.

The inequality (13) gives the following condition for the attack to work:

$$n \geq k + \frac{u+1}{u} \cdot W$$

which is an improvement over (12). Note that for the recommended parameters in [2], the attack does not apply. Therefore, to prevent this attack, one must have:

$$n < k + \frac{u+1}{u} \cdot W \tag{14}$$

Then, combining inequality (14) with inequality (7) which is necessary to be able to decrypt, one must have:

$$n \geq k + 2 \cdot (u+1) \cdot \omega$$

which shows that condition (11) of the attack of section 5 is always satisfied. Therefore, there is no set of parameters which makes the repaired cryptosystem secure against both attacks.

8 Conclusion

We have broken the cryptosystem published by Augot and Finiasz at Eurocrypt 2003 and its reparation in [2]. In both cases, our attack recover the plaintext from the ciphertext and the public-key in polynomial time. Moreover, both attack work well in practice, as for the recommended parameters, one recovers the plaintext in a few minutes on a single PC.

References

1. D. Augot and M. Finiasz, *A Public Key encryption scheme based on the Polynomial Reconstruction problem,* Proceedings of Eurocrypt 2003, LNCS vol. 2656, Springer-Verlag.
2. D. Augot, M. Finiasz and P. Loidreau, Using the Trace Operator to repair the Polynomial Reconstruction based Cryptosystem presented at Eurocrypt 2003, Cryptology ePrint Archive, Report 2003/209, 30 Sep 2003, http://eprint.iacr.org/.
3. E.R. Berlekamp and L.R. Welch, *Error correction* for *algebraic block codes.* US Patent 4 633 470, 1986.
4. D. Bleichenbacher, A. Kiayias and M. Yung, *Decoding of Interleaved Reed Solomon Codes over Noisy Data,* proceedings of ICALP 2003.
5. J.S. Coron, *Cryptanalysis of a public-key encryption scheme based on the polynomial reconstruction problem,* Cryptology ePrint Archive, Report 2003/036, 5 Mar 2003, http://eprint.iacr.org/.
6. P. Gemmell and M. Sudan, *Highly resilient correctors for multivariate polynomials,* Information Processing Letters, 43(4): 169–174, September 1992.
7. V. Guruswami and M. Sudan, *Improved decoding of Reed-Solomon and Algebraic-Geometric codes,* IEEE Transactions on Information Theory, 45 : 1757-1767, 1999.
8. A. Kiayias and M. Yung, *Cryptographic hardness based on the decoding of Reed-Solomon codes with applications,* Proceedings of ICALP 2002, LNCS 2380, pp 232-243, 2002.
9. A. Kiayias and M. Yung, *Cryptanalysis of the polynomial reconstruction based public-key cryptosystem of Eurocrypt 2003 in the optimal parameter setting,* available at http://www.cse.uconn.edu/~ akiayias/.
10. M. Naor and B. Pinkas, *Oblivious transfer and polynomial evaluation.* In ACM, editor, STOC 99, pp 245-254, 1999.
11. I.S. Reed and G. Solomon, *Polynomial codes over certain finite fields,* J. SIAM, 8:300-304, 1960.
12. V. Shoup, *NTL: A Library for doing Number Theory (version 5.3.1),* publicly available at www.shoup.net.
13. V. Shoup, *A fast deterministic algorithm for factoring polynomials over finite fields of small characteristic,* in Proc. 1991 International Symposium on Symbolic and Algebraic Computation, pp. 14-21, 1991.

A A Toy Example

In this section we illustrate the attack for small parameters. We take $n = 8$, $k = 3$, $\omega = 1$, $W = 3$. We work modulo $q = 11$. We take $x_i = i$ for $i = 1, \ldots, 8$. We take:

$$p(x) = x^2 + 5x + 3$$
$$E = (0, 0, 4, 0, 7, 6, 0, 0)$$

for the private key. The public-key is:

$$z = ev(p) + E = (9, 6, 9, 6, 5, 9, 10, 8)$$

Let the message m be $m(x) = 8x + 2$. Let $\alpha = 7$ and $e = (0, 5, 0, 0, 0, 0, 0, 0)$. The ciphertext y is:

$$y = ev(m) + \alpha \times z + e = (7, 10, 1, 10, 0, 3, 7, 1)$$

The matrix $M(\lambda)$ is then:

$$M(\lambda) = \begin{bmatrix} 7-9\lambda & 7-9\lambda & 10 & 10 & 10 & 10 \\ 10-6\lambda & 9-\lambda & 10 & 9 & 7 & 3 \\ 1-9\lambda & 3-5\lambda & 10 & 8 & 2 & 6 \\ 10-6\lambda & 7-2\lambda & 10 & 7 & 6 & 2 \\ -5\lambda & -3\lambda & 10 & 6 & 8 & 7 \\ 3-9\lambda & 7-10\lambda & 10 & 5 & 8 & 4 \\ 7-10\lambda & 5-4\lambda & 10 & 4 & 6 & 9 \\ 1-8\lambda & 8-9\lambda & 10 & 3 & 2 & 5 \end{bmatrix}$$

The determinant $f(\lambda)$ of the matrix $M'(\lambda)$ obtained by taking the first 6 lines of $M(\lambda)$ is equal to:

$$f(\lambda) = \det M'(\lambda) = 3\lambda^2 + 5\lambda + 5$$

which factors modulo $q = 11$ into:

$$f(\lambda) = 3 \cdot (\lambda - 6) \cdot (\lambda - 7)$$

For $\lambda = 7$, the matrix $M'(7)$ is non-invertible. We solve the linear system and find that $Y = (8, 7, 5, 1, 1, 0)$ is such that $M(7).Y = 0$; this gives $V(x) = 7x + 8$ and $N(x) = x^2 + x + 5$, which gives modulo $q = 11$:

$$m(x) = N(x)/V(x) = 8x + 2$$

Faster Scalar Multiplication on Koblitz Curves Combining Point Halving with the Frobenius Endomorphism

Roberto Maria Avanzi[1] *, Mathieu Ciet[2] *, and Francesco Sica[3] *

[1] IEM, University of Duisburg-Essen, Essen, Germany
mocenigo@exp-math.uni-essen.de
[2] Innova Card, La Ciotat, France
mathieu.ciet@innova-card.com
[3] Dept. of Mathematics and Computer Science, Mount Allison University, Canada
fsica@mta.ca

Dedicated to Preda Mihăilescu on occasion of the birth of his daughter Seraina.

Abstract. Let E be an elliptic curve defined over $\mathbb{F}_{2^n}$. The inverse operation of point doubling, called point halving, can be done up to three times as fast as doubling. Some authors have therefore proposed to perform a scalar multiplication by an "halve-and-add" algorithm, which is faster than the classical double-and-add method.
If the coefficients of the equation defining the curve lie in a small subfield of $\mathbb{F}_{2^n}$, one can use the Frobenius endomorphism τ of the field extension to replace doublings. Since the cost of τ is negligible if normal bases are used, the scalar multiplication is written in "base τ" and the resulting "τ-and-add" algorithm gives very good performance.
For elliptic Koblitz curves, this work combines the two ideas for the first time to achieve a novel decomposition of the scalar. This gives a new scalar multiplication algorithm which is up to 14.29% faster than the Frobenius method, *without* any additional precomputation.

Keywords. Koblitz curves, scalar multiplication, point halving, τ-adic expansion, integer decomposition.

1 Introduction

In 1985 Miller [9] and Koblitz [7] independently proposed to use the group of rational points of an elliptic curve over a finite field to create cryptosystems based on the discrete logarithm problem (DLP).

* The European Commission supported the research of the first author under Contract IST-2001-32613 (AREHCC), and the research of the second and third authors under Contract IST-1999-12324 (NESSIE). This research began when the second and third authors were at the UCL Crypto Group, Louvain-la-Neuve, Belgium. The third author's stay at the IEM was supported by the DFG, Graduiertenkolleg 647 *Crypto*.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 28–40, 2004.

The basic operation of a DLP-based cryptosystem is the *scalar multiplication,* i.e. given a point $\boldsymbol{P}$ and an integer s, to compute $s\boldsymbol{P}$. Some families of elliptic curves have arithmetic properties useful for speeding up this operation. One such family consists of the *Koblitz curves:* These curves, first proposed by Koblitz [8] and called *anomalous binary curves* by Solinas in [14], are defined over $\mathbb{F}_{2^n}$ by equations of the form

$$E_a \ : \ y^2 + xy = x^3 + ax^2 + 1 \quad \text{with} \quad a \in \{0, 1\} \ . \tag{1}$$

The present paper is devoted to scalar multiplication on Koblitz curves. We restrict our attention to those curves for which n is prime, and whose rational point group contains a (unique) subgroup of large prime order p with a cofactor at most 4, such as those in the standards [17,18].

Let τ denote the Frobenius endomorphism $\tau(x, y) = (x^2, y^2)$ and $\boldsymbol{P}$ be a point of order p on E_a. As τ commutes with point addition, $\tau(\boldsymbol{P})$ also has order p, and there exists a scalar λ with $\tau(\boldsymbol{P}) = \lambda\boldsymbol{P}$. This suggests that τ may be used to compute multiples of $\boldsymbol{P}$. In fact, we can write a "τ-adic expansion associated to the scalar s", i.e. an expression of the form $\sum_{i=0}^{m} s_i\tau^i$, with $s_i \in \{0, \pm 1\}$, such that $\sum_{i=0}^{m} s_i\tau^i(\boldsymbol{P}) = s\boldsymbol{P}$ for all $\boldsymbol{P} \in E_a(\mathbb{F}_{2^n})$. Then a "$\tau$-and-add" loop is used to compute $s\boldsymbol{P}$. Since τ is much faster than a point doubling, the resulting method is very efficient.

Knudsen [5] and Schroeppel [12] independently proposed a technique for elliptic curves over binary fields based on *point halving.* This method computes the multiple $\boldsymbol{R}$ of any point $\boldsymbol{P}$ of odd order such that $2\boldsymbol{R} = \boldsymbol{P}$ and $\boldsymbol{R} \in \langle \boldsymbol{P} \rangle$. Since for curves of order $2p$ point halving is up to three times as fast as doubling, it is possible to improve performance of scalar multiplication by expanding the scalar using "powers of 1/2" and replacing the double-and-add algorithm with a halve-and-add method.

In our paper, we combine for the first time the τ-NAF approach with a single point halving, thereby reducing the amount of point additions from $n/3$ to $2n/7$, and providing an asymptotic speed-up of about 14.29%. *The idea is that it is possible, using a single point halving, to replace some sequences of a τ-NAF having density* 1/2 *(and containing at least three non-zero coefficients) with sequences having weight* 2.

In the next section we collect some basic facts about τ-NAFs and point halving. In Section 3, we describe our new scalar decomposition, prove its correctness, and apply it to the computation of scalar multiplications. The complexity analysis is given in Section 4. In Section 5 we conclude.

Acknowledgements. The authors express their gratitude to Darrel Hankerson, Tanja Lange, Nicolas Thériault and to the anonymous referees for the many useful suggestions for improving the paper. The authors also thank Jean-Jacques Quisquater for fruitful discussions and support.

2 Background Concepts

2.1 τ Non Adjacent Forms

All facts here are stated without proofs: These are found in [14,15].

Let the Koblitz curve E_a defined over $\mathbb{F}_{2^n}$ by equation (1) have a (unique) subgroup G of large prime order p with a cofactor at most 4. Let τ denote the Frobenius endomorphism. It is easy to see that for each point $\boldsymbol{P}$ we have $(\tau^2 + 2)\boldsymbol{P} = \mu\tau(\boldsymbol{P})$ where $\mu = (-1)^{1-a}$, i.e.

$$\tau^2 + 2 = \mu\tau \ . \tag{2}$$

If τ is identified with a complex root of equation (2), say $\tau = (\mu + \sqrt{-7})/2$, we can view $\tau(\boldsymbol{P})$ as *multiplication by* τ and let $\mathbb{Z}[\tau]$ operate on $\boldsymbol{P}$.

The τ-adic non-adjacent form (τ-NAF for short) of an integer $z \in \mathbb{Z}[\tau]$ is a decomposition $z = \sum_i z_i \tau^i$ where $z_i \in \{0, \pm 1\}$ with the *non-adjacency* property $z_j z_{j+1} = 0$, similarly to the classical NAF [11]. The average *density* (that is the average ratio of non-zero bits related to the total number of bits) of a τ-NAF is 1/3. Each integer z admits a unique τ-NAF. The length of the τ-NAF expansion of a randomly chosen scalar is $\approx 2n$, whereas the bit length of is $\approx n$. But, for any point $\boldsymbol{P} \in E_a(\mathbb{F}_{2^n}) \setminus E_a(\mathbb{F}_2)$, $\tau^n \boldsymbol{P} = \boldsymbol{P}$ and $\tau\boldsymbol{P} \neq \boldsymbol{P}$. Since $\mathbb{Z}[\tau]$ is an Euclidian ring we can take the remainder of s mod $(\tau^n - 1)/(\tau - 1)$ and use it in place of s. This remainder will have smaller norm than that of $(\tau^n - 1)/(\tau - 1)$, and thus it will have length at most n. Its τ-NAF is called the *reduced* τ-NAF of s.

The computation of an element of $\mathbb{Z}[\tau]$ of minimal norm which is congruent to s modulo $(\tau^n - 1)/(\tau - 1)$ is a very slow operation. To overcome this problem, Solinas proposes to compute an element which is *almost* of minimal norm and whose computation is much faster. The length of its τ-NAF (the *partially reduced* τ-NAF of s) is at most $n + a + 3$. The corresponding τ-and-add algorithm runs marginally slower than with the reduced τ-NAF of the scalar, but the overall speed-up is significant.

2.2 Point Halving

Let E be a generic elliptic curve over $\mathbb{F}_{2^n}$ by an equation of the form

$$E \ : \ y^2 + xy = x^3 + ax^2 + b$$

with $a, b \in \mathbb{F}_{2^n}$ (hence, not necessarily a Koblitz curve) and having a subgroup $G \leq E(\mathbb{F}_{2^n})$ of large prime order. To a point $\boldsymbol{P}$ with affine coordinates (x, y) we associate the quantity $\lambda_{\boldsymbol{P}} = x + \frac{y}{x}$. Let $\boldsymbol{P} = (x, y)$ and $\boldsymbol{R} = (u, v)$ be points of $E(\mathbb{F}_{2^n}) \setminus \{0\}$ with $2\boldsymbol{R} = \boldsymbol{P}$. The affine coordinates of $\boldsymbol{P}$ and $\boldsymbol{R}$ are related as follows:

$$\lambda_{\boldsymbol{R}} = u + \frac{v}{u} \tag{3}$$

$$x = \lambda_{\boldsymbol{R}}^2 + \lambda_{\boldsymbol{R}} + a \tag{4}$$

$$y = u^2 + x(\lambda_{\boldsymbol{R}} + 1) \tag{5}$$

Given $\boldsymbol{P}$, point halving consists in finding $\boldsymbol{R}$. To do this, we have to solve (4) for λ, (5) for u, and finally (3) for v. After some simple manipulations, we see that we have to perform the following operations:

$$\begin{aligned} &\text{(i)} && \text{Solve } \lambda_{\boldsymbol{R}}^2 + \lambda_{\boldsymbol{R}} = a + x \text{ for } \lambda_{\boldsymbol{R}} && (6)\\ &\text{(ii)} && \text{Put } t = y + x(\lambda_{\boldsymbol{R}} + 1) && \\ &\text{(iii)} && \text{Find } u \text{ with } u^2 = t && (7)\\ &\text{(iv)} && \text{Put } v = t + u\lambda_{\boldsymbol{R}} \ . && \end{aligned}$$

Knudsen [5] and Schroeppel [12,13] show how to perform the necessary steps in an efficient way. A more thorough analysis of the costs of these steps is given in [3]. We shall return to this matter in Section 4.

Point halving is an automorphism of G. So, given a point $\boldsymbol{P} \in G$, there is a unique $\boldsymbol{R} \in G$ such that $2\boldsymbol{R} = \boldsymbol{P}$. In other words, the equations (6) and (7) can always be solved in $\mathbb{F}_{2^n}$. But, they do not determine a *unique* point $\boldsymbol{R}$ with $2\boldsymbol{R} = \boldsymbol{P}$. In fact, solving them will always yield two distinct points $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ such that $\boldsymbol{R}_1 - \boldsymbol{R}_2$ is the unique point of order 2 of the curve. It is possible, by performing an additional check, to determine the point $\boldsymbol{R} \in G$, but we do not need that in our applications. We refer the interested reader to [5,12,13] of [3] for details.

3 New Scalar Decomposition and Scalar Multiplication

Consider a Koblitz curve E_a and adopt the notation of Subsection 2.1. Equation (2) implies that $\tau^3 + 2\tau = \mu\tau^2 = \mu(\mu\tau - 2) = \tau - 2\mu$, hence

$$2 = -\mu\big(1 + \tau^2\big)\tau \ . \tag{8}$$

In particular, this means that we can compute $2\boldsymbol{P}$ as $-\mu\big(1+\tau^2\big)\tau\boldsymbol{P}$. This alone is not very useful, since it replaces a point doubling with one addition and three Frobenius operations. However, these relations become interesting if we can make repeated use of them:

Lemma 1. *Let* $\boldsymbol{P} = 2\boldsymbol{R}$. *Put* $\boldsymbol{Q} = \tau\boldsymbol{R}$. *The following equalities hold:*

$$\left(\sum_{j=0}^{k-1}(-1)^j\tau^{2j}\right)\boldsymbol{P} = -\mu(1 + (-1)^{k-1}\tau^{2k})\boldsymbol{Q}, \qquad \textbf{(I)}$$

$$\left(\sum_{j=0}^{k-2}(-1)^j\tau^{2j}\right)\boldsymbol{P} + (-1)^{k-2}\tau^{2(k-1)}\boldsymbol{P} = (-\mu + (-1)^{k-1}\tau^{2k-1})\boldsymbol{Q}, \qquad \textbf{(II)}$$

$$\left(\sum_{j=0}^{k-3}(-1)^j\tau^{2j}\right)\boldsymbol{P} + (-1)^{k-3}\big(\tau^{2(k-2)} + \tau^{2(k-1)}\big)\boldsymbol{P} = (-\mu + (-1)^{k-3}\tau^{2k-3})\boldsymbol{Q}. \qquad \textbf{(III)}$$

Proof. The first statement is simplified using (8), giving a telescopic sum

$$\sum_{j=0}^{k-1}(-1)^j\tau^{2j}\boldsymbol{P} = -\mu\sum_{j=0}^{k-1}(-1)^j\tau^{2j}(1+\tau^2)\boldsymbol{Q} = -\mu\big(1 + (-1)^{k-1}\tau^{2k}\big)\boldsymbol{Q} \ .$$

To prove the second equality we use the previous relation (with $k-1$ in place of k) in combination with the fact that $\boldsymbol{P} = (\mu - \tau)\boldsymbol{Q}$:

$$\begin{aligned}\left(\sum_{j=0}^{k-2}(-1)^j\tau^{2j}\right)\boldsymbol{P} + (-1)^{k-2}\tau^{2(k-1)}\boldsymbol{P} &= \\ = -\mu\big(1 + (-1)^{k-2}\tau^{2(k-1)}\big)\boldsymbol{Q} + (-1)^{k-2}\tau^{2(k-1)}(\mu-\tau)\boldsymbol{Q} &\\ = (-\mu + (-1)^{k-1}\tau^{2k-1})\boldsymbol{Q}\ . &\end{aligned}$$

The verification of the third equality proceeds in a similar fashion:

$$\begin{aligned}&\left(\sum_{j=0}^{k-3}(-1)^j\tau^{2j}\right)\boldsymbol{P} + (-1)^{k-3}\big(\tau^{2(k-2)} + \tau^{2(k-1)}\big)\boldsymbol{P} = \\ &= \big(-\mu + (-1)^{k-2}\tau^{2k-3}\big)\boldsymbol{Q} + (-1)^{k-3}\tau^{2(k-1)}(\mu-\tau)\boldsymbol{Q} \\ &= \big(-\mu + (-1)^{k-2}\tau^{2k-3}(1 - \mu\tau + \tau^2)\big)\boldsymbol{Q} = \big(-\mu + (-1)^{k-3}\tau^{2k-3}\big)\boldsymbol{Q}\ .\ \square\end{aligned}$$

We need more terminology and notation to describe and analyze our recoding.

Notation. *We write $\mathcal{S} = \langle s_n \dots s_j s_{j-1} \dots s_1 s_0\rangle$ for any τ-adic expansion (also called string) $\sum_{0\le j\le n} s_j\tau^j$. We call $\#\mathcal{S} = n$ the* length *of the expansion $\mathcal{S}$. Also by $\mathcal{S}[i \dots j]$ we denote the sub-expansion $\langle s_i \dots s_j\rangle$ of $\mathcal{S}$. Occasionally, we will encounter the string $x \times \langle s_i \dots s_j\rangle$, where $x = \pm 1$. It is then understood that $-1 \times \langle s_i \dots s_j\rangle = \langle -s_i \dots -s_j\rangle$ is the bitwise complement of the original string. Henceforth $\mathcal{S}$ will denote the τ-NAF expansion of any integer, namely an expansion as above with $s_j = 0, \pm 1$ and $s_j s_{j+1} = 0$. We write $\bar{1}$ for -1, and also $\bar{1}^t$ for $(-1)^t$.*

Definition 1. *Let $\mathcal{K} = \langle \star 0 \star \dots \star 0 \star\rangle$ be a substring of a τ-NAF expansion $\mathcal{S}$, where the symbol $\star$ denotes a 1 or a -1. $\mathcal{K}$ is a* k-block *if it contains k elements $\star$, i.e. it is of length $2k-1$. A k-block is* maximal *if the two digits preceding it and the two following it are all zero.*

Example 1. We highlight a few examples of k-blocks in a sequence

$$\langle 1\,0\,0\,\overbrace{1\,0\,1\,0\,1}^{\text{2-block}}\,0\,0\,0\,1\,0\,0\,\underbrace{\bar{1}\,0\,\overbrace{\bar{1}\,0\,\bar{1}\,0\,1}^{\text{3-block}}}_{\substack{\text{(maximal)}\\ \text{4-block}}}\,0\,0\,\bar{1}\rangle\ .$$
$$\underbrace{}_{\substack{\text{(maximal)}\\ \text{3-block}}}$$

We now give a practical application of Lemma 1.

Remark 1. Let s be an integer and $\boldsymbol{P}$ a point of odd order on a Koblitz curve. Let $\mathcal{S} = \langle s_{\ell-1} \dots s_j s_{j-1} \dots s_1 s_0\rangle$ be the τ-NAF associated to s, so that $s\boldsymbol{P} = \sum_{j=0}^{\ell-1} s_j\tau^j(\boldsymbol{P})$. By Lemma 1, the multiples of $\boldsymbol{P}$ corresponding to some special k-blocks appearing in $\mathcal{S}$ can be computed as suitable multiples of $\boldsymbol{Q} := \tau\big(\frac{1}{2}\boldsymbol{P}\big)$ by a τ-and-add method involving fewer group additions. The situation, in terms

of substrings of τ-adic expansions, is the following (where all blocks on the left-hand side are k-blocks).

$$\langle \underbrace{\bar{1}^{k-1}0\bar{1}^{k-2}0 \quad \ldots \quad 010\bar{1}01}_{\text{length } 2k-1}\rangle \boldsymbol{P} = \bar{\mu}\langle \underbrace{\bar{1}^{k-1}00 \ \ldots \ 001}_{\text{length } 2k+1}\rangle \boldsymbol{Q} \qquad \textbf{(I)}$$

$$\langle \underbrace{\bar{1}^{k-2}0\bar{1}^{k-2}0\bar{1}^{k-3}0 \quad \ldots \quad 010\bar{1}01}_{\text{length } 2k-1}\rangle \boldsymbol{P} = \langle \underbrace{\bar{1}^{k-1}00\ldots00\bar{\mu}}_{\text{length } 2k}\rangle \boldsymbol{Q} \qquad \textbf{(II)}$$

$$\langle \underbrace{\bar{1}^{k-3}0\bar{1}^{k-3}0\bar{1}^{k-3}0\bar{1}^{k-4}0 \ \ldots 010\bar{1}01}_{\text{length } 2k-1}\rangle \boldsymbol{P} = \langle \underbrace{\bar{1}^{k-3}00\ldots0\bar{\mu}}_{\text{length } 2k-2}\rangle \boldsymbol{Q}. \qquad \textbf{(III)}$$

Definition 2. *We call the k-blocks of the above three types together with their opposites in sign* good k*-blocks. A* maximal *good k-block is a good k-block which cannot be further extended at its sides.*

Remark 1 suggests a strategy for saving operations in the computation of $s\boldsymbol{P}$. From the τ-NAF $\mathcal{S}$ of s, we create *two* τ-adic expansions, $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$, by repeated replacements of subsequences, where:

1. $\mathcal{S}^{(1)}$ is obtained from $\mathcal{S}$ by discarding the maximal good k-blocks for $k \geq 3$, substituting them with a string of $2k-1$ zeros;
2. $\mathcal{S}^{(2)}$ consists of the weight two right-hand sequences replacing the maximal good k-blocks removed from $\mathcal{S}$, each at the same position where the original k-block was in $\mathcal{S}$, according to **I**, **II** or **III.**

It is clear from Lemma 1 and Remark 1 that $s\boldsymbol{P} = \mathcal{S}^{(1)}\boldsymbol{P} + \mathcal{S}^{(2)}\boldsymbol{Q}$.

Remark 2. It is easy to verify that no two k-block replacements overlap. For k-blocks of types **II** and **III** this is obvious. Since a maximal k-block of type **I** is followed by at least *two* zero bits (otherwise it would not be maximal), the next non-zero bit may only occur after the end of the replacement block. $\mathcal{S}^{(2)}$ need not satisfy the non-adjacency property.

We have written down explicity the algorithm which generates $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$ as Algorithm 1. Note that the length of $\mathcal{S}^{(1)}$ is equal to the length of $\mathcal{S}$ and that of $\mathcal{S}^{(2)}$ is at most the length of $\mathcal{S}$ plus two.

The total number of non-zero coefficients in $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$ is, by construction, no greater than that of $\mathcal{S}$. In fact, the number of non-zero coefficients decreases considerably on average (see Section 4). We now see how to use the new recoding to perform a scalar multiplication.

3.1 Field Represented Using a Normal Basis

If n is prime, then a normal basis for $\mathbb{F}_{2^n}$ exists and it is easy to construct [1]. Squaring an element of the field consists in a circular shift of the bits of the internal representation of its argument. The same holds for the inverse operation,

Input: A Koblitz curve E_a with corresponding parameter $\mu = (-1)^{1-a}$, a point $\boldsymbol{P}$ of odd order on E_a and a scalar s with associated (partially) reduced τ-NAF $\mathcal{S}$
Output: Two τ-adic expansions $\mathcal{S}^{(j)} = \sum_i s_i^{(j)} \tau^i$, $j = 1, 2$ such that $s\boldsymbol{P} = \mathcal{S}^{(1)}\boldsymbol{P} + \mathcal{S}^{(2)}\boldsymbol{Q}$, where $\boldsymbol{Q} = \tau\left(\frac{1}{2}\boldsymbol{P}\right)$

$\mathcal{S}^{(1)} \leftarrow \mathcal{S}$, $\mathcal{S}^{(2)} \leftarrow \langle 0 \dots 0 \rangle$ with $\#\mathcal{S}^{(2)} = \#\mathcal{S} + 2$, and $i \leftarrow 0$
DO {
  $x \leftarrow s_i$
  If $x = 0$ then { $i \leftarrow i + 1$ }
  else {
    Let $k \geq 1$ be the largest integer such that:
      $\mathcal{S}[i + 2(k-1) \dots i] = x \times \langle \bar{1} 0^{k-1} \bar{1} 0^{k-2} 0 \dots \bar{1} 0 1 \rangle$
    type $\leftarrow$ **I**
    If $s_{i+2k} = s_{i+2(k-1)}$ then { $k \leftarrow k + 1$ and type $\leftarrow$ **II** ,
      If $s_{i+2k} = s_{i+2(k-1)}$ then { $k \leftarrow k + 1$ and type $\leftarrow$ **III** } }
    (Observe that $s_{i+2k-1} = 0$)
    If $k \geq 3$ then {
      $\mathcal{S}^{(1)}[i + 2(k-1) \dots i] \leftarrow \langle 0 \dots 0 \rangle$
      If type = **I** then { $s^{(2)}_{i+2k} \leftarrow (-1)^k \mu x$ and $s^{(2)}_i \leftarrow -\mu x$ }
      If type = **II** then { $s^{(2)}_{i+2k-1} \leftarrow (-1)^{k-1} x$ and $s^{(2)}_i \leftarrow -\mu x$ }
      If type = **III** then { $s^{(2)}_{i+2k-3} \leftarrow (-1)^{k-3} x$ and $s^{(2)}_i \leftarrow -\mu x$ }
    }
    $i \leftarrow i + 2k$
  }
} WHILE $i \leq \#\mathcal{S}$
Output $(\mathcal{S}^{(1)}, \mathcal{S}^{(2)})$.

Algorithm 1. New τ-adic scalar recoding

the extraction of a square root. Therefore, τ, and its inverse, have the same minimal cost.

To compute $\mathcal{S}^{(1)}\boldsymbol{P} + \mathcal{S}^{(2)}\boldsymbol{Q}$, it is not necessary to precompute $\boldsymbol{Q}$: We can first compute $\mathcal{S}^{(2)}\boldsymbol{P}$, halve the result, apply a suitable power of τ, and then resume the τ-and-add loop using $\mathcal{S}^{(1)}$, thus avoiding an extra point storage. We give a realization of this idea which processes the τ-adic expansions right-to-left (i.e. beginning with the lowest powers of τ) and using τ^{-1} instead of τ. In Remark 3 we will see how this allows to interleave our recoding of $\mathcal{S}$ into $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$ with the scalar multiplication.

We begin by computing $\mathcal{S}^{(2)}\boldsymbol{P}$. We first set a variable $\boldsymbol{X}$ to $s_0^{(2)}\boldsymbol{P}$. For each $j = 1, 2, \dots, \ell_2 - 1$ with $\ell_2 = \#\mathcal{S}^{(2)}$ we apply τ^{-1} to $\boldsymbol{X}$ and add $s_j^{(2)}\boldsymbol{P}$. After these steps $\boldsymbol{X}$ equals $\tau^{-\ell_2+1}\mathcal{S}^{(2)}\boldsymbol{P}$ because we used the exponentiation algorithm from *right to left* with τ^{-1} instead of τ, so we apply τ^{ℓ_2-1-n} to get the correct result. (We use the fact that $\tau^n - 1$ is 0 on E.) We then replace $\boldsymbol{X}$ with $\tau\left(\frac{1}{2}\boldsymbol{X}\right)$ and repeat the above procedure with $\mathcal{S}^{(1)}$ in place of $\mathcal{S}^{(2)}$, starting from $\boldsymbol{X} + s_0^{(1)}\boldsymbol{P}$. We have thus Algorithm 2.

Remark 3. Once the τ-NAF $\mathcal{S}$ is given, there is no need to store $\mathcal{S}^{(j)}$ for $j = 1, 2$. The generation of $\mathcal{S}^{(j)}$ for $j = 1, 2$ can be done twice and online, during the run

Input: A Koblitz curve E_a with corresponding parameter $\mu = (-1)^{1-a}$, a point $\boldsymbol{P}$ of odd order on E_a and a scalar s with associated (partially) reduced τ-NAF $\mathcal{S}$
Output: $s\boldsymbol{P}$

Compute the two τ-adic expansions
$\mathcal{S}^{(j)} = \sum_{i=0}^{\ell_{j-1}} s_i^{(j)} \tau^i$ for $j = 1, 2$
from $\mathcal{S}$ using Algorithm 1
(If $\mathcal{S}$ is the reduced τ-NAF of s then $\#\mathcal{S}$ and $\ell_1 \leq n$.
If $\mathcal{S}$ is partially reduced then $\#\mathcal{S}$, $\ell_1 \leq n + a + 3$.
ℓ_2 is at most $\#\mathcal{S} + 2$.)
$\boldsymbol{X} \leftarrow s_0^{(2)} \boldsymbol{P}$
for $j = 1$ to $\ell_2 - 1$ do
{ $\boldsymbol{X} \leftarrow \tau^{-1}\boldsymbol{X}$, and $\boldsymbol{X} \leftarrow \boldsymbol{X} + s_j^{(2)}\boldsymbol{P}$ }
(Now $\boldsymbol{X} = \tau^{-\ell_2+1}\mathcal{S}^{(2)}\boldsymbol{P}$)
$\boldsymbol{X} \leftarrow \tau^{\ell_2-n}\boldsymbol{X}$, $\boldsymbol{X} \leftarrow \frac{1}{2}\boldsymbol{X}$
(Here we simplified $\boldsymbol{X} \leftarrow \tau^{\ell_2-1-n}\boldsymbol{X}$, $\boldsymbol{X} \leftarrow \tau(\frac{1}{2}\boldsymbol{X})$.
Now $\boldsymbol{X} = \mathcal{S}^{(2)}\tau(\frac{1}{2}\boldsymbol{P})$.)
$\boldsymbol{X} \leftarrow \boldsymbol{X} + s_0^{(1)}\boldsymbol{P}$
for $j = 1$ to $\ell_1 - 1$ do
{ $\boldsymbol{X} \leftarrow \tau^{-1}\boldsymbol{X}$, and $\boldsymbol{X} \leftarrow \boldsymbol{X} + s_j^{(1)}\boldsymbol{P}$ }
(Now $s = \tau^{-\ell_1+1}(\mathcal{S}^{(1)}\boldsymbol{P} + \mathcal{S}^{(2)}\tau(\frac{1}{2}\boldsymbol{P})) = \tau^{-\ell_1+1}s\boldsymbol{P}$)
$\boldsymbol{X} \leftarrow \tau^{\ell_1-1-n}\boldsymbol{X}$
Output $(\boldsymbol{X})$.

Algorithm 2. New scalar multiplication algorithm, right–to–left

of Algorithm 2. For simplicity we do not write down the resulting algorithm. The result is: *The scalar multiplication algorithm based on the new scalar decomposition can be done without any precomputations, and without requiring storage for the recoding.*

3.2 Field Represented Using a Polynomial Basis

In this case, squarings have a small, yet non-negligible cost: According to the experiments in [4, Section 3.5] we can assume $\frac{S}{M} \approx \frac{1}{8}$ for $n = 163$ and $\frac{S}{M} \approx \frac{1}{10}$ for $n = 233$. Knudsen [5] expects *"the time to compute a square root in a polynomial basis to be equivalent to half the time to compute a field multiplication plus a very small overhead"*. This is in the general case confirmed in [3]. So, τ and τ^{-1} have in general different costs. In [3] a special square root extraction algorithm is given if the field is represented via a trinomial: in the case of $\mathbb{F}_{2^{233}}$, a good trinomial is $f(x) = x^{233} + x^{74} + 1$ and a square root costs about $\frac{1}{8}M$.

If we use Algorithm 2 to perform a scalar multiplication, we pay a penalty due to the increased number of Frobenius (τ^{-1}) operations. One way to overcome this problem is to compute $\mathcal{S}^{(1)}\boldsymbol{P} + \mathcal{S}^{(2)}\boldsymbol{Q}$ using the *joint* representation obtained from $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$, i.e. the sequence of pairs $\left(s_i^{(1)}, s_i^{(2)}\right)_{i \geq 0}$ and Shamir's trick (actually due to Straus [16] and in a more general form). By Remark 2, at most one element in each pair $\left(s_i^{(1)}, s_i^{(2)}\right)$ is non-zero: Hence, we can use the

Input: A Koblitz curve E_a with corresponding parameter $\mu = (-1)^{1-a}$, a point $\boldsymbol{P}$ of odd order on E_a and a scalar s with associated (partially) reduced τ-NAF $\mathcal{S}$
Output: $s\boldsymbol{P}$

Compute the two τ-adic expansions
$\mathcal{S}^{(j)} = \sum_{i=0}^{\ell_{j-1}} s_i^{(j)} \tau^i$ for $j = 1, 2$
from $\mathcal{S}$ using Algorithm 1
$\boldsymbol{X} \leftarrow s^{(2)}_{\ell_2-1}\boldsymbol{P}$
for $j = \ell_2 - 2$ to 0 do
  { $\boldsymbol{X} \leftarrow \tau\boldsymbol{X}$, and $\boldsymbol{X} \leftarrow \boldsymbol{X} + s_j^{(2)}\boldsymbol{P}$ }
    (Now $\boldsymbol{X} = \mathcal{S}^{(2)}\boldsymbol{P}$)
$\boldsymbol{X} \leftarrow \tau^{n+2-\ell_1}\boldsymbol{X}$, $\boldsymbol{X} \leftarrow \frac{1}{2}\boldsymbol{X}$
    (Here we simplified $\boldsymbol{X} \leftarrow \tau^{-\ell_1+1+n}\boldsymbol{X}$, $\boldsymbol{X} \leftarrow \tau(\frac{1}{2}\boldsymbol{X})$.
     Now $\boldsymbol{X} = \mathcal{S}^{(2)}\tau^{-\ell_1+2}(\frac{1}{2}\boldsymbol{P})$.)
$\boldsymbol{X} \leftarrow \boldsymbol{X} + s^{(1)}_{\ell_1-1}\boldsymbol{P}$
for $j = \ell_1 - 2$ to 0 do
  { $\boldsymbol{X} \leftarrow \tau\boldsymbol{X}$, and $\boldsymbol{X} \leftarrow \boldsymbol{X} + s_j^{(1)}\boldsymbol{P}$ }
    (Now $s = \tau^{\ell_1-1}\mathcal{S}^{(2)}\tau^{-\ell_1+2}(\frac{1}{2}\boldsymbol{P}) + \mathcal{S}^{(1)}\boldsymbol{P} = \mathcal{S}^{(1)}\boldsymbol{P} + \mathcal{S}^{(2)}\boldsymbol{Q}$.)
Output $(\boldsymbol{X})$.

Algorithm 3. New scalar multiplication algorithm, left–to–right

Straus-Shamir trick without the need to precompute $\boldsymbol{P} \pm \boldsymbol{Q}$, and we only need to store $\boldsymbol{Q}$.

A better solution when the extraction of square roots is (relatively) expensive is to use a variant of Algorithm 2 with τ instead of τ^{-1}. We write it down as Algorithm 3: In this case we must store the τ-adic expansion before the scalar multiplication, and we need to compute and store each of $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$, before the corresponding τ-and-add loop.

4 Analysis and Performance Aspects

In the next subsection we prove the reduction of 14.29% in group additions of our method with respect to the τ-and-add method based on the τ-NAF. In Subsection 4.2 we estimate the effective improvement brought by our techniques by considering all group operations.

4.1 Complexity Analysis

The following lemma can be proved analysing the τ-NAF recoding algorithm. Similar results hold for the usual NAF (see for example [2]).

Lemma 2. *In a τ-NAF the probability that the digit immediately to the left of a 0 is another 0 is $\frac{1}{2}$ and that it is 1 or -1 is $\frac{1}{4}$ in each case*[(i)].

[(i)] The given probabilities are actually correct up to an error term exponentially decreasing in the length of the τ-NAF, and that does not influence the following analysis significantly.

To prove that our method gives an expected 14.29% reduction in group additions over the classical τ-and-add method, we model the reading of $\mathcal{S}$ in Algorithm 1 – and the consequent construction of $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$ – in terms of Markov chains. To do this, we describe the algorithm as a sequence of states taken from a list $\{\Sigma_0, \ldots, \Sigma_r\}$. States $\Sigma_0, \ldots, \Sigma_r$ occur with respective *limiting probabilities* $\sigma_0, \ldots, \sigma_r$. The states must be subject to the condition that the probability π_{ij} that the state following Σ_i is Σ_j depends only on the States Σ_i and Σ_j and not on the way State Σ_i has been reached. If $\Pi = (\pi_{ij})_{i,j=0}^{r}$ then the probabilities $\sigma_0, \ldots, \sigma_r$ sum up to 1 and form a vector $\boldsymbol{\sigma} = (\sigma_0 \ \ldots \ \sigma_r)$ such that $\boldsymbol{\sigma}\Pi = \boldsymbol{\sigma}$.

While scanning $\mathcal{S}$ in Algorithm 1 we are either attempting to form a maximal good k-block, or skipping zeros between blocks. We define five different states.

Σ_0: The state in which zeros outside k-blocks are skipped. Only one zero is skipped. All other states describe operations done to build k-blocks.

Σ_1: Entered whenever the first non-zero bit in a k-block is found. This is the one and only state where the first non-zero bit of a new k-block is read. Of course a zero bit follows and is skipped (the same also holds for States Σ_2–Σ_4). The following three states describe the scanning of the next bits in the k-block begun by entering State Σ_1.

Σ_2: Entered every time we find a non-zero bit which is the negative of the previous non-zero bit read. It can only follow States Σ_1 or Σ_2 itself.

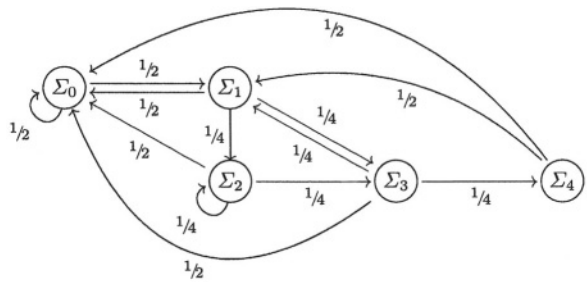
Σ_3: This state corresponds to the *first* non-zero bit having the same sign as the previous one. Either this bit is the last non-zero bit in a type **II** k-block or the second to last in a type **III** k-block.

Σ_4: Entered after Σ_3 if the third in a line of three non-zero bits having the same sign is found. This bit is the last bit in a type **III** k-block.

State Σ_0 is reached if and only if the bit to the left of the bit(s) of the previous state is 0. We recall that in all states except Σ_0 the algorithm actually processes *two* bits: a non-zero bit whose relation to the previous non-zero bits determines the actual state, and the following zero.

State Σ_1 may follow States Σ_3 and Σ_4 directly. This occurs when a k-block follows immediately a maximal good k-block of type **II** or **III.**

The following state diagram illustrates the flow of the algorithm. The nodes correspond to the states and the arrows are labelled with the transition probabilities, which follow immediately from Lemma 2.



Recall that π_{ij} denotes the transition probability from state Σ_i to state Σ_j. We have the following *probability transition matrix:*

$$\Pi = (\pi_{ij})_{i,j=0}^{4} = \begin{pmatrix} 1/2 & 1/2 & 0 & 0 & 0 \\ 1/2 & 0 & 1/4 & 1/4 & 0 \\ 1/2 & 0 & 1/4 & 1/4 & 0 \\ 1/2 & 1/4 & 0 & 0 & 1/4 \\ 1/2 & 1/2 & 0 & 0 & 0 \end{pmatrix}.$$

Now that Π is known, we can easily compute the limiting probabilities $\sigma_0, \ldots, \sigma_4$, which are uniquely determined, and are: $\boldsymbol{\sigma} = \frac{1}{42}(21\ 12\ 4\ 4\ 1)$.

Now suppose that, after λ state transitions, the algorithm has processed m bits of $\mathcal{S}$ and output a total of w non-zero bits in $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$. Since in state Σ_0 only one bit of $\mathcal{S}$ is scanned and in all other states two, after λ state transitions the expected number of processed bits is $m = \lambda(\sigma_0 + 2(1-\sigma_0)) = \lambda\left(\frac{1}{2} + 2\cdot\frac{1}{2}\right) = \frac{3}{2}\lambda$.

Now, good k-blocks of weight 1 and 2 are left in $\mathcal{S}^{(1)}$, whereas good k-blocks of weight at least 3 are cleared from $\mathcal{S}^{(1)}$ and appropriate sequences of weight 2 are inserted in $\mathcal{S}^{(2)}$ as described in Algorithm 1. Suppose the algorithm enters State Σ_1. If it immediately goes to State Σ_0, only one non-zero bit is output. In all other cases two non-zero bits are output. Then $w = \sigma_1\lambda\left(1\cdot\pi_{10} + 2\cdot(1-\pi_{10})\right) = \frac{12}{42}\lambda\left(\frac{1}{2} + 2\cdot\frac{1}{2}\right) = \frac{3}{7}\lambda$.

Last, suppose the length of the original τ-NAF is m. It has, as already recalled, about $m/3$ non-zero digits. However the number of the non-zero digits in $\mathcal{S}^{(1)} \cup \mathcal{S}^{(2)}$ is $2m/7$. Since the number of additions equals the number of non-zero digits, minus one, our method brings a reduction of $\left(\frac{1}{3} - \frac{2}{7}\right)/\frac{1}{3} \approx 14.29\%$ in additions with respect to the τ-and-add method.

4.2 Practical Estimates

We now estimate the actual speed-up for specific curves. As examples, we shall consider the Koblitz curves K-163 and K-233 over $\mathbb{F}_{2^{163}}$ and $\mathbb{F}_{2^{233}}$ from the FIPS standard issued by NIST [18].

Point halving (H), as described in Subsection 2.2, requires two field multiplications (M), the solution of an equation in λ of the type $\lambda^2 + \lambda = c$ (EQ) and the extraction of a square root ($\sqrt{}$). An elliptic curve addition (A) is done by one field inversion (I), two multiplications and one squaring (S). A point doubling (D) requires $I + 2M + 2S$. A Frobenius operation (τ) and its inverse (τ^{-1}) require $2S$ and $2\sqrt{}$ respectively.

With a polynomial basis, according to [4], $S \approx \frac{1}{7.5}M$ for $n = 163$ and $\frac{1}{9}M$ for $n = 233$. Following [3] we assume that, on average, $I \approx 8M$ when $n = 163$ and $I \approx 10M$ when $n = 233$. (For a comparison, [10] has $I \approx 9.3M$ for $n = 191$, for a software implementation on an embedded processor.) In $\mathbb{F}_{2^{233}}$, a field defined by a trinomial, a square root can be computed in $\approx \frac{1}{8}M$ [3, Example 3.12]. For $\mathbb{F}_{2^{163}}$ only a generic method is currently known, so $\sqrt{} \approx \frac{1}{2}M$. EQ takes, experimentally $\approx \frac{2}{3}M$.

If a normal basis is used, [5], S, $\sqrt{}$ and EQ have negligible costs. Because of the relatively high cost of a multiplication, we may assume $I \approx 3M$.

Since the length of a τ-expansion is $\approx n + a + 3$ (see Subsection 2.1), we see that the expected cost of the τ-and-add algorithm is $\frac{1}{3}(n+a+2)A + (n+a+2)\tau$. Algorithm 2 requires $\frac{2}{7}(n+a+2)A + 2(n+a+2)\tau^{-1}$ in the two loops; *Between* the two loops there are: H, 1 A, and on average $(n+a+3)-n = a+3$ Frobenius operations (τ). Algorithm 3 has similar costs in the main loops, with τ in place of τ^{-1}, but, on average, between the loops there is only a doubling and one addition. If the Straus-Shamir method is used (with a polynomial basis) right-to-left and with a single precomputation, the cost is $\frac{2}{7}(n+a+2)A + (n+a+3)\tau + H$.

In the following table we write down the costs of different scalar multiplication algorithms relative to that of one multiplication: the τ-and-add method based on the τ-NAF, our Algorithms 2 and 3 with the gain of the bast of the latter two over the τ-and-add. In the case of polynomial basis, we also show the costs of two methods requiring one precomputation: the one based on the Straus-Shamir trick from Subsection 3.2, and the usage of the width-2 τ-NAF (see [14,15]), which needs only $3\boldsymbol{P}$.

n	a	basis	τ-&-A	Algo. 2	Algo. 3	gain w.r.t. τ-&-A	*width-2* τ-&-A	*Straus-Shamir*
163	1	NB	276.7	244.1	–	11.8%	–	–
		poly	605	827	572.4	5.5%	*485.2*	*528.3*
233	0	NB	391.7	342.7	–	12.5%	–	–
		poly	1001	946.2	932.5	7%	*788.1*	*868.4*

The speed-ups are less than the theoretical estimate because of the additional overheads. The improvements will approach the theoretical maximum for large n. Our estimates are for software implementations. In hardware, where the ratio *I/M* is higher, the actual improvement will be much closer to the asymptotic maximum. But in that case one should also consider the use of projective coordinates. If one can store one precomputed point, the width-2 τ-NAF is faster than the Straus-Shamir trick.

5 Conclusions

In this paper we considered the problem of computing scalar multiplications on Koblitz curves. We combined for the first time the τ-adic expansion with point halving to give a new recoding of the scalar. By means of this we reduced the number of group operations required for a scalar multiplication by an asymptotic 14.29%.

For the curves K-163 and K-233 from NIST's FIPS standard we estimate an overall speedup of at least 12% if a normal basis is used.

The case where the field extension is represented using a normal basis is of particular relevance. It gives the highest speed-up, it allows to perform the scalar recoding online in the scalar multiplication, hence has no additional memory requirements (with respect to the classical τ-and-add method), apart from code size.

References

1. D.W. Ash, I. F. Blake and S. Vanstone. *Low complexity normal bases.* Discrete Applied Math. **25** (1989), pp. 191–210.
2. R. M. Avanzi. *On the complexity of certain multi-exponentiation techniques in cryptography.* To appear in Journal of Cryptology.
3. K. Fong, D. Hankerson, J. Lopez and A. Menezes. *Field inversion and point halving revisited.* Available from `http://www.cs.siu.edu/~kfong/research/ECCpaper.ps`, Unpublished Manuscript.
4. D. Hankerson, J. Lopez-Hernandez, and A. Menezes. *Software Implementatin of Elliptic Curve Cryprography over Binary Fields.* In: *Proceedings of CHES 2000.* LNCS 1965, pp. 1–24. Springer, 2001.
5. E. W. Knudsen. *Elliptic Scalar Multiplication Using Point Halving.* In: *Proocedings of ASIACRYPT 1999,* LNCS 1716, pp. 135–149. Springer, 1999.
6. D.E. Knuth. *The Art of Computer Programming.* Addison-Wesley, 1999. 3rd ed.
7. N. Koblitz. *Elliptic curve cryptosystems.* Mathematics of computation **48** (1987), pp. 203–209.
8. N. Koblitz. *CM-curves with good cryptographic properties.* In: *Proceedings of CRYPTO 1991,* LNCS 576, pp. 279–287. Springer, 1991.
9. V. S. Miller. *Use of elliptic curves in cryptography.* In: *Proceedings of CRYPTO '85.* LNCS 218, pp. 417–426. Springer, 1986.
10. J. Pelzl, T. Wollinger, J. Guajardo and C. Paar. *Hyperelliptic Curve Cryptosystems: Closing the Performance Gap to Elliptic Curves.* In: *Proceedings of CHES 2003.* LNCS 2779, pp. 351–365. Springer 2003.
11. G. W. Reitwiesner. *Binary arithmetic.* Advances in Computers, 1:231–308, 1960.
12. R. Schroeppel. *Point halving wins big.* Talks at: (i) Midwest Arithmetical Geometry in Cryptography Workshop, November 17–19, 2000, University of Illinois at Urbana-Champaign; and (ii) ECC 2001 Workshop, October 29–31, 2001, University of Waterloo, Ontario, Canada.
13. R. Schroeppel. *Elliptic curve point ambiguity resolution apparatus and method.* International Application Number PCT/US00/31014, filed 9 November 2000.
14. J. A. Solinas. *An improved algorithm for arithmetic on a family of elliptic curves.* In: *Proceedings of CRYPTO 1997,* LNCS 1294, pp. 357–371. Springer, 1997.
15. J. A. Solinas. *Efficient Arithmetic on Koblitz Curves.* Designs, Codes and Cryptography, Vol. 19 (2000), No. 2/3, pp. 125–179.
16. E. G. Straus, *Addition chains of vectors (problem 5125).* American Mathematical Monthly, vol. 71, 1964, pp. 806–808.
17. IEEE Std 1363-2000. *IEEE Standard Specifications for Public-Key Cryptography.* IEEE Computer Society, August 29, 2000.
18. National Institute of Standards and Technology. *Digital Signature Standard.* FIPS Publication 186-2, February 2000.

Application of Montgomery's Trick to Scalar Multiplication for Elliptic and Hyperelliptic Curves Using a Fixed Base Point

Pradeep Kumar Mishra and Palash Sarkar

Cryptology Research Group,
Applied Statistics Unit,
Indian Statistical Institute, 203 B T Road,
Kolkata-700108, INDIA

Abstract. We propose a scalar multiplication algorithm for elliptic and hyperelliptic curve cryptosystems, which uses affine arithmetic and is resistant against simple power attacks. Also, using a modification of known techniques the algorithm can be made immune against differential power attacks. The algorithm uses Montgomery's trick and a precomputed table consisting of multiples of the base point. Consequently, the algorithm is useful in a scenario where the base point is fixed, like Elgamal encryption or signature generation. Under such circumstances, for hyperelliptic curves, the algorithm compares favourably with other known algorithms over all fields. For elliptic curves, under similar circumstances, the algorithm performs better than other algorithms over prime fields. The increase in speed is due to a proper application of Montgomery's trick to efficiently perform the simultaneous inversion of several field elements.
Keywords : elliptic curves, hyperelliptic curves, scalar multipication, field inversion, explicit formulae, side-channel attacks.

1 Introduction

Elliptic curve cryptosystems (ECC) in recent years are gradually being inducted into many standards like ANSI, IEEE, NIST etc. The main advantage of these cryptosystems is that the key size is quite small in comparison to other cryptosystems like RSA, making these suitable for resource constrained devices, like smart card. Hyperelliptic curve cryptosystems (HECC) are also attractive, as the underlying field size is smaller and there are many more curves to choose from. ECC has already established itself as a popular public key cryptosystem. However, computational complexity of the HECC has till now come in the way of its commercial utilisation. Several research groups around the world have now diverted their attention to HECC to reduce its complexity and make it available for popular applications.

Both ECC and HECC are based on the discrete logarithm problem. The underlying group in ECC is provided by the set of points on the curve over a finite field on which an additive group operation is defined. On the other hand,

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 41–54, 2004.

cryptography using hyperelliptic curves is carried out in the Jacobian of such curves. The Jacobian is an additively written group and the elements of the Jacobian are called divisors. *In this paper, we will use the term point to mean both a point on an elliptic curve and a divisor in the Jacobian of a hyperelliptic curve.* The most important and computationally costly operation in (H)ECC is the scalar multiplication. Scalar multiplication is the operation of multiplying a point X with a scalar (an integer) m i.e. computing mX.

The efficiency of scalar multiplication depends to a large extent on the efficiency of addition and doubling operation of points. For elliptic curves point addition and doubling are relatively simple. Various co-ordinate systems have been proposed in the literature to reduce the complexity further. Divisors can be added in the Jacobian of hyperelliptic curves by Cantor's algorithm. However, this approach is not very efficient. One approach to improve the efficiency is to fix the genus of the curve and compute addition and doubling by explicit formulae. Such addition and doubling formulae were first described by Spallek [23] and have gone through many changes since then (see [6], [16], [24], [18], [19], [20]). For genus 2 curves an efficient set of formulae have been described by Lange in [12] and [13].

In the first paper [12], the author presents algorithms for addition and doubling, which involve the inversion of a field element along with some squarings and multiplications. In [13], algorithms are presented for addition and doubling which do not require inversion. Avoiding the inversion leads to some extra squarings and multiplications. This extra cost in terms of multiplications and squarings is not always desirable. Particularly in binary fields, where the the ratio of cost of inversion to cost of multiplication is not so high (between 3 and 8), inversion-free arithmetic is unnecessary. In prime fields, where this ratio is quite higher (≥ 30), inversion-free arithmetic seems to be more appropriate.

Side-channel attacks (SCA) were first proposed by Paul Kocher in 1996. The aim of SCA is to attack a specific implementation by measuring side-channel data like timing, power consumption traces, electro-magnetic radiation etc. One important class of these attacks, called power analysis attacks, uses the power consumption traces of the execution(s) of the implementation. Several measures have been proposed to defeat SCA in ECC.

In the current work, we present a new algorithm for computing the scalar multiplication for both elliptic and hyperelliptic curve cryptosystems. Our approach uses arithmetic with inversion and performs better than algorithms using inversion-free arithmetic in prime fields, where the cost of inversion is much higher than in binary fields. Moreover, the proposed algorithm is SCA resistant.

The efficiency of the algorithm is derived from the fact that, while computing the scalar multiplication, instead of scanning one bit at a time, several bits can be scanned from different locations in the binary representation of the multiplier and the point additions and doublings can be done simultaneously. As each of the additions and doublings involve one inversion, all these inversions can be computed simultaneously by Montgomery's trick (see Section 2.2) with only one inversion and some extra multiplications. The partial results so obtained are

added by another point addition algorithm, TreeADD, which computes addition of several points in a tree structure. The partial sums at the nodes of a particular level of the tree are computed together and the involved inversions are computed simultaneously by Montgomery's trick. This yields a very efficient scalar multiplication algorithm.

Our algorithm uses a precomputed table. So, it is useful in applications where the base point is fixed, like Elgamal encryption and signature generation etc. Also, the use of Montgomery's trick requires storage of several points which increases the memory requirement. In Section 5, it can be seen that for HECC over prime fields, when the base point is fixed, the algorithm is 77% faster than DPA resistant version of Coron's dummy addition method. Over binary fields the speed enhancement is about 28.1%. The performance is lower due to the fact that inversion is cheaper over such fields. For ECC over fields of characteristic > 3, under similar assumptions, the performance of our algorithm compares favourably against all SCA-resistant algorithms. The speed up is around 10% in the best scenario (window-size = 5).

2 Preliminaries

We first present a brief overview of hyperelliptic curves. For details, readers can refer to [10], [15], [11] or [3]. Let K be a field and let $\overline{K}$ be the algebraic closure of K. A *hyperelliptic curve* C of genus g (≥ 1) over K is an equation of the form $C : v^2 + h(u)v = f(u)$ where $h(u)$ in $K[u]$ is a polynomial of degree at most $g, f(u)$ in $K[u]$ is a monic polynomial of degree $2g + 1$, and there are no "singular points". *Elliptic curves are hyperelliptic curves of genus 1.*

A *divisor* D is an element of the free abelian group generated by all the points of the curve C over K. Let $\mathcal{D}$ stand for the set of all divisors. The *degree* of a divisor is defined to be the sum of all integer coefficients of the points occuring in the divisor. The set $\mathcal{D}^0$ of all divisors of degree 0 forms a subgroup of $\mathcal{D}$.

The set $\mathcal{D}^0$ can be partitioned into equivalence classes of divisors, each of which contains and hence is represented by an unique special type of divisor, called *reduced* divisors. Reduced divisors have a beautiful cannonical representation by means of two polynomials $[a(u), b(u)]$ of small degree over K. This is called Mumford's representation. The reduced divisors can be effectively added using Cantor's algorithm [3]. This group of reduced divisors is called the *Jacobian* of the curve C. It is generally denoted by $J_C(K)$. The discrete logarithm problem on the Jacobian of hyperelliptic curves of lower genus ($g \leq 4$) over suitable finite fileds K, is believed to be hard. This opens the possibility of realising different cryptographic primitives over it.

2.1 Point Arithmetic in (H)ECC

Let $[i], [m]$ and $[s]$ denote the amount of time required to compute an inversion, a multiplication and a squaring respectively in the underying field. We will use the notation $\mathbf{i}$ to denote the ratio $[i]/[m]$, which represents the relative cost of an

inversion compared to a multiplication. The value of **i** depends on the choice of the underlying field. For binary fields this value has been reported to be between 3 and 10 and for prime fields it is somewhere between 30 and 40 (see [5]). In prime fields the cost of a squaring is known to be somewhat less than the cost of a multiplication. *For simplicity, in the current work we assume [m] = [s].*

Elliptic Curve Arithmetic We only consider elliptic curves over prime fields. The equation of an elliptic curve over such a field is $y^2 = x^3 + ax + b$ where $a, b \in K$ and $4a^3 + 27b^2 \neq 0$. The cost of addition (ECADD) and doubling (ECDBL) algorithms for ECC in affine co-ordinates are $1[i] + 1[m] + 1[s]$ and $1[i] + 2[m] + 1[s]$ respectively.

Hyperelliptic Curve Arithmetic For addition of divisors in the Jacobian of hyperlliptic curves, use of explicit formulae has been proved to be the most efficient method. Many such formulae have been proposed in the literature by various authors. In this work, we will mostly concentrate on hyperlliptic curves of genus 2. For these curves Lange has provided a set of efficient formulae for addition and doubling in [12], [13]. The formulae proposed in [12] involve inversions in the underlying field. We will refer to these formulae as affine arithmetic. The formulae proposed in [13] do not involve inversions. We will refer to these as inversion-free arithmetic. For genus 3 and genus 4 curves affine formulae are proposed by Pelzl et al [19], [20]. Our proposed algorithm can also be used for curves of genera 2 and 3, which require 1 inversion each for divisor addition and doubling. In the current paper we will concentrate on curves of genus 2. Table 1 describes the complexity of addition and doubling formulae, proposed in [12], [13].

Table 1. Complexity of Explicit Formulae described in [12,13]

Name/Proposed in	Cost(Add)	Cost(Double)
Lange [12]	$1[i] + 22[m] + 3[s]$	$1[i] + 22[m] + 5[s]$
Lange [13]	$40[m] + 6[s]$	$47[m] + 4[s]$

2.2 Computing Inverses Simultaneously

Our scalar multiplication algorithm derives its efficiency from the fact that inversions of several field elements can be computed simultaneously by one inversion and some extra multiplications. One well known technique for doing this is Montgomery's trick [22], [17] which works as follows. Let $x_1, \cdots, x_n$ be the elements to be inverted. The algorithm first computes $a_1 = x_1, a_2 = x_1x_2, \cdots, a_n = x_1 \cdots x_n$, by $(n-1)$ multiplications. Then it inverts a_n. Now, x_n^{-1} is computed by multiplying a_n^{-1} by a_{n-1}. Also, $a_{n-1}^{-1} = a_n^{-1}x_n$ and x_{n-1}^{-1} is $a_{n-1}^{-1}a_{n-2}$. Similarly, it

computes inverse of other elements. It is not difficult to see that the algorithm uses only $3(n-1)$ multiplications and one inversion. We will denote the cost of computing the inverses of n field elements by $\mathcal{I}(n)$. Montgomery's trick shows that we can take $\mathcal{I}(n) = 1[i] + 3(n-1)[m]$.

2.3 Side Channel Attacks

In this subsection we discuss variuos countermeasures proposed in literature to resist side-channel attacks on (H)ECC.

Countermeasures Against SPA The usual binary algorithm for scalar multiplication is not secure against side channel attacks. To resist SPA, two approaches are generally resorted to in ECC. The first one is to make the computation independent of the bit pattern representing the scalar multiplier. Several countermeasure against SPA fall in this category. The simplest one is Coron's dummy addition method, i.e. to carry out one dummy addition if the corresponding bit is 0. Other approaches are based on various addition chains and window based methods. Particularly, for ECC, the Montgomery's ladder along with x-coordinate only encapsulated add-and-double algorithm proposed by Izu and Takagi and window-based methods proposed by Moller are very efficient and secure against SPA. The second approach uses indistinguishable algorithms for point addition and doubling. Certain elliptic curves like Hess and Jacobi form elliptic curves admit such algorithms. For a detailed treatment of these methods reader can refer to [8].

There is no result specific to HECC proposed in the literature to immunize the scalar multiplication algorithm against SPA. However, Coron's dummy addition method can easily be carried over to HECC.

Countermeasures Against DPA Several remedies have been proposed for immunising ECC from DPA. We briefly mention one such method – the Joye-Tymen countermeasure [9]. Let z be a random nonzero field element. The steps are as follows.

1. Compute z^2, z^3, z^4, z^6.
2. Transform the base point $P(x, y)$ to (z^2x, z^3y).
3. Transform the curve coefficients (a, b) to $a' = z^4a, b' = z^6b$.
4. Compute scalar multiplication with the new point on the new curve.
5. Transform the result (x, y) back to the original curve using $(x, y) \rightarrow (x/z^2, y/z^3)$.

The additional cost of obtaining DPA resistance is $4[m]$ for Step 1; $2[m]$ for Step 2; $2[m]$ for Step 3 and finally $1[i] + 2[m]$ for Step 5.

Recently, Avanzi [1] has generalised these techniques for HECC. Briefly, we describe the curve randomisation countermeasure which we will use in our algorithm. Let the underlying curve C of the cryptosystem be $y^2 + h(x)y = f(x)$ where $h(x) = h_2x^2 + h_1x + h_0$ and $f(x) = x^5 + f_4x^4 + f_3x^3 + f_2x^2 + f_1x + f_0$. Let $D = [u(x), v(x)]$ be the base divisor in C and let z be a random field element.

1. Compute $z^{-1}, z^{-2}, z^{-3}, z^{-4}, z^{-5}, z^{-6}, z^{-8}$ and z^{-10}.
2. Transform $h(x)$ and $f(x)$ into $\widetilde{h}(x)$ and $\widetilde{f}(x)$ as follows:
   $\widetilde{h}(x) = z^{-1}h_2x^2 + z^{-3}h_1x + z^{-5}h_0$ and
   $\widetilde{f}(x) = x^5 + z^{-2}f_4x^4 + z^{-4}f_3x^3 + z^{-6}f_2x^2 + z^{-8}f_1x + z^{-10}f_0$.
3. Transform D to $\widetilde{D} = [\widetilde{u}(x), \widetilde{v}(x)]$, where $\widetilde{D}$ is defined as follows :
   If $\deg(u) = 2$ and $u(x) = x^2 + u_1x + u_0$ and $v(x) = v_1x + v_0$ then
   $\widetilde{u}(x) = x^2 + z^{-2}u_1x + z^{-4}u_0$ and $\widetilde{v}(x) = z^{-3}v_1x + z^{-5}v_0$.
   If $\deg(u) = 1$ and $u(x) = x + u_0$ and $v(x) = v_0$ then $\widetilde{u}(x) = x + z^{-2}u_0$ and $\widetilde{v}(x) = z^{-5}v_0$.
4. Compute scalar multiplication using the new curve and the new divisor.
5. The result is transformed back to the original curve using the inverse of Step 3 and the relevant powers of z (rather than z^{-1}).

The additional cost of attaining DPA resistance is as follows: $1[i] + 7[m]$ for Step 1; $8[m]$ for Step 2; maximum $4[m]$ for Step 3 and $8[m]$ for Step 5. If the characteristic of the field is odd, then the polynomial $h(x)$ can be taken to be zero and hence the cost of Step 2 would come down to $5[m]$.

2.4 Scalar Multiplication Methods for HECC

Many scalar multiplication algorithms immunized against SCA have been proposed for ECC. We discuss some specific methods for genus 2 HECC using Lange's formula (see Table 1) along with their costs below.

(a) The usual add and double algorithm: For an n-bit multiplier such an algorithm requires n doublings and $n/2$ additions on the average (though this computation is not SCA resistant). So cost of computing the scalar multiplication using [13] is $n \times 46[m] + (n/2) \times 51[m] = 71.5n[m]$. Using affine arithmetic [12] the cost of n doublings and $n/2$ additions is $n \times (1[i] + 22[m] + 5[s]) + (n/2) \times (1[i] + 22[m] + 3[s]) \approx ((3/2)\mathbf{i} + 39.5)n[m]$. However, this computation is not SCA resistant.

(b) To make the computation SPA resistant we can resort to Coron's countermeasure for ECC. That is we can make some dummy additions if the corresponding bit in the binary representation is 0. In this case, we have to compute n additions and n doublings. Cost of the computation in inversion-free arithmetic will be $51n[m] + 46n[m] = 97n[m]$. This computation is costlier than that of (a), but is SPA resistant. But, again in binary fields the affine arithmetic will be preferable. In binary fields, the cost will be $n(1[i] + 22[m] + 5[s]) + n(1[i] + 22[m] + 3[s]) \approx (2\mathbf{i} + 52)n[m]$. It can be made resistant against DPA using the methods described above.

(c) We can encapsulate the addition and doubling formula of affine co-ordinates to obtain a more efficient formula. Suppose, we wish to compute an addition an doubling simultaneously. In affine co-ordinates, both will involve one inversion. Instead of computing two inversions, we can compute them by Montgomery's trick with 1 inversion and 3 multiplications. So cost of one addition and doubling is $1[i] + 3[m] + 52[m] \approx (\mathbf{i} + 55)[m]$. We can now use this algorithm in Montgomery's ladder type scalar multiplication algorithm to com-

pute the scalar multiplication. The method involves one doubling at the outset. Amount of computations involved in computing the scalar multiplication would be $((\mathbf{i}+55)n+\mathbf{i}+27)[m]$. Again the computation will be SPA resistant. It can also be made resistant against DPA.

We produce the summary of this discussion in Table 2. In the table we have considered two specific values of $\mathbf{i}$, 8 and 30. It is clear from the Table that, (c) is better than (b). Although, the average case complexity of (a) is better than (c), the former is not SCA resistant. Note that both (b) and (c) can be made DPA resistant (at an additional cost) by using Avanzi's countermeasure as discussed in Section 2.3.

Table 2. Complexity of different algorithms for HECC.

ALGORITHM	$\mathbf{i}$	COMPLEXITY
(a)	30	$71.5n[m]$ (avg case)
	8	$51.5n[m]$
(b)	30	$97n[m]$
	8	$68n[m]$
(c)	30	$(84n+57)[m]$
	8	$(63n+35)[m]$

3 New Algorithm for Scalar Multiplication

Before describing the proposed algorithm, let us have a close look at the addition and doubling algorithms in affine co-ordinates.

3.1 Addition and Doubling in Affine Co-ordinates

Let us consider the addition (HCADD) and doubling (HCDBL) algorithms for HECC in [12] in the most general and frequent case. These can be divided into three parts. In part one, some multiplications and squarings of the underlying field elements are carried out. In part two, a field element, generated in part one is inverted. The inverse so obtained in part two is used in part three along with some more multiplications and squarings of field elements. The output of part three provides the required divisor. See Figure 1.

Let us name the modules of these algorithms as $\mathbf{A_1}, \mathbf{A_2}, \mathbf{A_3}$ (parts of addition algorithms) and $\mathbf{D_1}, \mathbf{D_2}, \mathbf{D_3}$ (parts of doubling algorithms). In each of $\mathbf{A_2}$ and $\mathbf{D_2}$, we compute only one inverse. Let the number of multiplications and squarings required in $\mathbf{A_1}$, $\mathbf{A_3}$, $\mathbf{D_1}$ and $\mathbf{D_3}$ be $\mathbf{a_1}$, $\mathbf{a_3}$, $\mathbf{d_1}$ and $\mathbf{d_3}$ respectively. We will use the notation $\mathbf{a}$ for $\mathbf{a_1}+\mathbf{a_3}$ and $\mathbf{d}$ for $\mathbf{d_1}+\mathbf{d_3}$. By $\mathbf{A_1}(D_1, D_2)$, we will mean the field element α created in module $\mathbf{A_1}$ of the addition algorithm and which is inverted in $\mathbf{A_2}$. Similarly, by $\mathbf{D_1}(D_1)$, we will mean the field element β

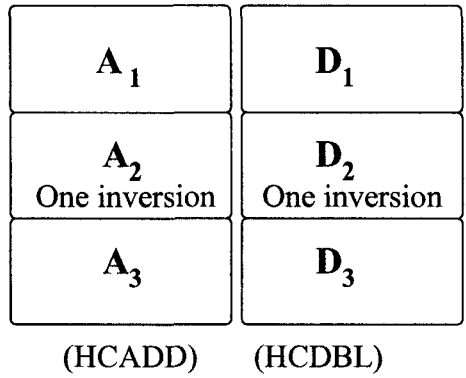


Fig. 1. HCADD and HCDBL algorithms proposed in [12]

created in module $\mathbf{D_1}$ of the doubling algorithm and which is inverted in module $\mathbf{D_2}$. By $\mathbf{A_3}(D_1, D_2, \alpha^{-1})$ (resp. $\mathbf{D_3}(D_1, \beta^{-1})$) we mean the divisor produced by the module $\mathbf{A_3}$ (resp. $\mathbf{D_3}$) as sum of D_1 and D_2 (resp. D_1).

The same can be said about addition and doubling algorithms for ECC in affine co-ordinates. In ECC, the values of $\mathbf{a_1}, \mathbf{a_3}, \mathbf{a}$ etc will be much smaller.

3.2 The Proposed Algorithm

Let $w \geq 2$ be a positive integer. We express m in the base 2^w. Let $m = c_0 + c_1 2^w + \cdots + c_{t-1} 2^{w(t-1)}$, where each $c_j \in \{0, \ldots, 2^w - 1\}$. Then $mD = c_0 D + c_1 2^w D + \cdots + c_{t-1} 2^{w(t-1)} D$. For all $j, 0 \leq j \leq t-1$ we precompute $2^{jw} D$ and store it in a table $T[]$. Thus $T[j] = 2^{jw} D$ for $0 \leq j \leq t-1$. This table is used to simultaneously compute $c_0 D, c_1 2^w D, \cdots, c_{t-1} 2^{w(t-1)} D$ using the right-to-left binary method. (A similar algorithm can also be developed using the left-to-right binary method.) Finally we add them to obtain mD.

Let the n-bit representation of m be $m_{n-1} \ldots m_0$. Note that $t = \lceil (n/w) \rceil$ We express c_j in binary, i.e., we write $c_j = c_j^0 + c_j^1 2 + \cdots + c_j^{w-1} 2^{w-1}$, where $c_j^i = m_{wj+i}$. We require $2t+1$ point type variables $R_0, \ldots, R_{t-1}$ and $Q_0, Q_1, \ldots, Q_t$. The variable R_j is initialized to $2^{jw} D$. Starting from the least significant bit of c_i, we scan a bit in each iteration. If the scanned bit is 1, then we add R_j to Q_{j+1}; if the scanned bit is 0, then we add R_j to Q_0; in either case we double R_j. After w iterations, Q_{j+1} is $c_j 2^{wj} D$. For $0 \leq j \leq t-1$, we compute all the expressions $c_j 2^{wj} D$ simultaneously. Each of the additions and doublings in each iteration will involve one inversion. While doing these additions and doublings, we carry out all the inversions simultaneously by Montgomery's trick. This yields an efficient algorithm for scalar multiplication. Our algorithm calls a routine INVERT, which simultaneously inverts a number of field elements using Montgomery's trick. Recall that by $\mathcal{I}(n)$ we denote the cost of inversion of n elements using INVERT and $\mathcal{I}(n) = 1[i] + 3(n-1)[m]$.

Algorithm EFF-SCLR-MULT
Input: $m, t, c_0, c_1, \cdots, c_{t-1}, D$.
Output: mD.

1. For $j = 0$ to $t-1$ $\{R_j = T[j]$;
2. If $c_j^0 = 0$ then $b = 0, t_b = j+1$
else $b = j+1, t_b = 0$, $Q_b = R_j$; $\}$
2. For $j = 0$ to $t-1$ let $\beta_j = \mathbf{D_1}(R_j)$;
3. Let $(\beta_0^{-1}, \cdots, \beta_{t-1}^{-1}) = \text{INVERT}(\beta_0, \cdots, \beta_{t-1})$
4. For $j = 0$ to $t-1$ let $R_j = \mathbf{D_3}(R_j, \beta_j^{-1})$;
5. For $i = 1$ to $w-2$
6. For $j = 0$ to $t-1$ $\{$ $\alpha_j = \mathbf{A_1}(R_j, Q_{j+1})$; $\beta_j = \mathbf{D_1}(R_j)$;$\}$
9. Let $(\alpha_0^{-1}, \cdots, \alpha_{t-1}^{-1}, \beta_0^{-1}, \cdots, \beta_{t-1}^{-1},) = \text{INVERT}(\alpha_0 \cdots \alpha_{t-1}, \beta_0, \cdots, \beta_{t-1})$
10. For $j = 0$ to $t-1$ $\{Q_{c_j^i(j+1)} = \mathbf{A_3}(R_j, Q_{j+1}, \alpha_j^{-1})$; $R_j = \mathbf{D_3}(R_j, \beta_j^{-1})$;$\}$
13. end do.
14. For $j = 0$ to $t-1$ let $\alpha_j = \mathbf{A_1}(R_j, Q_{j+1})$;
15. Let $(\alpha_0^{-1}, \cdots, \alpha_{t-1}^{-1}) = \text{INVERT}(\alpha_0, \cdots, \alpha_{t-1})$
16. For $j = 0$ to $t-1$ let $Q_{c_j^{w-1}(j+1)} = \mathbf{A_3}(R_j, Q_{j+1}, \alpha_j^{-1})$;
17. Let RES = TreeADD$(Q_1, \cdots, Q_t)$
18. Return (RES)

Proposition 1. *The cost of the above algorithm is* $[t(w-1)(\mathbf{a}+\mathbf{d}) + t\mathbf{a}][m] + (w-1)\mathcal{I}(2t) + \mathcal{I}(t)+$ *cost(TreeADD), where cost(TreeADD) is the cost of the TreeADD algorithm invoked by the algorithm.*

The algorithm TreeADD adds a number of points efficiently. It uses a tree structure for computation. Suppose $D_0, D_1, \cdots, D_{k-1}$ are the input points. For simplicity, assume $k = 2^r$. Imagine a tree of depth r with the input points at the leaf nodes. We pairwise add the points at the nodes with a common parent and put the sum at the parent node of each pair. There are 2^{r-1} nodes at level $r-1$ and to get the points at these nodes we have to perform 2^{r-1} additions. Note that, each of these additions needs one inversion. Instead of computing 2^{r-1} inversions separately, we can compute them with 1 inversion and $(3 \times 2^{r-1} - 1)$ multiplications using Montgomery's algorithm. This process is carried out at each level to the root. The root then contains the sum of all the points.

Algorithm TreeADD
Input: $D_0, \cdots, D_{2^k-1}$
Output: $D_0 + D_1 + \cdots + D_{2^k-1}$
1. For $i = 0$ to $2^k - 1$ let $D_i^{(0)} = D_i$.
2. For $j = 1$ to k
3. let $(D_0^{(j)}, D_1^{(j)}, \cdots, D_{2^{k-j}-1}^{(j)}) = \text{ADD}(D_0^{(j-1)}, D_1^{(j-1)}, \cdots, D_{2^{k-j+1}-1}^{(j-1)})$
4. Return $(D_0^{(k)})$

TreeADD invokes the algorithm ADD, which takes as input $2k$ points, $D_0, D_1, \cdots, D_{2k-1}$ and returns $D_0 + D_1, D_2 + D_3, \cdots, D_{2k-2} + D_{2k-1}$. ADD

computes k additions at one invocation. Hence, the inversions at $\mathbf{A_2}$ step of all these additions can be done simultaneously using the Montgomery's algorithm.

Algorithm ADD
Input: $D_0, \cdots, D_{2k-1}$.
Output: $D_0 + D_1, \cdots, D_{2k-2} + D_{2k-1}$.
1. For $i = 0$ to $k-1$, let $\alpha_i = \mathbf{A_1}(D_{2i}, D_{2i+1})$.
2. Let $(\alpha_0^{-1}, \alpha_1^{-1}, \cdots, \alpha_{k-1}^{-1}) = \text{INVERT}(\alpha_0, \alpha_1, \cdots, \alpha_{k-1})$.
3. For $i = 0$ to $k-1$ let $E_i = \mathbf{A_3}(D_{2i}, D_{2i+1}, \alpha_i^{-1})$.
4. Return $(E_0, E_1, \cdots, E_{k-1})$.

Proposition 2. *ADD takes* $2^r\mathbf{a}[m] + \mathcal{I}(2^r)$ *computations to compute the* $k = 2^r$ *sums of* $2k = 2^{r+1}$ *input points.*

With $\mathcal{I}(n) = 1[i] + 3(n-1)[m]$ (see Subsection 2.2), the cost of ADD becomes, $((2^r\mathbf{a} + 3(2^r - 1)[m] + 1[i]$.

Now we can compute the complexity of the algorithm TreeADD. TreeADD repeatedly calls the algorithm ADD, first with 2^r points, then with 2^{r-1} points and so on. Let the cost of ADD with k input points be $[kA]$. Then, cost of TreeADD with 2^r points is $[2^rA] + [2^{r-1}A] + \cdots + [1A]$. By Proposition 2, $[2^iA] = 2^{i-1}\mathbf{a}[m] + \mathcal{I}(2^{i-1})$. Hence computational cost of TreeADD is $\sum_{i=1}^{r-1}[2^iA] = \sum_{i=1}^{r-1} 2^{i-1}\mathbf{a}[m] + \mathcal{I}(2^{i-1}) = ((2^r - 1)\mathbf{a}[m] + \sum_{i=1}^{r-1}\mathcal{I}(2^{i-1})$. With $\mathcal{I}(n) = 1[i] + 3(n-1)[m]$, we have:

Proposition 3. *TreeADD takes* $(2^r - 1)\mathbf{a}[m] + \sum_{i=1}^{r-1}\mathcal{I}(2^i) = [(2^r - 1)\mathbf{a} + 3 \times 2^r - 3r - 3][m] + r[i]$ *computations to compute the sum of* 2^r *input points.*

We now compute the complexity of the algorithm EFF-SCLR-MULT. In the Steps 2–4 we double t points, inverting t elements by INVERT. In each iteration of the loop in Steps 5–13, we add t points and double t points. So in each iteration we invoke INVERT with $2t$ field elements. In the Steps 14–16 we add t points, inverting t elements by INVERT. Finally in Step 17 the TreeADD algorithm is invoked.

Proposition 4. *EFF-SCLR-MULT takes* $(w+r)[i] + [2^r(w-1)(\mathbf{a}+\mathbf{d}+6) - 3w + (2^r - 1)\mathbf{a} + 3 \times 2^r - 3r - 3][m]$ *computations to compute the scalar multiplication* mD *where,* m *is an* n*-bit integer,* w *is the window size and* $t = \lceil n/w \rceil = 2^r$.

The algorithm uses a table which must be precomputed. Online computation will be very costly. The table will store t points. An elliptic curve point is an ordered pair of field elements. Each field element is of 160 bits. So a point occupies 320 bits of memory. Similarly, a divisor of a hyperelliptic curve of genus 2 is a 4-tuple of field elements, where each field element is of around 80 bits. So, a divisor also occupies almost the same amount of memory. The algorithm needs to store t points means, it requires about $320t$ bits of storage.

4 Resistance Against SCA

In this section we discuss the resistance of our algorithm to side-channel attacks and the cost involved in achieving such resistance.

4.1 Resistance Against SPA

Algorithm EFF-SCLR-MULT is resistant against simple power attacks, the reason being the following. At each iteration in steps 2 to 10, we are scanning t bits from the binary representation of m and computing some additions and doublings. The numbers of additions and doublings are fixed and independent of the actual bit pattern scanned. Similarly in steps 12 to 17, the same number of additions are being computed irrespective of the actual bits scanned. Hence we conclude that the computations are resistant against SPA.

4.2 Resistance Against DPA

We discuss a method for making the algorithm resistant against DPA. Recall that we use a look-up table $T[\,]$ of t points. The steps for the counter-measure for both ECC and HECC are as follows.

1. Choose a random nonzero field element z.
2. Compute the relevant powers of z. (see Subsection 2.3
3. Transform the curve parameters.
4. Transform each of the t points of $T[\,]$.
5. Perform scalar multiplication using Algorithm EFF-SCLR-MULT.
6. Transform the result back to the original curve.

The specific transformations are different for ECC and HECC. For ECC we use Joye-Tymen transformation while for HECC we use Avanzi's transformation. Accordingly the costs are also different. From the discussion in Section 2.3 we get the following costs.

ECC : $1[i] + 4[s] + (4 + 2t)[m] \approx 1[i] + (8 + 2t)[m]$ assuming $[m] = [s]$.
HECC : $1[i] + (23 + 4t)[m]$ $(1[i] + (20 + 4t)[m]$ for odd characteristic).

5 Results and Comparison

In this section, we present some results of our algorithm and compare it with other algorithms. We do it separately for HECC of genus 2 and ECC.

5.1 HECC

We compare the performance of Algorithm EFF-SCLR-MULT to the algorithms (a), (b) and (c) described in Section 2.4. Table 3 displays these calculations. In Table 3, columns (a)-(c) refer to the algorithms listed in rows (a)-(c) of Table 2. Cost of DPA resistance has been added to the cost of (b) and (c). Column (d) stands for our algorithm. The column n stands for the bit size of the multiplier m. The parameter w stands for the window size. The complexity of the algorithms (a)-(c) do not vary with w as they are not window-based. The parameter t stands for the size of the look up table and is equal to the number of points required

Table 3. HECC: Comparison of the number of multiplications for different values of the number of bits n required to represent the scalar multiplier m.

Parameters			$\mathbf{i} = 8$				$\mathbf{i} = 30$			
n	w	t	(a)	(b)	(c)	(d)	(a)	(b)	(c)	(d)
160	5	32	8240	10896	10129	8501	11440	15539	13665	8743
160	10	16	8240	10896	10129	8937	11440	15539	13665	9267
160	20	8	8240	10896	10129	9190	11440	15539	13665	9718
160	40	4	8240	10896	10129	9389	11440	15539	13665	10335
160	80	2	8240	10896	10129	9690	11440	15539	13665	11440

to be stored. One can easily observe that, for certain window sizes over prime fields, the proposed algorithm (d) is better than even average case complexity of double-and-add (column (a)), which is not SPA resistant. In the best scenario, the new algorithm achieves a speed-up of around 77 percent if $\mathbf{i} = 30$ over usual SPA resistant double and always add approach (b). Over binary fields, the performance enhancement is lower, only 28.1%, due to the fact that $\mathbf{i}$ is lower. For other values of $\mathbf{i}$ similar comparisons can be made. We have observed that as $\mathbf{i}$ increases from 30 to 40, the cost of (b) goes up by around 10%, whereas the cost of our algorithm goes up by about 1% only.

5.2 Efficiency for Elliptic Curves

The algorithm EFF-SCLR-MULT can also be used for ECC over prime fields. This is because ECADD and ECDBL algorithms in affine co-ordinates have a structure similar to that of Figure 1. For 160 bits scalar multiplier, the amount of computation required by the algorithm to compute the scalar multiplication is shown in Table 4. To compare the performance of the algorithm with other

Table 4. ECC: Number of multiplications required by EFF-SCLR-MULT assuming $\mathbf{i} = 30$.

n	w	t	Complexity
160	5	32	2222[m]
160	10	16	2410[m]
160	20	8	2693[m]
160	40	4	3226[m]

algorithms proposed in the literature we show Table 5 which is taken from [8]. It shows efficiency of some other SCA resistant methods. Note that the table does not exactly matches with the table presented in [8], as we have not taken additions into account and have taken $[s] = [m]$. Table 5 shows that for efficient and secure computation of scalar multiplication, Improved Moller's method with

Table 5. ECC: Number of multiplications required by previous algorithms under the assumptions $[\mathbf{i}] = 30$ and $[m] = [s]$.

Method	(160-bit ECC)
Coron's dummy addition	3375[m]
Coron's dummy addition with $a = -3$	3057[m]
Improved Moller with $w = 2$	3220[m]
Improved Moller with $w = 2$ and $a = -3$	3064[m]
Improved Moller with $w = 3$	2543[m]
Improved Moller with $w = 3$ and $a = -3$	2429[m]
Improved Izu-Takagi	2758[m]
Improved Izu-Takagi with $a = -3$	2439[m]

window size 3 and $a = -3$ is the best. It takes $2429[m]$ computations. Our algorithm takes $2222[m]$ in the best situation, when the window size is 5, nearly 10% performance enhancement.

5.3 Memory Requirement

The parameter t in Table 3 determines the size of the look-up table. It is equal to the number of points to be stored in the look-up table. It is natural that the efficiency of the algorithm goes up as we invert more and more elements together. If a window size of 5 is chosen then the table size will be 32. In hyperelliptic curve cryptosystem with reasonable security, a point size is around 320 bits. So, the table will occupy around 1.2 kilobyte of memory.

Additionally Algorithm EFF-SCLR-MULT requires some more intermediate points and field elements. The calculation for these is as follows.

- $2t+1$ intermediate points including one dummy point (Q_0) for Coron's trick.
- $2t$ field elements $(\alpha_0, \ldots, \alpha_{t-1})$ and $(\beta_0, \ldots, \beta_{t-1})$ for applying Montgomery's trick.

This memory requirement might be costly for memory constrained applications (as in smart card applications). In such situations our algorithm cannot be used. However, we note that in situations where the amount of memory is not a constraint (as in desktops), our algorithm provides a speed-up over the known algorithms (for fixed base point).

References

1. Roberto M. Avanzi. Countermeasures Against Differential Power Analysis for Hyperelliptic Curve Cryptosystems. In *CHES 2003,* to appear.
2. E. Brior and M. Joye. Weierstrass Elliptic Curves and Side-Channel Attacks. In *PKC 2002,* LNCS 2274, pages 335-345, Springer-Verlag,2002.
3. D. G. Cantor. Computing in the Jacobian of a Hyperelliptic curve. In *Mathematics of Computation,* volume 48, pages 95-101, 1987.

4. J.-S. Coron. Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In *CHES 1999,* pages 292-302.
5. K. Fong and D. Hankerson and J. López and A. Menezes. Field inversion and point halving revisited. Technical Report, CORR 2003-18, Department of Combinatorics and Optimization, University of Waterloo, Canada, 2003.
6. R. Harley Fast Arithmetic on Genus 2 Curves. *Avaiable at http://cristal.inria.fr/ harley/hyper,2000.*
7. T. Izu and T. Takagi. A Fast Parallel Elliptic Curve Multiplication Resistant against Side-Channel Attacks Technical Report CORR 2002-03, University of Waterloo,2002. Available at http://www.cacr.math.uwaterloo.ca
8. T. Izu, B. Moller and T. Takagi. Improved Elliptic Curve Multiplication Methods Resistant Against Side Channel Attacks. Proceedings of Indocrypt 2002, LNCS 2551, pp 296-313, Springer-Verlag.
9. M. Joye and C. Tymen Protection against differential attacks for elliptic curve cryptography. CHES 2001, LNCS 2162, pp 402-410, Springer-Verlag.
10. N. Koblitz. Hyperelliptic Cryptosystems. In *Journal of Cryptology,* 1: pages 139–150, 1989.
11. N. Koblitz. *Algebraic Aspects of Cryptology,* Algorithms and Computation in Mathematics. Springer Verlag, 1998.
12. T. Lange. Efficient Arithmetic on Genus 2 Curves over Finite Fields via Explicit Formulae. Cryptology ePrint Archive, Report 2002/121, 2002. *http://eprint.iacr.org/* .
13. T. Lange. Inversion-free Arithmetic on Genus 2 Hyperelliptic Curves. Cryptology ePrint Archive, Report 2002/147, 2002. *http://eprint.iacr.org/* .
14. A. J. Menezes, P. C. van Oorschot and S. A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1997.
15. A. Menezes, Y. Wu, R. Zuccherato. An Elementary Introduction to Hyperelliptic Curve. Technical Report CORR 96-19, University of Waterloo(1996),Canada. Available at http://www.cacr.math.uwaterloo.ca
16. Y. Miyamoto, H. Doi, K. Matsuo, J. Chao and S. Tsujii. A fast addition algorithm for genus 2 hyperelliptic curves. In *Proc of SCIS2002, IEICE, Japan,*pp 497-502,2002, in Japanese.
17. P. Montgomery. Speeding the Pollard and Elliptic Curve Methods for Factorisation. In *Math. Comp.,* vol 48, pp 243-264, 1987.
18. K. Nagao. Improving Group Law Algorithms for Jacobians of Hyperelliptic Curves. In W. Bosma, editor, *ANT IV,* LNCS 1838, Berlin 2000, Springer-Verlag.
19. J. Pelzl, T. Wollinger, J. Guajardo and C. Paar. Hyperelliptic Curve Cryptosystems: Closing the Performance Gap to Elliptic Curves. Cryptology ePrint Archive, Report 2003/26, 2003. *http://eprint.iacr.org/* .
20. J. Pelzl, T. Wollinger, J. Guajardo and C. Paar. Low Cost Security: Explicit Formulae for Genus 4 Hyperelliptic Curves . Cryptology ePrint Archive, Report 2003/97, 2003. *http://eprint.iacr.org/* .
21. *K.* Okeya and K. Sakurai. Efficient Elliptic Curve Cryptosystems from a Scalar Multiplication Algorithm with Recovery of the y-coordinate on a Montgomery form Elliptic Curve. In *CHES 2001,* LNCS 2162, pp 126-141, Springer-Verlag 2001.
22. H. Shacham, D. Boneh. Improving SSL Handshake Performance via Batching. In *CT-RSA,* LNCS 2020, pp 28-43 Springer-Varlag, 2001.
23. A. M. Spallek. Kurven vom Geschletch 2 und irhe Anwendung in Public-Key-Kryptosystemen. Ph D Thesis, Universitat Gesamthochschule, Essen, 1994.
24. M. Takahashi. Improving Harley Algorithms for Jacobians of Genus 2 Hyperelliptic Curves. In *Proc of SCIS 2002,* ICICE, Japan, 2002, in Japanese.

Fast Arithmetic on Jacobians of Picard Curves

Stéphane Flon[1] and Roger Oyono[2*]

[1] Mathematisches Institut, Universität Bonn, Beringstr. 4, 53115 Bonn, Germany
flon@math.uni-bonn.de
[2] IEM, Universität Essen, Ellernstr. 29, 45326 Essen, Germany
oyono@exp-math.uni-essen.de

Abstract. In this paper we present a fast addition algorithm in the Jacobian of a Picard curve over a finite field $\mathbb{F}_q$ of characteristic different from 3. This algorithm has a nice geometric interpretation, comparable to the classic "chord and tangent" law for the elliptic curves. Computational cost for addition is $144M+12SQ+2I$ and $158M+16SQ+2I$ for doubling.

Introduction

The discrete logarithm problem (DLP) is one of the two main problems on which public key cryptography is based (the other one being integer factorisation, in RSA cryptosystem): for example, Diffie-Hellman key exchange protocol [3] and ElGamal cryptosystem [4] are based on this problem.

In 1987, Miller [16] and Koblitz [11] suggested (independently) the use of the group of points of an elliptic curve over a finite field for DLP. It is now a well treated subject, and is even used in some industrial applications. Most of today's research is focused on the natural generalization of this example: DLP in the Jacobian of higher genus curves. One advantage is that, given an abstract finite group, one can use smaller fields (as Hasse-Weil formula shows).

In order to produce cryptosystems based on these Jacobian varieties, the first thing to worry about is to have secure cryptosystems (see [12] to find secure Picard curves). Still, it is very important to compute efficiently in the group, and an important part of today's reseach is devoted to allow fast arithmetic in Jacobians of curves. For instance, many papers study the case of hyperelliptic curves of genus 2 and 3 ([14, 15, 13, 19]).

In this article, we find explicit formulae for computing in the Jacobian of a Picard curve, basing us on some geometric aspects of these curves. Volcheck [23], Huang and Ierardi [10] already proposed general methods for computing in the Jacobians of arbitrary algebraic curves. These algorithms are not practical from a computational point of view though, and in addition they need to extend the base field. Hess' paper [9] is closer to our geometrical point of view, in such as it provides an explicit version of Riemann-Roch theorem (see also [8]).

* This work was supported by the European Community's Human Potential Programme (G.T.E.M. network) for the first author and by DFG for the second one.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 55–68, 2004.

1 Preliminaries and Notations

1.1 Jacobian Varieties of Algebraic Curves

In this section, we briefly recall fundamental facts on Picard groups and Jacobians. The letter k stands for an arbitrary perfect field, and $\overline{k}$ denotes a given algebraic closure of k.

Let C be a complete non-singular curve over k. The *divisor group of* C is the free abelian group $\mathrm{Div}(C)$ consisting of formal sums $\sum_{P\in C(\overline{k})} m_P \cdot P$, in which the m_P's are integers, finitely many of them being non-zero. Each divisor consists in an obvious way of a *positive part* and a *negative part.* It is called *effective* if there is no negative part.

A divisor is *defined over* k if it is fixed by the natural Galois action of $\mathrm{Gal}\,(\overline{k}|k)$. The *divisor group of* C *over* k, denoted $\mathrm{Div}_k(C)$, is the group of elements of $\mathrm{Div}(C)$ defined over k.

Given any $D = \sum_{P\in C(\overline{k})} m_P \cdot P \in \mathrm{Div}(C)$, one can define the *degree of D,* denoted *deg*(*D*), as $\sum_P m_P$.

Let f be a non-zero element of the function field of C. Then, the *divisor* of f is

$$(f) := \sum_{P\in C(\overline{k})} v_P(f) \cdot P$$

where $v_P(f)$ denotes the valuation of f in the discrete valuation ring $\overline{k}[C]_P$.

Any such divisor is called a *principal divisor,* and two divisors are said to be *equivalent* if they differ from a principal divisor. One can check that any principal divisor is indeed a degree zero divisor. Moreover, if f is defined over k, then $(f) \in \mathrm{Div}_k(C)$.

The *divisor class group (or the Picard group),* denoted $\mathrm{Pic}(C)$, is then the quotient of the group $\mathrm{Div}(C)$ by the subgroup of principal divisors. We let $\mathrm{Pic}_k(C)$ be the subgroup of $\mathrm{Pic}(C)$ fixed by the natural Galois action of $\mathrm{Gal}\,(\overline{k}|k)$. If we substitute $\mathrm{Div}(C)$ by $\mathrm{Div}^0(C)$, we respectively obtain the *degree* 0 *part of the divisor class group of C,* denoted $\mathrm{Pic}^0(C)$, and its subgroup $\mathrm{Pic}^0_k(C)$.

The most important and striking fact about $\mathrm{Pic}^0_k(C)$ is that it admits a kind of a "reification" (as D. Mumford suggestively presents them), the *Jacobian variety* J_C of C. More precisely, J_C represents a functor attached to the Picard group of C (see [17] for a very dense introduction to Jacobian varieties). It is automatically an abelian variety, whose dimension is the genus of C. Moreover, for each field L such that C has a L-rational point, the group $J_C(L)$ is canonically isomorphic to $\mathrm{Pic}^0_L(C)$.

Suppose the curve C has an affine model over k, with only one point at infinity (this is the case for Picard curves). Then, one can see the Jacobian in a third way, namely as the *ideal class group* of the integral closure of $k[x]$ in $k(C)$ (which is a Dedekind ring) associated to this model ([5, p. 6] or [7]). The sum of two divisors corresponds to the product of the associated ideals.

Of course, it may appear obvious to compute in the Jacobian (or, equivalently, in the degree zero Picard group): the sum of two divisors is just the resulting formal sum. But it is of considerable importance for cryptographic ends to have a unique and concise way to express divisors. This leads to the notion of a *reduced divisor.* Indeed, a consequence of Riemann-Roch theorem is the following representation theorem of divisors:

Theorem 1 (Representation by reduced divisors). *Let C be a non-singular curve over k of genus g, with a given k-point P_∞. Let D be an element of $Div_k^0(C)$. Then, there exists an effective divisor E over k of degree $m \leq g$, whose support does not contain P_∞, and such that $E - m \cdot P_\infty$ is equivalent to D (we refer to such a divisor as an* almost reduced divisor).

It is unique if we demand m to be minimal, and it is then called the reduced representation *of (the divisor class of) D.*

1.2 Picard Curves and Their Jacobians

In the following k is any field of characteristic different from 3.

A *Picard curve* is a genus 3 cyclic trigonal curve. Any Picard curve C admits a projective model of the following form

$$z \cdot y^3 = z^4 \cdot f_4(x/z)$$

where f_4 is a monic degree 4 separable polynomial of one variable over k. It has a unique point at infinity, P_∞, namely $(0:1:0)$.

Any Picard curve C appears as a cyclic Galois cover of degree 3 of the projective line, with 5 (totally) ramified points (including P_∞). The automorphism group of this cover is generated by

$$\sigma : (x : y : z) \mapsto (x : \zeta y : z)$$

where ζ is a non-trivial cubic root of unity. Two points are *conjugate* if they lie on the same geometric fibre of the cover. Each non-ramification point P of C has thus two *conjugate points,* namely P^σ and P^{σ^2}.

Note that $v_{P_\infty}(x) = -3$ and $v_{P_\infty}(y) = -4$. Let f be a polynomial in $k[x, y]$, of degree m, not lying in the ideal of C. According to Bézout theorem (as C is irreducible), the intersection multiplicity of f with C at P_∞, denoted by $\mathrm{ord}_\infty(f)$, is equal to $4m + v_{P_\infty}(f)$.

In the following, we will use the so-called "Mumford representation" of divisors. This represention arises from the one proposed in [18], page 3.17, for reduced divisors of hyperelliptic curves. One may see it as an interpolation theorem for the points in the support of the divisor. This is harmless for hyperelliptic curves, as there can not be any pair of conjugate points in the support of a reduced divisor of a hyperelliptic curve. Unfortunately, this is not true anymore for Picard curves, and in fact Mumford representation is only suitable for a peculiar (but very likely) class of reduced divisors, namely the ones that do not have any two conjugate points in their support (they are called *typical* in [2], a terminology that we will keep in this paper).

Theorem 2 (Reduced divisors and Mumford representation). *An almost reduced divisor is not reduced if and only if its positive part D_0 is of degree 3, and such that there exists a line l with $(l)_0 \geq D_0$.*

Let D be a typical reduced divisor over k. It can then be uniquely represented as the intersection divisor of u and $y - v$, with:

- $u, v \in k[x]$,
- u *monic,*
- $\deg(v) < \deg(u) \leq 3$, *and*
- $u | v^3 - f_4$.

Note 1. For any typical reduced divisor D, we will note its Mumford representation polynomials by u_D and $y - v_D$. In the ideal class group, D corresponds to $< u_D, y - v_D >$.

Proof. The presented proof differs from the one of [2].

First of all, let us treat the case where $D_0 = P + Q$ is of degree 2. Suppose we have $P + Q - 2 \cdot P_\infty = R - P_\infty + (f)$ for a $f \in k(C)$. Then,

$$P + Q + R^\sigma + R^{\sigma^2} - 4 \cdot P_\infty = (f_1)$$

for a $f_1 \in k(C)$. As $v_{P_\infty}(f_1) = -4$, f_1 must be a line not passing through P_∞. This contradicts the fact that it goes through R^σ and R^{σ^2}.

Suppose now that $D = P_1 + P_2 + P_3 - 3 \cdot P_\infty$. The divisor D can not be equivalent to some $R - P_\infty$, because this would prove the existence of a polynomial f such that $v_{P_\infty}(f) = -5$.

If D is equivalent to some $Q_1 + Q_2 - 2 \cdot P_\infty$, we have to distinguish two cases, namely whether Q_1 and Q_2 are conjugate or not.

If they are not conjugate, then

$$P_1 + P_2 + P_3 + Q_1^\sigma + Q_1^{\sigma^2} + Q_2^\sigma + Q_2^{\sigma^2} - 7 \cdot P_\infty = (f)$$

with f a conic crossing C once through P_∞. It crosses the line $(Q_1 P_\infty)$ (resp. $(Q_2 P_\infty)$) in three points, thus it should contain these two lines. This contradicts the previous statement.

In the remaining case (D equivalent to $Q_1 + Q_1^\sigma - 2 \cdot P_\infty$), one has

$$P_1 + P_2 + P_3 + Q_1^{\sigma^2} - 4 \cdot P_\infty = (f)$$

This means that there exists a line f such that $(f)_0 \geq P_1 + P_2 + P_3$.

The second part of the theorem is straightforward. □

Remark 1. In the case of a non-typical divisor $D = P_1 + P_1^\sigma + P_2$, then one can write D as the intersection divisor of $u \in k[x]$ (corresponding to the two lines $(P_1 P_\infty)$ and $(P_2 P_\infty)$), $\deg(u) \leq 2$, with an element of the k-vector space spanned by $1, x, y, x^2, y^2, xy$ (corresponding to the two lines $(P_1 P_2)$ and $(P_1^\sigma P_2)$).

The presented algorithm in the next section only works for typical divisors, and the result is an almost reduced divisor, which is with very high probability a typical one.

2 Fast Addition Algorithm for Jacobian of Picard Curves

2.1 Main Algorithm

As said in the introduction, the following algorithm is inspired by the "chord and tangent" law on the group of points of an elliptic curve. In our case, we will have to replace the chord or the tangent by a cubic, and we will introduce a conic in order to get the opposite of a divisor. Note that for an elliptic curve, or even a hyperelliptic curve, the latter operation requires no computation.

In [20], the authors make use of similar geometric constructions to propose a reduction algorithm. Instead of using a cubic, they work recursively, reducing a degree 4 effective divisor into a degree ≤ 3 effective divisor, with the help of two conics. Their algorithm requires to work with rational points (or to perform some field extensions). It also requires to make a final factorisation of a polynomial in $k[x]$ of degree at most 3. As our algorithm is completely explicit (i.e. we only perform some elementary operations in the base field k), we will not need any of these requirements.

Geometric Description of the Jacobian Group Addition. In the most common case, we have two typical reduced divisors $D_1 := P_1 + P_2 + P_3 - 3 \cdot P_\infty$ and $D_2 := Q_1 + Q_2 + Q_3 - 3 \cdot P_\infty$, and we want to find the reduced divisor equivalent to $P_1 + P_2 + P_3 + Q_1 + Q_2 + Q_3 - 6 \cdot P_\infty$. Let us consider the divisor

$$D := -(P_1 + P_2 + P_3 + Q_1 + Q_2 + Q_3 - 9 \cdot P_\infty)$$

This is a degree 3 divisor defined over k. Riemann-Roch theorem asserts that

$$l(D) - l(K - D) = \deg(D) + 1 - g = 1$$

(where K stands for the canonical divisor), so that in any case $l(D) \geq 1$.

In particular, there exists a w in $k(C)$ such that $(w) \geq -D$. As the only pole of w is P_∞, it is a polynomial in $k[x, y]$. Moreover, as $v_{P_\infty}(w) \geq -9$, one knows that w is an element of the k-vector space spanned by $1, x, x^2, xy, y, y^2, x^3$. From now on, we take w to be the unique such element (up to a multiplicative factor) with maximal valuation at P_∞.

If w is a conic, a very unlikely situation, then geometric considerations on *J(C)* allow a very easy computation of the reduction of $D_1 + D_2$. Let us illustrate this in the case where the support of $D_1 + D_2$ consists of six points aside from P_∞ that lie on a (unique) conic, not going through P_∞. Then the conic crosses C in exactly two more points Q_1 and Q_2. Taking the line through those two points gives us two new points K_1 and K_2, such that $K_1 + K_2 - 2 \cdot P_\infty$ is the reduction of $D_1 + D_2$ (see Fig. 1).

If w is a cubic, Bézout theorem asserts that the corresponding variety crosses C in exactly three more points, say R_1, R_2 and R_3. One has the obvious relation

$$(P_1 + P_2 + P_3 - 3 \cdot P_\infty) + (Q_1 + Q_2 + Q_3 - 3 \cdot P_\infty) = -(R_1 + R_2 + R_3 - 3 \cdot P_\infty) + (w)$$

so that we have obtained an almost reduced form of the opposite of $D_1 + D_2$.

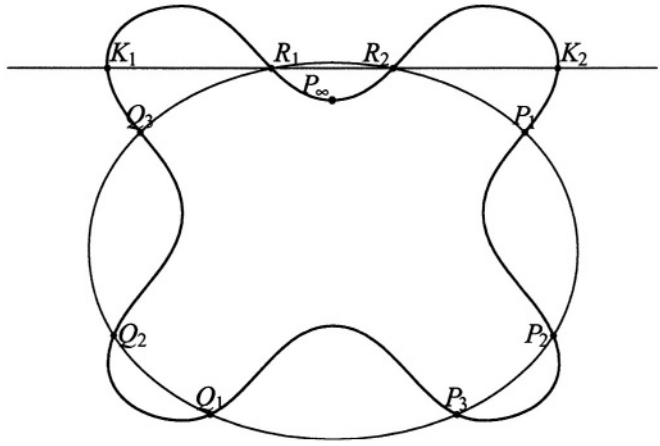


Fig. 1. Case where w is a conic

Using Riemann-Roch in the same way as we have just done, one can show that there exists a unique conic v going through R_1, R_2, R_3 and twice in P_∞. It crosses C in three further points K_1, K_2, K_3, and by construction, $K_1+K_2+K_3-3\cdot P_\infty$ is in the class of D_1+D_2.

One can roughly sum-up how the algorithm works by Fig. 2.

Algebraic Interpretation and Formulae. The presented algorithm can be naturally divided into three steps: finding w, reduce $-(D_1+D_2)$, and then taking the opposite (with the conic). Now we give an algebraic interpretation of these steps.

First step: computation of the cubic

This is the only step where one has to distinguish between addition and doubling.

Addition

First of all, let us treat the most common case, in which w can be expressed as

$$w = y^2 + s \cdot y + t$$

where s and t are polynomials in x, with $\deg(s) \leq 1$ and $\deg(t) \leq 3$. As the support of D_1 (resp. D_2) is contained in the support of (w), we are naturally led to find three polynomials s, δ_1 and δ_2 in x, of degree ≤ 1, such that

$$w = (y - v_1) \cdot (y + v_1 + s) + u_1 \cdot \delta_1 = (y - v_2) \cdot (y + v_2 + s) + u_2 \cdot \delta_2$$

It is easy to see that the leading coefficient of δ_1 (resp. δ_2) has to be the square of that of v_1 (resp. v_2).

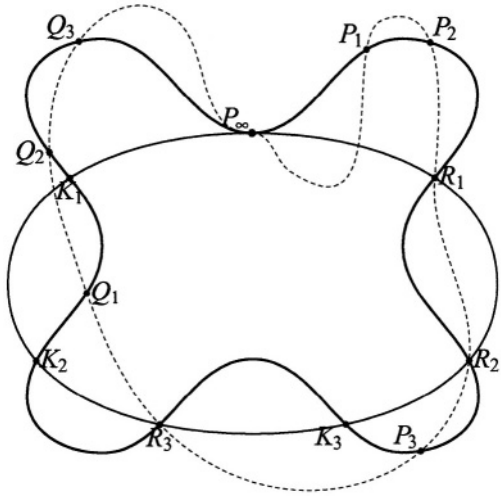


Fig. 2. Description of the algorithm

It then leads to the unique condition:

$$(v_1 + v_2 + s) \cdot (v_1 - v_2) + u_2 \cdot \delta_2 - u_1 \cdot \delta_1 = 0$$

In case w has no y^2 term, then the same strategy gives the condition

$$s \cdot (v_1 - v_2) + \delta_2 \cdot u_2 - \delta_1 \cdot u_1 = 0$$

where δ_1 and δ_2 are constant polynomials.

Note that these two equations are very similar. In fact, during the computation of s and δ_1, we consider in both subcases the remainder r of $t_1 \cdot u_1$ by u_2, where t_1 is the inverse of $v_1 - v_2$ modulo u_2. It turns out that if r is of degree 2, then we are in the first subcase, if not we are in the second one.

The only remaining case is a trivial one; namely when the points of the support of D_1 are conjugate of the points of the support of D_2.

Doubling

In that case, we are looking for a w in the ideal $I^2 =< u_1^2, u_1 \cdot (y - v_1), (y - v_1)^2 >$. Here we only treat the main subcase, where w has a y^2 part, and hence when w can be written in the following manner:

$$(y - v_1) \cdot (y + v_1 + s) + u_1 \cdot \delta_1$$

(the other subcases are either similar or trivial, and very unlikely anyway). The unique condition, obtained in the same way as above, is then

$$(y - v_1) \cdot (2v_1 + s) + u_1 \cdot \delta_1 \in I^2$$

In other respects, an easy computation shows that:

$$3v_1^2(y - v_1) - u_1 \cdot w_1 \in I^2$$

where w_1 is defined by $v_1^3 - f_4 = u_1 \cdot w_1$.

This implies that

$$3v_1^2 u_1 \cdot \delta_1 + (2v_1 + s) \cdot u_1 \cdot w_1 \in I^2$$

If v_1 is prime to u_1, that is if the support of D_1 does not contain any ramification point (different from P_∞), then we have

$$u_1 | \left(3v_1^2 \cdot \delta_1 + (2v_1 + s) \cdot w_1\right)$$

and the computation of the inverse of w_1 in $k[x]/(u_1)$ gives us δ_1, and then s.

Remark 2. If the support of $D_1 + 3 \cdot P_\infty$ does contain a ramification point, then the geometry of the curve allows us to compute the reduction of $2 \cdot D_1$ easily.

Second step: computation of $-(D_1 + D_2)$

Here, we only treat the most common case (which is also the most difficult one), namely when w has a y^2 term, and hence can be written

$$w = y^2 + s \cdot y + t^3$$

with $s, t \in k[x]$, $\deg(s) \leq 1$ and $\deg(t) \leq 3$.

We already know how to characterize the reduced divisor equivalent to $-(D_1 + D_2)$: it suffices to compute the intersection divisor of the (variety attached to the) cubic w with C.

A way to find $u_{-(D_1+D_2)}$ is thus to compute the resultant $\mathrm{Res}(w, C)$ of w with $y^3 - f_4$ (relative to y), to compute the quotient of $\mathrm{Res}(w, C)$ by $u_1 \cdot u_2$, and then to normalize.

To compute $v_{-(D_1+D_2)}$, one can exploit the relation

$$(t - s^2) \cdot v_{-(D_1+D_2)} \equiv (s \cdot t - f_4) \bmod (u_{-(D_1+D_2)})$$

so that $v_{-(D_1+D_2)}$ is the remainder of the quotient of $\alpha_1 \cdot (s \cdot t - f_4)$ by $u_{-(D_1+D_2)}$, where α_1 is the inverse of $t - s^2$ in $k[x, y]/(u_{-(D_1+D_2)})$.

Third step: computation of $D_1 + D_2$

Obviously, one has $v_{D_1+D_2} = v_{-(D_1+D_2)}$. Thus, we are reduced to compute $u_{D_1+D_2}$. It is easily obtained as the (normalized) euclidean quotient of $(v_{D_1+D_2})^3 - f_4$ by $u_{-(D_1+D_2)}$.

2.2 Explicit Formulae in the Most Common Case

The given algorithms correspond to the case when w has a y^2 term. Note that in order to speed up the algorithm, we have used Karatsuba tricks to multiply two polynomials. Similarly, we only compute the coefficients we need in the algorithm. For instance, as we only need to know the quotient of the resultant of w and C by $u_1 \cdot u_2$, the degree ≤ 5 part of this resultant is irrelevant. The reader can find the tables for addition and doubling at the end in the appendix of this article.

3 Remarks and Outlook

As far as we know, the presented algorithm for computing in the Jacobian of a Picard curve is quite efficient. In [2, p. 24], the authors present estimations for the cost of various algorithms computing the reduction of a typical divisor of degree 6 in the Jacobian of a Picard curve. The most efficient algorithm is supposed to need roughly $150M$ and $6I$. The composition in itself has a computational cost of about $50M$ and $1I$.

The cost for addition in the Jacobian of hyperelliptic curves of genus 3 is substantially lower than ours (it is about $I + 70M + 6SQ$, see [19]). On the other hand, for cryptographic purposes, scalar multiplication is the main topic. In that respect, our algorithm benefits from the two following remarks, which should approximately halve the complexity: one can speed up scalar multiplication using the fast automorphism σ defined p. 57, (see [6]), and rather use –2-adic expansions instead of 2-adic usual expansions (see [1]).

Our viewpoint was definitely geometric, and we did not separate composition from reduction. One may hope that this viewpoint can be generalised to a much broader class of curves. This statement is strenghtened by the fact that Cantor algorithm and its improvements [14] for computing in the Jacobian of a hyperelliptic curve of genus 2 can be interpreted in the very same way as our algorithm. Note though that this case is the only one where Cantor's algorithm and ours coincide.

We have presented formulae for Picard curves. We stress the fact that they are immediately adaptable to non-singular curves of genus 3 with a hyperflex. In that case, addition requires $160M + 17SQ + 2I$ and a doubling requires $177M + 21SQ + 2I$.

Acknowledgements

The authors would like to thank Gerhard Frey for the valuable ideas he gave, as well Tanja Lange for reading the manuscript in depth, and also for the support she provided. We also thank the referees for their comments, and Roberto Avanzi for pointing out to us the use of fast endomorphisms.

References

[1] R. M. Avanzi, G. Frey, T. Lange, and R. Oyono. On using expansions to the base of –2. submitted, see arXiv, math.NT/0312060,2003, 2003.

[2] A. Basiri, A. Enge, J-C. Faugère, and N. Gürel. The arithmetic of Jacobian groups of superelliptic cubics. Technical report, INRIA, 2002.

[3] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE transactions on information theory,* 22:644–654, 1976.

[4] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory,* 31-4:469–472, 1985.

[5] S. Galbraith, S.M. Paulus, and N. Smart. Arithmetic on superelliptic curves. *Math. Comp.,* 71(237):393–405, 2002.

[6] R. P. Gallant, J. L. Lambert, and S. A. Vanstone. Faster point multiplication on elliptic curves with efficient endomorphisms. In *Advances in Cryptology – Crypto'2001,* volume 2139 of *Lect. Notes Comput. Sci.,* pages 190–200. Springer, 2001.

[7] R. Hartshorne. *Algebraic geometry,* volume 52. Springer-Verlag, GTM, 1977.

[8] F. Hess. *Zur Divisorenklassengruppenberechnung in globalen Funktionenkörpern.* PhD thesis, TU Berlin, 1999.

[9] F. Hess. Computing Riemann-Roch spaces in algebraic function fields and related topics. *J, Symbolic Comp.,* 33(4):425–445, 2002.

[10] M-D. Huang and D. Ierardi. Efficient algorithms for the Riemann-Roch problem and for addition in the Jacobian of a curve. *J. of symb. comp.,* 18:519–539, 1994.

[11] N. Koblitz. Elliptic curve cryptosystems. *Math. Comp.,* 48(177):203–209, 1987.

[12] K. Koike and A. Weng. Construction of cryptographically secure cm-picard curves, 2003. preprint.

[13] J. Kuroki, M. Gonda, K. Matsuo, J. Chao, and S. Tsujii. Fast genus three hyper–elliptic curve cryptosystems. In *SCIS 2002,* 2002.

[14] T. Lange. Efficient arithmetic on genus 2 hyperelliptic curves over finite fields via explicit formulae. In *Cryptology ePrint archive,* Report 2002/121, 2002. http://eprint.iacr.org/.

[15] K. Matsuo, J. Chao, and S. Tsujii. Fast genus two hyperelliptic curve cryptosystems. Technical report, IEICE, 2001. ISEC2001-31.

[16] V.S. Miller. The use of elliptic curves in cryptography. In *Advances in cryptology-CRYPTO '85,* volume 218 of *LNCS,* pages 417–426, Santa Barbara, California, 1986. Springer-Verlag.

[17] J-S. Milne. Jacobian varieties. In Cornell G. and J.H. Silverman, editors, *Arithmetic Geometry,* pages 167–212. Springer, 1986.

[18] D. Mumford. *Tata lectures on theta II,* volume 43 of *Progress in mathematics.* Birkhaüser, 1984.

[19] J. Pelzl, T. Wollinger, J. Guajardo, and C. Paar. Hyperelliptic curves cryptosystems: closing the performance gap to elliptic curves. In *Cryptology ePrint archive,* 2003. http://eprint.iacr.org/.

[20] E. Reinaldo-Barreiro, J. Estrada-Sarlabous, and J-P. Cherdieu. Efficient reduction on the Jacobian variety of Picard curves. In *Coding theory, cryptography, and related areas,* volume 877, pages 13–28. Springer, 1998.

[21] J.H. Silverman. *The arithmetic of elliptic curves,* volume 106. Springer-Verlag, GTM, 1994.

[22] H. Stichtenoth. *Algebraic function fields and codes.* Springer-Verlag, 1993.

[23] E. Volcheck. Computing in the Jacobian of a plane algebraic curve. In Adleman, editor, *ANTS-I,* volume 877, pages 221–233. Springer-Verlag, 1994.

Table 1. Addition, $\deg u_1 = \deg u_2 = 3$

Input	$D_1 = [u_1, v_1]$ and $D_2 = [u_2, v_2]$ $u_i = x^3 + u_{i2}x^2 + u_{i1}x + u_{i0}, v_i = v_{i2}x^2 + v_{i1}x + v_{i0}$ $f = x^4 + f_3x^3 + f_2x^2 + f_1x + f_0$
Output	$D = [u_{D_1+D_2}, v_{D_1+D_2}] = D_1 + D_2$ with $u_{D_1+D_2} = x^3 + d_1x^2 + d_2x + d_3$ $v_{D_1+D_2} = v_2'x^2 + v_1'x + v_0'$

Step	Expression	Operations
1	compute resultant res_1 of $(v_1 - v_2)$ and u_2, and $z_1 := res_1/(v_1 - v_2) \bmod u_2$ $t_1 = u_{21}(v_{22} - v_{12})$, $t_2 = u_{22}(v_{22} - v_{12})$, $t_3 = u_{20}(v_{22} - v_{12})$; $t_4 = u_{22}(v_{20} - v_{10})$, $t_5 = u_{21}(v_{21} - v_{11})$, $t_6 = (v_{22} - v_{12})(t_1 + v_{10} - v_{20})$; $t_7 = (v_{21} - v_{11})(v_{21} - v_{11} - t_2)$, $t_8 = (t_4 - t_3 - t_5)(t_2 + v_{11} - v_{21})$; $t_9 = (v_{22} - v_{12})(t_4 - t_3 - t_5)$, $t_{10} = (v_{21} - v_{11})(v_{20} - v_{10} - t_1)$; $inv_0 = t_6 + t_7$, $t_{11} = inv_0 \cdot u_{22}$, $t_{12} = u_{20}(v_{21} - v_{11})$; $t_{13} = inv_0 \cdot t_{12}$, $t_{14} = t_3(t_9 - t_{10})$, $s_1 = (v_{20} - v_{10} - t_1)^2$; $inv_2 = t_8 + s_1$, $t_{15} = inv_2(v_{20} - v_{10})$; $inv_1 = t_{11} + t_9 - t_{10}$, $res_1 = t_{15} - t_{13} - t_{14}$; $z_1 = inv_0x^2 + inv_1x + inv_2$	15M+1SQ
2	compute the cubic $w = y^2 + sy + t$ $t_{16} = (u_{12} - u_{22})inv_0$, $t_{17} = (u_{11} - u_{21})inv_1$; $t_{18} = (u_{10} - u_{20})inv_2$, $t_{19} = (u_{12} + u_{11} - u_{22} - u_{21})(inv_0 + inv_1)$; $t_{20} = (u_{12} + u_{10} - u_{22} - u_{20})(inv_0 + inv_2)$; $t_{21} = (u_{11} + u_{10} - u_{21} - u_{20})(inv_1 + inv_2)$; $t_{22} = u_{22} \cdot t_{16}$, $t_{23} = u_{21} \cdot t_{16}$, $t_{24} = u_{22}(t_{22} + t_{16} + t_{17} - t_{19})$; $t_{25} = (u_{21} + u_{20})(t_{19} - t_{22} - t_{17})$, $t_{26} = u_{20}(t_{22} + t_{16} + t_{17} - t_{19})$; $r_0 = t_{24} + t_{20} + t_{17} - t_{23} - t_{16} - t_{18}$; $r_1 = t_{21} + t_{23} - t_{17} - t_{18} - t_{25} - t_{26}$, $r_2 = t_{18} + t_{26}$, $s_2 = v_{12}^2$; $t_{27} = r_0 \cdot res_1$, $t_{28} = r_0 \cdot s_2$, $t_{29} = r_0 \cdot t_{28}$, $t_{30} = t_{28} \cdot res_1$; $t_{31} = -res_1 \cdot (v_{12} + v_{22})$, $t_{32} = r_1 \cdot s_2$, $t_{33} = u_{22} \cdot t_{28}$; $\gamma_1 = t_{31} + t_{33} - t_{32}$, $t_{34} = res_1 \cdot \gamma_1$, $t_{35} = -t_{27}(v_{11} + v_{21})$; $t_{36} = -t_{27}(v_{10} + v_{20})$, $t_{37} = r_1\gamma_1$, $t_{38} = r_2 \cdot t_{28}$, $t_{39} = r_2 \cdot \gamma_1$; $t_{40} = u_{21} \cdot t_{29}$, $t_{41} = u_{20} \cdot t_{29}$; $\lambda_1 = t_{35} + t_{40} - t_{37} - t_{38}$, $\mu_1 = t_{36} + t_{41} - t_{39}$; $t_{42} = -t_{27} \cdot v_{12}$, $t_{43} = -t_{27} \cdot v_{11}$; $t_{44} = -t_{27} \cdot v_{10}$, $t_{45} = (v_{12} + v_{11})(t_{42} + t_{43} - \lambda_1)$; $t_{46} = v_{11}(t_{43} - \lambda_1)$, $t_{47} = (v_{12} + v_{10})(t_{42} + t_{44} - \mu_1)$; $t_{48} = v_{10}(t_{44} - \mu_1)$, $t_{49} = (v_{11} + v_{10})(t_{43} + t_{44} - \lambda_1 - \mu_1)$; $t_{50} = t_{30}(u_{12} + u_{11})$, $t_{51} = u_{11} \cdot t_{30}$, $t_{52} = t_{34}(u_{12} + u_{10})$, $t_{53} = u_{10} \cdot t_{34}$; $t_{54} = (u_{11} + u_{10})(t_{30} + t_{34})$, $B_0 = t_{34} + t_{50} + t_{45} + t_{30} - t_{51} - t_{46}$; $B_1 = t_{52} + t_{30} + t_{51} + t_{47} + t_{46} - t_{53} - t_{48}$; $B_2 = t_{54} + t_{49} - t_{51} - t_{53} - t_{46} - t_{48}$; $B_3 = t_{53} + t_{48}$; $t_{55} = B_0 \cdot t_{27}$, $i_1 = (t_{55})^{-1}$, $t_{56} = i_1 \cdot B_0$; $t_{57} = i_1 \cdot t_{27}$, $t_{58} = t_{57} \cdot t_{27}$, $t_{59} = t_{57} \cdot B_1$; $t_{60} = t_{57} \cdot B_2$, $t_{61} = t_{57} \cdot B_3$, $t_{62} = t_{56} \cdot \lambda_1$, $t_{63} = t_{56} \cdot \mu_1$; $t_{64} = t_{56} \cdot B_0$, $t_{65} = t_{56} \cdot B_1$, $t_{66} = t_{56} \cdot B_2$, $t_{67} = t_{56} \cdot B_3$; $w = y^2 + (t_{62}x + t_{63})y + t_{64}x^3 + t_{65}x^2 + t_{66}x + t_{67}$	52M+1SQ+1I
3	compute $res(w, C, y)$ $s_3 = t_{59}^2$, $t_{68} = t_{59}(6t_{60} + s_3)$, $s_4 = t_{62}^2$, $s_5 = (t_{62} + t_{63})^2$; $s_6 = t_{63}^2$, $t_{69} = t_{62}t_{64}$, $t_{70} = t_{62}(s_4 - 3t_{65})$; $t_{71} = t_{63}t_{64}$, $t_{72} = -3f_3t_{69}$, $t_{73} = t_{62}(s_5 - 3t_{66} - s_4 - s_6)$; $t_{74} = t_{63}(s_4 - 3t_{65})$, $t_{75} = f_3t_{70}$, $t_{76} = -3f_2t_{69}$, $t_{77} = -3f_3t_{71}$; $s_7 = t_{58}^2$, $t_{78} = t_{58}s_7$, $t_{79} = t_{78}(1 - 3t_{69})$; $t_{80} = t_{78}(t_{70} + t_{72} + 2f_3 - 3t_{71})$; $t_{81} = t_{78}(t_{73} + t_{74} + t_{75} + t_{76} + t_{77} + 2f_2 + f_3^2)$;	14M+5SQ
4	compute $u_{-(D_1+D_2)}$ $t_{82} = u_{12}u_{22}$, $t_{83} = u_{12}u_{21}$, $t_{84} = u_{11}u_{22}$; $t_{85} = (u_{11} + u_{21} + u_{10} + u_{20} + t_{82} + t_{83} + t_{84})(1 + t_{79} + 3t_{59} - u_{12} - u_{22})$; $t_{86} = (u_{10} + u_{20} + t_{83} + t_{84})(t_{79} + 3t_{59} - u_{12} - u_{22})$; $c_1 = t_{79} + 3t_{59} - u_{12} - u_{22}$, $t_{87} = c_1(u_{12} + u_{22})$; $c_2 = t_{80} + 3t_{60} + 3s_3 - u_{11} - u_{21} - t_{82} - t_{87}$, $t_{88} = c_2(u_{12} + u_{22})$; $c_3 = u_{11} + u_{21} + t_{68} + t_{81} + t_{82} + t_{86} + 3t_{61} - t_{88} - t_{85}$; $u_{-(D_1+D_2)} = x^3 + c_1x^2 + c_2x + c_3$	7M

5	compute $res(t-s^2, u_{-(D_1+D_2)}, x)$ and precomputations for $v_{D_1+D_2}$ $t_{89} = c_3 t_{64}$, $t_{90} = c_1 t_{64}$, $t_{91} = c_2 t_{64}$, $t_{92} = c_2(t_{65} - s_4)$; $t_{93} = c_1(t_{66} + s_4 + s_6 - s_5)$, $t_{94} = c_3(t_{66} + s_4 + s_6 - s_5)$; $t_{95} = c_2(t_{67} - s_6)$, $t_{96} = c_3(t_{65} - s_4)$, $t_{97} = c_1(t_{67} - s_6)$; $s_8 = (t_{89} + s_6 - t_{67})^2$, $s_9 = (t_{91} + s_5 - t_{66} - s_4 - s_6)^2$; $t_{98} = (t_{94} - t_{95})(t_{90} + s_4 - t_{65})$; $t_{99} = (s_8 - t_{98})(t_{89} + t_{92} + s_6 - t_{67} - t_{93})$; $t_{100} = (t_{96} - t_{97})(t_{90} - t_{65} + s_4)$; $t_{101} = (t_{91} + s_5 - t_{66} - s_4 - s_6)(t_{89} + s_6 - t_{67})$; $t_{102} = (t_{96} - t_{97})(t_{100} - 2t_{101})$; $t_{103} = s_9(t_{94} - t_{95})$, $res_2 = t_{99} + t_{102} + t_{103}$; $t_{104} = (t_{90} + s_4 - t_{65})(t_{92} + t_{89} + s_6 - t_{93} - t_{67})$; $j_0 = t_{104} - s_9$, $t_{105} = c_1 j_0$, $t_{106} = c_1(t_{100} - t_{101})$; $t_{107} = c_2 j_0$, $t_{108} = c_3(t_{66} + s_4 + s_6 - s_5)$; $t_{109} = (t_{108} - t_{95})(t_{90} + s_4 - t_{65})$, $j_1 = t_{105} + t_{101} - t_{100}$; $j_2 = t_{107} + t_{109} - t_{106} - s_8$, $t_{110} = t_{62}(t_{65} + t_{66})$; $t_{111} = t_{62} t_{66}$, $t_{112} = t_{63}(t_{65} + t_{67})$, $t_{113} = t_{63} t_{67}$; $t_{114} = (t_{62} + t_{63})(t_{66} + t_{67})$, $t_{115} = c_1(1 - t_{69})$; $t_{116} = c_1(t_{115} + t_{71} + t_{110} - f_3 - t_{111})$, $t_{117} = c_2(1 - t_{69})$; $t_{118} = (c_2 + c_3)(1 + f_3 + t_{111} - t_{69} - t_{115} - t_{71} - t_{110})$; $t_{119} = c_3(t_{115} + t_{71} + t_{110} - f_3 - t_{111})$; $t_{120} = j_0(t_{116} + f_2 + t_{113} - t_{117} - t_{112} - t_{111})$; $t_{121} = (j_0 + j_1)(t_{116} + f_2 + f_1 + 2t_{113} - t_{112} - t_{114} - t_{118} - t_{119})$; $t_{122} = j_1(f_1 + t_{111} + t_{113} + t_{117} - t_{114} - t_{118} - t_{119})$; $t_{123} = (j_0 + j_2)(t_{116} + f_2 + f_0 + t_{119} - t_{112} - t_{117} - t_{111})$; $t_{124} = j_2(f_0 + t_{119} - t_{113})$; $t_{125} = (j_1 + j_2)(f_1 + f_0 + t_{111} + t_{117} + t_{119} - t_{114} - t_{118} - t_{119})$; $t_{126} = c_1 t_{120}$, $t_{127} = c_2 t_{120}$; $t_{128} = c_1(t_{126} + t_{120} + t_{122} - t_{121})$, $t_{129} = (c_2 + c_3)(t_{121} - t_{126} - t_{122})$; $t_{130} = c_3(t_{126} + t_{120} + t_{122} - t_{121})$;	42M+2SQ
6	compute $v_{D_1+D_2}$ $t_{131} = res_2(t_{128} + t_{123} + t_{122} - t_{127} - t_{120} - t_{124})$, $i_2 = (t_{131})^{-1}$; $t_{132} = i_2(t_{128} + t_{123} + t_{122} - t_{127} - t_{120} - t_{124})$; $t_{133} = t_{132}(t_{128} + t_{123} + t_{122} - t_{127} - t_{120} - t_{124})$; $t_{134} = t_{132}(t_{125} + t_{127} - t_{122} - t_{124} - t_{129} - t_{130})$; $t_{135} = t_{132}(t_{124} + t_{130})$; $v_2' = -t_{133}$, $v_1' = -t_{134}$, $v_0' = -t_{135}$;	5M+1I
7	compute $u_{D_1+D_2}$ $s_{10} = res_2^2$, $t_{136} = i_2 s_{10}$, $s_{11} = t_{136}^2$, $t_{137} = t_{136} s_{11}$; $t_{138} = t_{136} t_{134}$, $s_{12} = t_{138}^2$, $t_{139} = t_{136} t_{135}$; $t_{140} = t_{138}(s_{12} + 6t_{139})$, $t_{141} = t_{137} f_3$; $t_{142} = c_1(3t_{138} - c_1)$, $d_1 = 3t_{138} - c_1$; $d_2 = 3t_{139} + 3s_{12} + t_{137} - c_2 - t_{142}$; $t_{143} = c_1 d_2$, $t_{144} = c_2(3t_{138} - c_1)$; $d_3 = t_{140} + t_{141} - c_3 - t_{143} - t_{144}$;	9M+3SQ
total		144M, 12S, 2I

Table 2. Doubling, $\deg u_1 = 3$

Input	$D_1 = [u_1, v_1]$ $u_1 = x^3 + u_{12}x^2 + u_{11}x + u_{10}, v_1 = v_{12}x^2 + v_{11}x + v_{10}$ $f = x^4 + f_3x^3 + f_2x^2 + f_1x + f_0$
Output	$D = [u_{2D_1}, v_{2D_1}] = 2D_1$ with $u_{2D_1} = x^3 + d_1x^2 + d_2x + d_3$ $v_{2D_1} = v_2'x^2 + v_1'x + v_0'$

Step	Expression	Operations
1	compute w_1 such that $u_1w_1 = v_1^2 - f$ $s_1 = v_{12}^2$, $s_2 = v_{11}^2$, $t_1 = -s_1v_{12}$, $t_2 = -3s_1v_{11}$; $t_3 = v_{12}v_{10}$, $t_4 = -3v_{12}(t_3 + s_2)$; $t_5 = -v_{11}(s_2 + 6t_3)$, $t_6 = t_1u_{12}$, $t_7 = t_1u_{11}$; $t_8 = u_{12}(t_2 - t_6)$, $t_9 = u_{12}(t_4 + 1 - t_7 - t_8)$; $t_{10} = (u_{11} + u_{10})(t_1 + t_2 - t_6)$, $t_{11} = u_{10}(t_2 - t_6)$;	11M+2SQ
2	compute resultant res_1 of w_1 and u_1, and $z_1 := res_1/w_1 \bmod u_1$ $t_{12} = -u_{10}t_1$, $t_{13} = u_{11}(t_6 - t_2)$; $t_{14} = u_{12}(t_7 + t_8 - t_4 - 1)$, $t_{15} = u_{10}(t_7 + t_8 - t_4 - 1)$; $t_{16} = u_{11}(t_9 + t_{10} - t_5 - f_3 - t_7 - t_{11})$; $t_{17} = u_{12}(t_9 + t_{10} - t_5 - f_3 - t_7 - t_{11})$; $s_3 = (t_{12} + t_5 + f_3 + t_7 + t_{11} - t_9 - t_{10})^2$; $s_4 = (t_4 + 1 - 2t_7 - t_8)^2$, $t_{18} = (t_2 - 2t_6)(t_{15} - t_{16})$; $t_{19} = (t_{12} + t_{13} + t_5 + f_3 + t_7 + t_{11} - t_9 - t_{10} - t_{14})(s_3 - t_{18})$; $t_{20} = (t_2 - 2t_6)(-t_{11} - t_{17})$; $t_{21} = (t_4 + 1 - 2t_7 - t_8)(t_5 + t_{12} + t_7 + f_3 + t_{11} - t_9 - t_{10})$; $t_{22} = (t_{20} - 2t_{21})(-t_{11} - t_{17})$, $t_{23} = (t_{15} - t_{16})s_4$; $res_1 = t_{19} + t_{22} + t_{23}$; $t_{24} = (t_2 - 2t_6)(t_{13} + t_{12} + t_7 + t_{11} + t_5 + f_3 - t_9 - t_{10} - t_{14})$; $inv_0 = t_{24} - s_4$, $t_{25} = u_{12} \cdot inv_0$; $t_{26} = u_{12}(t_{20} - t_{21})$, $t_{27} = u_{11} \cdot inv_0$; $inv_1 = t_{25} + t_{21} - t_{20}$, $inv_2 = t_{27} + t_{18} - t_{26} - s_3$; $z_1 = inv_0x^2 + inv_1x + inv_2$	16M+2SQ
3	compute the cubic $w = y^2 + sy + t$ $t_{28} = v_{12}v_{11}$, $t_{29} = v_{11}v_{10}$, $s_5 = v_{10}^2$; $t_{30} = u_{12}s_1$, $t_{31} = u_{11}s_1$, $t_{32} = u_{12}(t_{30} - 2t_{28})$; $t_{33} = (u_{11} + u_{10})(s_1 + 2t_{28} - t_{30})$; $t_{34} = u_{10}(t_{30} - 2t_{28})$; $t_{35} = (t_{32} + 2t_3 + s_2 - t_{31})inv_0$; $t_{36} = (2t_{29} + t_{31} - t_{33} - t_{34})inv_1$; $t_{37} = (s_5 + t_{34})inv_2$; $t_{38} = (t_{32} + s_2 + 2t_3 + 2t_{29} - t_{33} - t_{34})(inv_0 + inv_1)$; $t_{39} = (t_{32} + t_{34} + s_2 + s_5 + 2t_3 - t_{31})(inv_0 + inv_2)$; $t_{40} = (t_{31} + s_5 + 2t_{29} - t_{33})(inv_1 + inv_2)$; $t_{41} = u_{12}t_{35}$, $t_{42} = u_{11}t_{35}$; $t_{43} = u_{12}(t_{41} + t_{36} + t_{35} - t_{38})$; $t_{44} = (u_{11} + u_{10})(t_{38} - t_{41} - t_{36})$; $t_{45} = u_{10}(t_{41} + t_{36} + t_{35} - t_{38})$; $r_0 = t_{43} + t_{39} + t_{36} - t_{42} - t_{35} - t_{37}$; $r_1 = t_{40} + t_{42} - t_{36} - t_{37} - t_{44} - t_{45}$; $r_2 = t_{37} + t_{45}$, $t_{46} = res_1r_0$, $t_{47} = r_0s_1$; $t_{48} = t_{47}res_1$, $t_{49} = -2res_1v_{12}$, $t_{50} = 3r_1s_1$; $t_{51} = 3t_{47}u_{12}$, $\gamma_1 = t_{51} - t_{49} - t_{50}$); $t_{52} = res_1\gamma_1$, $t_{53} = -t_{46}v_{11}$, $t_{54} = -t_{46}v_{10}$; $t_{55} = r_1\gamma_1$, $t_{56} = 3r_2t_{47}$, $t_{57} = r_2\gamma_1$; $t_{58} = 3t_{47}u_{11}$, $t_{59} = 3t_{47}u_{10}$; $t_{60} = t_{58}r_0$, $t_{61} = t_{59}r_0$; $\lambda_1 = 3(2t_{53} + t_{55} + t_{56} - t_{60})$; $\mu_1 = 3(2t_{54} + t_{57} - t_{61})$, $t_{62} = -3t_{46}v_{12}$; $t_{63} = -(v_{12} + v_{11})(\lambda_1 - t_{62} - 3t_{53})$; $t_{64} = -v_{11}(\lambda_1 - 3t_{53})$; $t_{65} = -(v_{12} + v_{10})(\mu_1 - t_{62} - 3t_{54})$; $t_{66} = -v_{10}(\mu_1 - 3t_{54})$; $t_{67} = -(v_{11} + v_{10})(\lambda_1 + \mu_1 - 3t_{53} - 3t_{54})$; $t_{68} = 3t_{48}(u_{12} + u_{11})$, $t_{69} = 3t_{48}u_{11}$; $t_{70} = (u_{12} + u_{10})t_{52}$, $t_{71} = u_{10}t_{52}$; $t_{72} = (u_{11} + u_{10})(3t_{48} + t_{52})$; $B_0 = t_{52} + t_{68} + t_{63} + 3t_{48} - t_{69} - t_{64}$; $B_1 = t_{70} + t_{69} + t_{65} + t_{64} + 3t_{48} - t_{71} - t_{66}$; $B_2 = t_{72} + t_{67} - t_{69} - t_{71} - t_{64} - t_{66}$; $B_3 = t_{71} + t_{66}$, $t_{73} = 3t_{46}B_0$, $i_1 = (t_{73})^{-1}$; $t_{74} = i_1B_0$, $t_{75} = 3t_{46}i_1$, $t_{76} = 3t_{46}t_{75}$; $t_{77} = t_{75}B_1$, $t_{78} = t_{75}B_2$, $t_{79} = t_{75}B_3$; $t_{80} = t_{74}\lambda_1$, $t_{81} = t_{74}\mu_1$, $t_{82} = t_{74}B_0$; $t_{83} = t_{74}B_1$, $t_{84} = t_{74}B_2$, $t_{85} = t_{74}B_3$; $w = y^2 + (t_{80}x + t_{81})y + t_{82}x^3 + t_{83}x^2 + t_{84}x + t_{85}$	58M+1SQ+1I

4	compute $res(w, C, y)$ $s_6 = t_{77}^2$, $t_{86} = t_{77}(6t_{78} + s_6)$, $s_7 = t_{80}^2$; $s_8 = (t_{80} + t_{81})^2$, $s_9 = t_{81}^2$, $t_{87} = t_{80}t_{82}$; $t_{88} = t_{80}(s_7 - 3t_{83})$, $t_{89} = t_{81}t_{82}$, $t_{90} = -3f_3t_{87}$; $t_{91} = t_{80}(s_8 - 3t_{84} - s_7 - s_9)$, $t_{92} = t_{81}(s_7 - 3t_{83})$; $t_{93} = f_3t_{88}$, $t_{94} = -3f_2t_{87}$, $t_{95} = -3f_3t_{89}$, $s_{10} = t_{76}^2$; $t_{96} = t_{76}s_{10}$, $t_{97} = t_{96}(1 - 3t_{87})$; $t_{98} = t_{96}(t_{88} + t_{90} + 2f_3 - 3t_{89})$; $t_{99} = t_{96}(t_{91} + t_{92} + t_{93} + t_{94} + t_{95} + 2f_2 + f_3^2)$;	14M+5SQ
5	compute u_{-2D_1} $s_{11} = u_{12}^2$, $t_{100} = u_{12}u_{11}$; $t_{101} = (2u_{11} + 2u_{10} + 2t_{100} + s_{11})(1 + t_{97} + 3t_{77} - 2u_{12})$; $t_{102} = (2u_{10} + 2t_{100})(t_{97} + 3t_{77} - 2u_{12})$; $c_1 = t_{97} + 3t_{77} - 2u_{12}$, $t_{103} = 2u_{12}c_1$; $c_2 = t_{98} + 3t_{78} + 3s_6 - s_{11} - t_{103} - 2u_{11}$; $t_{104} = 2u_{12}c_2$; $c_3 = 2u_{11} + s_{11} + t_{102} + t_{99} + t_{86} + 3t_{79} - t_{104} - t_{101}$; $u_{-(2D_1)} = x^3 + c_1x^2 + c_2x + c_3$	5M+1SQ
6	compute $res(t - s^2, u_{-2D_1}, x)$ and precomputations for v_{2D_1} $t_{105} = c_3t_{82}$, $t_{106} = c_1t_{82}$, $t_{107} = c_2t_{82}$; $t_{108} = c_2(t_{83} - s_7)$, $t_{109} = c_1(t_{84} + s_7 + s_9 - s_8)$; $t_{110} = c_3(t_{84} + s_7 + s_9 - s_8)$, $t_{111} = c_2(t_{85} - s_9)$; $t_{112} = c_3(t_{83} - s_7)$, $t_{113} = c_1(t_{85} - s_9)$; $s_{12} = (t_{105} + s_9 - t_{85})^2$; $s_{13} = (t_{107} + s_8 - t_{84} - s_7 - s_9)^2$; $t_{114} = (t_{106} + s_7 - t_{83})(t_{110} - t_{111})$; $t_{115} = (t_{105} + t_{108} + s_9 - t_{85} - t_{109})(s_{12} - t_{114})$; $t_{116} = (t_{112} - t_{113})(t_{106} + s_7 - t_{83})$; $t_{117} = (t_{107} + s_8 - t_{84} - s_7 - s_9)(t_{105} + s_9 - t_{85})$; $t_{118} = (t_{112} - t_{113})(t_{116} - 2t_{117})$; $t_{119} = (t_{110} - t_{111})s_{13}$, $res_2 = t_{115} + t_{118} + t_{119}$; $t_{120} = (t_{108} + s_9 + t_{105} - t_{109} - t_{85})(t_{106} - t_{83} + s_7)$; $j_0 = t_{120} - s_{13}$, $t_{121} = j_0 \cdot c_1$; $t_{122} = c_1(t_{116} - t_{117})$, $t_{123} = j_0 \cdot c_2$; $j_1 = t_{121} + t_{117} - t_{116}$, $j_2 = t_{123} + t_{114} - t_{122} - s_{12}$; $t_{124} = t_{80}(t_{83} + t_{84})$, $t_{125} = t_{80}t_{84}$; $t_{126} = t_{81}(t_{83} + t_{85})$, $t_{127} = t_{81}t_{85}$; $t_{128} = (t_{80} + t_{81})(t_{84} + t_{85})$, $t_{129} = c_1(1 - t_{87})$; $t_{130} = (t_{129} + t_{89} + t_{124} - f_3 - t_{125})c_1$; $t_{131} = c_2(1 - t_{87})$; $t_{132} = (c_2 + c_3)(1 + f_3 + t_{125} - t_{87} - t_{129} - t_{89} - t_{124})$; $t_{133} = c_3(t_{129} + t_{89} + t_{124} - f_3 - t_{125})$; $t_{134} = (t_{130} + f_2 + t_{127} - t_{131} - t_{126} - t_{125})j_0$; $t_{135} = (j_0 + j_1)(t_{130} + f_2 + f_1 + 2t_{127} - t_{126} - t_{128} - t_{132} - t_{133})$; $t_{136} = (f_1 + t_{125} + t_{127} + t_{131} - t_{128} - t_{132} - t_{133})j_1$; $t_{137} = (j_0 + j_2)(t_{130} + t_{133} + f_2 + f_0 - t_{131} - t_{126} - t_{125})$; $t_{138} = (f_0 + t_{133} - t_{127})j_2$; $t_{139} = (j_1 + j_2)(f_1 + f_0 + t_{125} + t_{131} - t_{128} - t_{132})$; $t_{140} = t_{134}c_1$, $t_{141} = c_2t_{134}$; $t_{142} = c_1(t_{140} + t_{134} + t_{136} - t_{135})$; $t_{143} = (c_2 + c_3)(t_{135} - t_{140} - t_{136})$; $t_{144} = c_3(t_{140} + t_{134} + t_{136} - t_{135})$;	40M+2SQ
7	compute v_{2D_1} $t_{145} = res_2(t_{142} + t_{137} + t_{136} - t_{141} - t_{134} - t_{138})$; $i_2 = (t_{145})^{-1}$; $t_{146} = i_2(t_{142} + t_{137} + t_{136} - t_{141} - t_{134} - t_{138})$; $t_{147} = t_{146}(t_{142} + t_{137} + t_{136} - t_{141} - t_{134} - t_{138})$; $t_{148} = t_{146}(t_{139} + t_{141} - t_{136} - t_{138} - t_{143} - t_{144})$; $t_{149} = t_{146}(t_{138} + t_{144})$; $v_2' = -t_{147}$, $v_1' = -t_{148}$, $v_0' = -t_{149}$;	5M+1I
8	compute u_{2D_1} $s_{14} = res_2^2$, $t_{150} = i_2s_{14}$, $s_{15} = t_{150}^2$; $t_{151} = t_{150}s_{15}$, $t_{152} = t_{150}t_{148}$, $s_{16} = t_{152}^2$; $t_{153} = t_{150}t_{149}$, $t_{154} = t_{152}(s_{16} + 6t_{153})$; $t_{155} = t_{151}f_3$, $t_{156} = c_1(3t_{152} - c_1)$; $d_1 = 3t_{152} - c_1$, $d_2 = 3t_{153} + 3s_{16} + t_{151} - c_2 - t_{156}$; $t_{157} = c_1d_2$, $t_{158} = c_2(3t_{152} - c_1)$; $d_3 = t_{154} + t_{155} - t_{157} - c_3 - t_{158}$;	9M+3SQ
total		158M, 16S, 2I

Undeniable Signatures Based on Characters: How to Sign with One Bit

Jean Monnerat* and Serge Vaudenay

Swiss Federal Institute of Technology (EPFL) - LASEC
http://lasecwww.epfl.ch

Abstract. We present a new undeniable signature scheme which is based on the computation of characters. Our signature scheme offers the advantage of having an arbitrarily short signature. Its asymptotic complexity is attractive: the asymptotic complexity of all algorithms (even the key setup) are quadratic in the size of the modulus n in bits when the other parameters are fixed. The practical complexity can be quite low depending on parameter and variant choices. We present also a proof of security of our scheme containing the standard security requirements of an undeniable signature.

Key words: Undeniable Signatures, Residue Characters.

1 Introduction

The concept of undeniable signature has been first introduced in 1989 by Chaum and van Antwerpen [6]. This kind of signature is similar to a classical digital signature except that one has to interact with the signer in order to be convinced of the validity of this one. This property offers the advantage of avoiding that any entity can verify the validity of a signature. In fact, limiting this universal verifiability (as it is in the case of a classical digital signature) is desirable in certain circumstances e.g. for privacy reasons. Here, the signer can control how the verification spreads in a community.

To be complete, an undeniable signature should be composed of three main components that are the signature generation, the confirmation protocol and the denial protocol. The role of the confirmation protocol is to allow the signer to prove the validity of a given signature. Conversely, the denial protocol allows a signer (prover) to prove the invalidity of a given signature. It is important to keep in mind that a failure in the confirmation protocol is not a proof of the invalidity of a signature but could be only due to a lack of cooperation from the prover. A similar argument holds also for the denial protocol. So, the confirmation resp. denial protocol is only used to prove the validity resp. invalidity of a signature.

Since their introduction, undeniable signatures received a certain attention and several papers related to them have been published. We give here a list of

* Supported in part by a grant of the Swiss National Science Foundation, 200021-101453/1.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 69–85, 2004.

some of them, [3,4,5,8,9,10,15]. It turns out that almost all of the undeniable signature schemes are based on the discrete logarithm. In [10], Gennaro et al. presented an undeniable signature based on RSA. In this paper, we propose a new undeniable signature that is based on another type of problems, namely the ability of computing a character on $\mathbb{Z}_n^*$. This corresponds actually to a generalization of the quadratic residuosity problem. In the present work, we focus our study on the characters of order 2,3 and 4. Note that the characters of order 3 have already been used in some public-key cryptosystems, e.g. [17] as well as more general characters, e.g. [18].

In section 2, we survey the mathematical theory of the characters on $\mathbb{Z}_n^*$. Section 3 is dedicated to the study of some problems related to the security of our scheme, in particular for cases of order $d = 2, 3, 4$. The new scheme is presented in the section 4. Section 5 is devoted to the security of our scheme. We provide some proofs of some security properties such as the resistance against existential forgery of our scheme or the soundness of the confirmation and denial protocol. Section 6 concludes the article.

2 Characters on $\mathbb{Z}_n^*$

In this section, we introduce the notion of multiplicative characters. The order 2, 3 and 4 cases will be exposed in the following subsections.

Definition 1. *Let n be an integer. A character χ on $\mathbb{Z}_n^*$ is a map from $\mathbb{Z}_n^*$ to $\mathbb{C} - \{0\}$ satisfying $\chi(ab) = \chi(a)\chi(b)$ for all $a, b \in \mathbb{Z}_n^*$.*

From this definition, we can quickly deduce that $\chi(1) = 1$ and that the value $\chi(a)$ is always a $(\lambda(n))^{\text{th}}$ root of the unity for all $a \in \mathbb{Z}_n^*$, where $\lambda(n)$ denotes the Carmichael function. We can also define a group structure on the set of characters on $\mathbb{Z}_n^*$. In this group, the product (group operation) $\chi_1\chi_2$ of the two characters χ_1 and χ_2 represents the map $a \longmapsto \chi_1(a)\chi_2(a)$ and the inverse χ^{-1} maps each element a to $\chi(a)^{-1}$.

Proposition 2. *Let p be a prime and d an integer such that $d|p-1$.*

1. *The group of characters defined on $\mathbb{Z}_p^*$ is a cyclic group of order $p-1$.*
2. *The characters on $\mathbb{Z}_p^*$ of order dividing d form a cyclic subgroup of order d.*

A proof of this proposition can be found at the beginning of the chapter 8 of [12].

The second part of this proposition is especially interesting for us because we will consider characters of small order (e.g. 2, 3, 4) defined on $\mathbb{Z}_n^*$ for n large. We notice also that a character of order d maps the elements of $\mathbb{Z}_p^*$ to the set $\{\zeta_d^j \mid 0 \leq j \leq d-1\}$ where ζ_d denotes the unit $e^{2\pi i/d}$ and $i := \sqrt{-1}$.

We provide a way to define certain multiplicative characters on $\mathbb{Z}_n^*$ for a n being the product of two special primes. Since $\mathbb{Z}_n^*$ is not cyclic, using the above definition to this case is not suitable. It is more natural for our purposes to define such characters in the similar way as the Jacobi symbol is defined from the Legendre symbol. First, assume we are given an integer d and two different

primes p, q such that $d|p-1$ and $d|q-1$. From two characters χ_1 and χ_2 of order d defined on $\mathbb{Z}_p^*$ respectively $\mathbb{Z}_q^*$, we define a character η of order d in the following way $\eta(a) := \chi_1(a \bmod p) \cdot \chi_2(a \bmod q)$.

For each character χ of order d we will sometimes associate a logarithm function denoted as $\log_\chi$. For an element $a \in \mathbb{Z}_n^*$, we know that $\chi(a)$ is of the form ζ_d^j for a $j \in \{0, 1, \ldots d-1\}$. We define $\log_\chi(a)$ equal to this j.

In the following subsections we present some complements that are specific to the cases $d = 2, 3, 4$. For more details, we refer to Ireland and Rosen [12].

2.1 Characters of Order 2

Let p be an odd prime number. By Proposition 2, we know that there are only two characters of order 2, namely the trivial character ϵ that maps every elements to 1 and the Legendre symbol. We recall that the Legendre symbol (a/p) for an integer a with $(a, p) = 1$ is 1 if a is congruent to a square modulo p (quadratic residue) and -1 if it is not the case (quadratic non-residue). It turns out that there are $\frac{p-1}{2}$ quadratic residues resp. non quadratic residues in $\mathbb{Z}_p^*$.

For an odd integer n, the Jacobi symbol (a/n) for an $a \in \mathbb{Z}$ s.t. $(a, n) = 1$ is defined as $(a/n) = (a/p_1)^{i_1} \cdot (a/p_2)^{i_2} \cdots (a/p_k)^{i_k}$ where the factorization into primes of n is $p_1^{i_1} \cdots p_k^{i_k}$. Some additional properties are given below.

Proposition 3. *Let p be an odd prime, $a, b \in \mathbb{Z}$ and an odd $n \in \mathbb{Z}$. Then*

1. $a^{(p-1)/2} \equiv (a/p) \pmod{p}$.
2. $(ab/n) = (a/n)(b/n)$.
3. *If* $a \equiv b \pmod{n}$, *then* $(a/n) = (b/n)$.
4. *(Quadratic Reciprocity)* $(a/b)(b/a) = (-1)^{(\frac{a-1}{2})(\frac{b-1}{2})}$ *for a and b odd.*
5. $(2/n) = (-1)^{\frac{(p^2-1)}{8}}$.

Let us consider a modulus $n = pq$. From the above discussion we deduce that the complete list of characters of order 2 on $\mathbb{Z}_n^*$ is $(\cdot/p)$, $(\cdot/q)$, $(\cdot/n)$ and the trivial character. Note that the properties given in Proposition 3 are used in order to compute the Jacobi symbol in a time complexity of $\mathcal{O}(\log(n)^2)$.

2.2 Characters of Order 3

Here, we introduce the ring of Eisenstein integers. Indeed, this ring is the natural structure to study the characters of order 3 or the cubic residuosity. Most of the results below are taken from [12].

In what follows, ω will always denote the complex number $\frac{-1+\sqrt{-3}}{2}$. We define the ring of the Eisenstein integers as the set $\mathbb{Z}[\omega] := \{a + b\omega | a, b \in \mathbb{Z}\}$ with the classical operations (addition, multiplication) of $\mathbb{C}$. We notice that ω is a non trivial cubic root of 1 and satisfies $\omega^2 + \omega + 1 = 0$.

For an element $\alpha \in \mathbb{Z}[\omega]$, we define the norm $N(\alpha) = \alpha\bar{\alpha}$, where $\bar{\alpha}$ denotes the complex conjugate of α. This is the classical (squared) norm induced by the complex plane. From the definition, we have $N(a + b\omega) = a^2 - ab + b^2$.

It can be shown that $\mathbb{Z}[\omega]$ is a unique factorization domain i.e. every elements can be decomposed in a product of irreducible elements uniquely up to a unit element. We can also call the irreducible elements the prime elements of $\mathbb{Z}[\omega]$. To avoid some confusion a prime of $\mathbb{Z}$ will be called a rational prime if the context is not clear. The units are the invertible elements and in this case all have a norm equal to one. Hence, the units of $\mathbb{Z}[\omega]$ are ± 1, $\pm\omega$, $\pm\omega^2$. All prime numbers of $\mathbb{Z}[\omega]$ are classified below.

Proposition 4. *The following statements describe all primes of* $\mathbb{Z}[\omega]$.

1. *Let* p *be a rational prime s. t.* $p \equiv 1 \pmod 3$. *There exists a prime* π *s. t.* $N(\pi) = \pi\bar{\pi} = p$.
2. *If* q *is a rational prime s. t.* $q \equiv 2 \pmod 3$, *then* q *is also a prime in* $\mathbb{Z}[\omega]$.
3. $1 - \omega$ *is prime and* $N(1-\omega) = 3$.

The ideal generated by a $\sigma \in \mathbb{Z}[\omega]$ is denoted by (σ) and is equal to $\sigma \cdot \mathbb{Z}[\omega]$.

Proposition 5. *Let* π *be a prime in* $\mathbb{Z}[\omega]$. *Then* $\mathbb{Z}[\omega]/(\pi)$ *is a finite field with* $N(\pi)$ *elements.*

We can also prove that the set $\{a+b\omega | 0 \leq a, b \leq q\}$ resp. $\{0, 1, 2 \ldots, p-1\}$ form all representatives of the residue class field in the case where $q \equiv 2 \pmod 3$ resp. $p \equiv 1 \pmod 3$. We can also prove that for a prime π s.t. $N(\pi) \neq 3$ and $\alpha \in \mathbb{Z}[\omega]$ s.t. $\alpha \not\equiv 0 \pmod \pi$, we have $\alpha^{\frac{N(\pi)-1}{3}} \equiv \omega^i \pmod \pi$ for an $i \in \{0, 1, 2\}$. Here, ω^i is called the cubic residue character of α modulo π and is denoted as $(\alpha/\pi)_3$ or as $\chi_\pi(\alpha)$. If $\alpha \equiv 0 \pmod \pi$, we set $\chi_\pi(\alpha) = 0$.

Let α and β be in $\mathbb{Z}[\omega]$. Suppose the prime factorization of β is $u\prod_{i=1}^{k} \pi_i^{e_i}$ where $N(\pi_i) \neq 3$ for all $1 \leq i \leq k$ and u is a unit. Then the Jacobi-like symbol $(\alpha/\beta)_3$ is defined as $\prod_{i=1}^{k}(\alpha/\pi_i)_3^{e_i}$. In order to formulate the law of cubic reciprocity, we have to introduce the concept of primary. We say that an element α of $\mathbb{Z}[\omega]$ is primary iff $\alpha \equiv -1 \pmod 3$. Note that the term "primary" does not only apply to prime number[1]. Every elements possess exactly one associate that is primary. (An associate of an element σ is an element that is of the form $u\sigma$ for a unit u.)

Proposition 6. *Let* π *be a prime s.t.* $N(\pi) \neq 3$ *and* α, β, $\gamma \in \mathbb{Z}[\omega]$. *Let* $\sigma = 3(A + B\omega) - 1$ *be a primary with* $A, B \in \mathbb{Z}$.

1. $(\alpha/\pi)_3 = 1$ *iff* $x^3 \equiv \alpha \pmod \pi$ *is solvable, i.e., iff* α *is a cubic residue.*
2. $(\alpha\beta/\gamma)_3 = (\alpha/\gamma)_3(\beta/\gamma)_3$.
3. $\alpha \equiv \beta \pmod \gamma \Longrightarrow (\alpha/\gamma)_3 = (\beta/\gamma)_3$.
4. *(Law of Cubic Reciprocity) If* α *and* β *are primary. Then* $(\alpha/\beta)_3 = (\beta/\alpha)_3$.
5. $(\omega/\sigma)_3 = \omega^{A+B}$.
6. $(1-\omega/\sigma)_3 = \omega^{2A}$.

[1] The analog notion of "primary" in $\mathbb{Z}$ is the notion of "negative" number.

We are now in the position to define the characters of order 3 on $\mathbb{Z}_p^*$ for a rational prime p and their extensions on a composite modulus that is a Jacobi-like symbol. We consider only the case where $p \equiv 1 \pmod 3$, since the characters are not trivial only in this case. Set $p = \pi\bar{\pi}$. Recall first that the field $\mathbb{Z}[\omega]/(\pi)$ can be represented by $\mathbb{Z}_p^*$ since the set $\{0, 1 \dots p-1\}$ contains all representatives and the multiplications are equivalent in the two cases. Thus, the cubic residue characters χ_π is completely defined on $\mathbb{Z}_p^*$. We directly deduce that χ_π^2 is another non trivial character of order 3 and is even equal to $\chi_{\bar{\pi}}$ on the rational integers. Let p, q be two different rational primes such that $p \equiv q \equiv 1 \pmod 3$ and $\pi, \sigma \in \mathbb{Z}[\omega]$ such that $N(\pi) = p$ and $N(\sigma) = q$. Let $n = pq$, the character on $\mathbb{Z}_n^*$ produced by χ_π and χ_σ is denoted by $\chi_{\pi\sigma}$ and is defined as $\chi_{\pi\sigma}(a) = \chi_\pi(a) \cdot \chi_\sigma(a)$. The other characters are defined exactly in the same multiplicative way. There are 8 non trivial characters of order 3 defined on $\mathbb{Z}_n^*$, namely χ_π, $\chi_{\bar{\pi}}$, χ_σ, $\chi_{\bar{\sigma}}$, $\chi_{\pi\sigma}$, $\chi_{\bar{\pi}\sigma}$, $\chi_{\pi\bar{\sigma}}$ and $\chi_{\bar{\pi}\bar{\sigma}}$.

Here, we explain how to find these characters and how they can be computed. The first statement consists of finding a prime $\pi \in \mathbb{Z}[\omega]$ such that $N(\pi) = p \equiv 1 \bmod 3$ for a rational prime p. We assume here some knowledge on the algorithms of Tonelli and Cornacchia (For more details see Cohen [7]).

For a given p, we have to find an element $a+b\omega \in \mathbb{Z}[\omega]$ such that $a^2-ab+b^2 = p$. This is equivalent to $(a-\frac{b}{2})^2 + \frac{3b^2}{4} = p$. By introducing the two new variables $s = a - \frac{b}{2}$ and $t = \frac{b}{2}$, we obtain $s^2 + 3t^2 = p$ for $s, t \in \mathbb{Z}$. Now, it suffices to apply the algorithm of Cornacchia to solve this equation in s and t. This algorithm consists of finding an $x \in \mathbb{Z}$ such that $x^2 \equiv -3 \pmod p$ (apply algorithm of Tonelli) and then applying the Euclid algorithm to x and p until we get the first rest term r_n such that $r_n^2 < p$. A solution is given by setting $s = r_n$.

Suppose we have a character χ_α where α can be for example $\pi\sigma$ or $\pi\bar{\sigma}$. The computation of a residue character $(\sigma/\alpha)_3$ can be done using a similar technique to the computation of the Jacobi symbol in the context of quadratic residuosity. Indeed, this consists of reducing $\sigma \bmod \alpha$ by an Euclidean division in $\mathbb{Z}[\omega]$ and then applying the cubic reciprocity law to exchange the two elements of the character. This last step can be done only after having extracted some units in order that α and σ become primary. Then by iterating this operation, we reduce the size of the elements involved in the cubic residue character until this one becomes trivial. Note that the asymptotic complexity of the computation is $\mathcal{O}(\log(n)^3)$ using standard arithmetic and $\mathcal{O}(\log(n)^2 \log\log(n) \log\log\log(n))$ using fast arithmetic. This is almost the same order of magnitude as the classical Jacobi symbol that is $\mathcal{O}(\log(n)^2)$ (See Cohen [7] p. 31). For more details about this algorithm and its complexity we refer to Scheidler [17].

2.3 Characters of Order 4

Studying the characters of order 4 consists principally of the theory of bi-quadratic residuosity. This one is quite similar to that of cubic residuosity and is done in the ring of Gaussian integers $\mathbb{Z}[i]$. A rational prime p of the form $p \equiv 1 \pmod 4$ is the norm of a prime π in $\mathbb{Z}[i]$. The field $\mathbb{Z}[i]/(\pi)$ has the set

of representatives $\{0, 1 \ldots p-1\}$ and is identical to $\mathbb{Z}_p$. The biquadratic residue character of an $\alpha \in \mathbb{Z}[i]$ is defined as $\chi_\pi(\alpha) := i^j$ where $j \in \{0,1,2,3\}$ and such that $\alpha^{(N(\pi)-1)/4} \equiv i^j \pmod{\pi}$. Moreover, this character generates the two other nontrivial characters of order 4. Note also that the square of χ_π is equal to the quadratic residue character χ_p. We can also define a Jacobi-like symbol in this context similarly to that in the theory of characters of order 3. Moreover, there is also a law of reciprocity in a similarly way as before.

2.4 Characters of Higher Orders

It is possible to extend our character constructions to some orders greater than 4. By introducing a power residue symbol defined on the integers of a cyclotomic field. A general treatment of these cases would be beyond the scope of this paper. Moreover, the computation seems to be more difficult to deal with and the ring of these integers becomes a non unique factorization domain when the order is large. Since such a ring is not a principal ideal domain, we should work with ideals that are generated by more than one element. However, we do not loose the existence of the reciprocity laws, namely there exists a so called Kummer's reciprocity law (see [14]).

3 On the Hardness of Related Problems

Here we expose some different computational problems that will be related with the security of our scheme. In particular, we focus this treatment to the case of characters of order $d \in \{2,3,4\}$.

For two problems $\mathbf{P}$ and $\mathbf{P}'$, we use the Karp reduction, i.e. we say that $\mathbf{P}$ is at most as hard as $\mathbf{P}'$ if the problem $\mathbf{P}$ can be solved in a polynomial time by using one access to an oracle $\mathcal{O}_{P'}$ that can solve $\mathbf{P}'$. We will denote this as $\mathbf{P} \leq \mathbf{P}'$. Moreover, this is also equivalent to say that $\mathbf{P}'$ is at least as hard as P. We say also that two problems $\mathbf{P}$ and $\mathbf{P}'$ are equivalent if $\mathbf{P} \leq \mathbf{P}'$ and $\mathbf{P}' \leq \mathbf{P}$ are satisfied. We denote this property as $\mathbf{P} \equiv \mathbf{P}'$.

Let θ be a dth primitive root of 1 in $\mathbb{C}$, where d is typically equal to 2,3,4. Below we expose the different problems.

FACT. For a given $n \in \mathbb{Z}$, find the factorization of n in $\mathbb{Z}$.

CYCLOFACT^d. Let σ be an element of $\mathbb{Z}[\theta]$. Find the factorization of σ.

ROOT(–3). Let $n \in \mathbb{Z}$ be such that -3 is a quadratic residue modulo n. Given n, find an $u \in \mathbb{Z}$ such that $u^2 \equiv -3 \pmod{n}$.

ROOT(–1). Let $n \in \mathbb{Z}$ be such that -1 is a quadratic residue modulo n. Given n, find an $u \in \mathbb{Z}$ such that $u^2 \equiv -1 \pmod{n}$.

FERMAT^d. Let $n \in \mathbb{Z}$ be such that $n = \pi\bar{\pi}$ for a $\pi \in \mathbb{Z}[\theta]$. Given n, find π.

CHARACTER^d. Let $n \in \mathbb{Z}$. Devise an algorithm which given $x \in \mathbb{Z}_n^*$ computes $\chi(x)$ where χ is a *hard character* of order d on $\mathbb{Z}_n^*$.

MOVA^d. Let $n \in \mathbb{Z}$, s be a positive integer and χ a *hard character* of order d on $\mathbb{Z}_n^*$. Given s pairs $(\alpha_i, \chi(\alpha_i))$, where $\alpha_i \in \mathbb{Z}_n^*$ for all $1 \leq i \leq s$ and $x \in \mathbb{Z}_n^*$ compute $\chi(x)$.

Remark. By "hard character" we mean a nontrivial character and for $d = 2$ we also exclude the Jacobi symbol $(\cdot/n)$.

Lemma 7. $\mathbf{FACT} \equiv \mathbf{CYCLOFACT}^d$ *and* $\mathbf{FERMAT}^d \leq \mathbf{CYCLOFACT}^d$ *for* $d = 2, 3, 4$. $\mathbf{FERMAT}^3 \equiv \mathbf{ROOT}(-3)$ *and* $\mathbf{FERMAT}^4 \equiv \mathbf{ROOT}(-1)$.

The proof is given in the appendix A. See also Landrock [13] for another cryptographic application of Fermat numbers (i.e. $\mathbf{FERMAT}^4$ and $\mathbf{ROOT}(-1)$).

$\mathbf{CHARACTER}^d$ plays an important role in the security of our signature. Indeed, the ability of signing will be related to the computation of hard characters when n cannot be factorized. Notice that this is a generalization of the quadratic residuosity problem on which the security of the probabilistic Goldwasser-Micali encryption is based [11]. In practice, we will consider a modulus of the form $n = pq$. For $d = 2$, such characters are simply the Legendre symbols modulo p and q. For $d = 3$, we can use the non trivial characters. For example, $\chi_{\pi\sigma}$ is a case where the security is related to $\mathbf{FERMAT}^3$ since $N(\pi\sigma) = n$. Indeed, an enemy that knows a square root of –3 modulo n would be able to retrieve this character by Lemma 7. Thus, $\mathbf{FERMAT}^3 \geq \mathbf{CHARACTER}^3$ and similarly $\mathbf{FERMAT}^4 \geq \mathbf{CHARACTER}^4$. Note also that $\mathbf{MOVA}^d \leq \mathbf{CYCLOFACT}^d$ but $\mathbf{MOVA}^d \leq \mathbf{CHARACTER}^d$ in some cases only, because the character devising in $\mathbf{CHARACTER}^d$ may be independent from the character required for $\mathbf{MOVA}^d$.

4 Description of the MOVA Scheme

We present here the components of our undeniable signature scheme called "MOVA"[2]

Public Parameters. Let s, t, k, ℓ be some positive integers whose size depend on the required security level of the scheme. We let θ denote a primitive dth root of 1 in $\mathbb{C}$, where $d \in \{2, 3, 4\}$.

Primitives. We assume the existence of two pseudorandom generators $G_1 : \{0,1\}^* \longrightarrow (\mathbb{Z}_n^*)^s$ and $G_2 : \{0,1\}^* \longrightarrow (\mathbb{Z}_n^*)^t$. We also assume the existence of a commitment scheme denoted as $\mathrm{COMMIT} : x \mapsto (< x >, \mathrm{OPEN}_x)$ and $\mathrm{CHECK}(x, < x >, \mathrm{OPEN}_x)$.

Setup. The signer generates an n and a hard character χ of order d on $\mathbb{Z}_n^*$. Then he takes a string $Id \in \{0,1\}^*$ and computes $G_1(Id) = (\alpha_1, \ldots, \alpha_s)$. Finally, he computes the logarithm of the character residues of the α_i's. We set $\Sigma_\alpha := (e_1, \ldots, e_s)$ an element of $\{0, 1 \ldots d-1\}^s$ where $e_i = \log_\chi(\alpha_i)$ for all $1 \leq i \leq s$. If the e_i's do not span $\mathbb{Z}_d$ or $e_i = (\frac{\alpha_i}{n})$ for all $1 \leq i \leq s$ in the $d = 2$ case then restart with another Id.[3] For $d = 3$ or 4 we can either start by generating prime numbers p and q, take $n = pq$, get π such that $\pi\bar{\pi} = n$

[2] "MOVA" is related to the names of the authors of the present paper.

[3] As discussed in Subsection 5.6 an authority could be involved in this scheme in order to tolerate low s parameter.

and set $\chi = (./\pi)_d$, or directly generate $n = \pi\bar{\pi}$ from a random $\pi \in \mathbb{Z}[\theta]$. The latter is performed with smaller complexity but the factorization of n is unknown.

Public Key. $K_P = (n, Id, \Sigma_\alpha)$.

Secret Key. $K_S = \chi$.

Signature generation. Let $m \in \{0,1\}^*$ be a message to sign. The signer generates $G_2(m) = (\beta_1, \ldots, \beta_t)$. Then the signer computes $c_i = \log_\chi(\beta_i)$. The signature of m is Σ, where Σ is defined as

$$\Sigma := (c_1, c_2, \ldots, c_t).$$

Confirmation Protocol. We denote here the prover as P and the verifier as V. The signer is given (m, Σ) that is also public. Here is the sketch of the protocol.

Repeat k times :

1. V picks some values $a_1, a_2, \ldots a_s, b_1, \ldots b_t \in \{0, 1 \ldots d-1\}$ and a $\gamma \in \mathbb{Z}_n^*$ randomly. Set $\delta := \gamma^d \cdot \prod_{i=1}^s \alpha_i^{a_i} \cdot \prod_{i=1}^t \beta_i^{b_i} \bmod n$. V then sends δ to P.
2. P computes $r = \log_\chi(\delta)$ and sends r to V.
3. V checks if $r = \sum_{i=1}^s a_i e_i + \sum_{i=1}^t b_i c_i \bmod d$. If this equality does not hold, V rejects the signature.

For some security reasons, this protocol must include a commitment function. Indeed, we notice that somebody could use this protocol several times in order to sign a message of his choice. This can be easily done by sending the β_i's instead of δ to the prover. A way to prevent against a such attack is to use a commitment function as mentioned in Gennaro and al. [10]. In our confirmation protocol, the modification works in the following way. After having computed r in Step 2., P runs $\mathrm{COMMIT}(r)$ and sends $< r >$ to V and then V sends $\gamma, a_1, \ldots a_s, b_1, \ldots b_s$ to P. The prover checks that $\delta = \gamma^d \cdot \prod_{i=1}^s \alpha_i^{a_i} \cdot \prod_{i=1}^t \beta_i^{b_i} \bmod n$ really holds. Finally, P sends r, OPEN_r to the verifier that can then effect Step 3 and do $\mathrm{CHECK}(r, < r >, \mathrm{OPEN}_r)$.

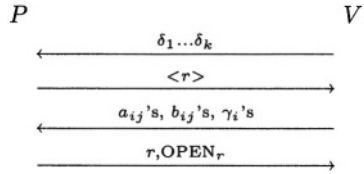


Fig. 1. Confirmation Protocol with commitment

Note that the confirmation protocol can be completely parallelized (see Figure 1). V sends $\delta_1, \ldots \delta_k$ defined as $\delta_i = \gamma_i^d \cdot \prod_{j=1}^s \alpha_j^{a_{ij}} \cdot \prod_{j=1}^t \beta_j^{b_{ij}} \bmod n$ where the a_{ij}'s, b_{ij}'s and γ_i's are picked at random. This protocol continues similarly with $r := (r_1, \ldots, r_k)$ as the prover will commit. Finally, after V has sent the

a_{ij}'s, b_{ij}'s and the γ_i's to P, this one opens the commitment of the values r_i's. Note that V can generate the a_{ij}'s, b_{ij}'s and the γ_i's in a pseudorandom way and send the seed of the pseudorandom generator. This method can considerably decrease the communication complexity.

Denial Protocol. Here, the verifier V is given a message $m \in \{0,1\}^*$ and an alleged non-signature Σ where $\Sigma = (c_1, \ldots, c_t)$. The protocol works as follows.

Repeat ℓ times:

1. The prover picks a matrix $A = (a_{ij}) \in \mathbb{Z}_d^{t \times s}$ at random and a matrix $B = (b_{ij}) \in \mathbb{Z}_d^{t \times t}$ of rank t. He then computes $q_i := \sum_{j=1}^{s} a_{ij} e_j + \sum_{j=1}^{t} b_{ij} c_j$ and $r_i := \sum_{j=1}^{s} a_{ij} e_j + \sum_{j=1}^{t} b_{ij} \log_\chi(\beta_j)$ for all $1 \leq i \leq t$. Set $Q = (q_i)$ and $R = (r_i)$. P computes $\delta_i := \gamma_i^d \cdot \prod_{j=1}^{s} \alpha_j^{a_{ij}} \cdot \prod_{j=1}^{t} \beta_j^{b_{ij}} \bmod n$. He finally runs COMMIT$(\gamma, A, B)$, COMMIT$(R)$ and sends $< \gamma, A, B >$, $< R >$ and the values δ_i's, Q to V.
2. V picks a challenge $u \in \{0,1\}$ at random and sends u to P.
3. If $u = 0$, he sends $\gamma, A, B,$ OPEN$_{(\gamma,A,B)}$ to V. If $u = 1$, he sends $R,$ OPEN$_R$ to V.
4. If $u = 0$, V does CHECK$(\gamma, A, B, < \gamma, A, B >,$ OPEN$_{(\gamma,A,B)})$ and checks that $\delta_i = \gamma_i^d \cdot \prod_{j=1}^{s} \alpha_j^{a_{ij}} \cdot \prod_{j=1}^{t} \beta_j^{b_{ij}} \bmod n$ for all $1 \leq i \leq t$, $q_i = \sum_{j=1}^{s} a_{ij} e_j + \sum_{j=1}^{t} b_{ij} c_j$ for all $1 \leq i \leq t$. If $u = 1$, V does CHECK$(R, < R >,$ OPEN$_R)$ and checks that $Q \neq R$. He then checks that $r_i = \log_\chi(\delta_i)$ for all $1 \leq i \leq t$ by interacting with P in a confirmation protocol on the "signature" R of δ.

5 Security Analysis

Here we analyze the security of our proposed scheme. We do not recall here every security properties suitable for an undeniable signature and refer to [8] and [10].

5.1 Validity of the Public Key

We say that a public key is valid if

1. the set $\{e_1 \ldots e_s\}$ spans $\mathbb{Z}_d$,
2. when $d = 2$ there exists at least one j s.t. $e_j \neq (\frac{\alpha_j}{n})$,
3. the set $\{\alpha_1 \ldots \alpha_s\}$ spans $\mathbb{Z}_n^*/(\mathbb{Z}_n^*)^d$.

If these conditions are fulfilled, we can prove that there exists at most one character χ such that $\chi(\alpha_i) = e_i$ for $1 \leq i \leq s$ and that this character is a hard one of order d. Note that the third condition is the only one which cannot be checked by V. This will be probabilistically satisfied depending on s. The first two are already avoided in the Setup of the scheme. Assuming that G_1 behaves like a random oracle, an analysis of the probability shows that the third condition is not checked with probability $\frac{3}{2^s} - \frac{2}{4^s}$ for $d = 2$ and $\frac{4}{3^s} - \frac{3}{9^s}$ for $d = 3$. For $d = 4$, this probability has magnitude $\mathcal{O}(\frac{1}{2^s})$. See Appendix B for more details on this computation. So, for $d = 3$ and $s = 52$ this probability is approximately 2^{-80}. Thus invalid keys cannot be forged in practice.

5.2 Signature Forgery and Impersonation

In this subsection we show that our signature scheme is resistant to an existential forgery attack and that nobody else than the prover can confirm or deny a given signature.

Let first consider an attacker $\mathcal{A}_1$ living in the model of security of an undeniable signature. In a such model, $\mathcal{A}_1$ is supposed to have access to an oracle able to sign some queried messages, to a second oracle playing the role of the prover in the confirmation protocol and to an oracle able to play the role of the prover in the denial protocol. In fact, by looking at the confirmation protocol and denial protocol and assuming that G_2 is a random oracle, we can see that $\mathcal{A}_1$ does not learn more information in this model than having a random source $\mathcal{S}$ generating some pairs $(\mu, \log_\chi(\mu)) \in \mathbb{Z}_n^* \times \mathbb{Z}_d$. Hence, this attacker reduces to a new attacker $\mathcal{A}_2$ having $\mathcal{S}$ to his disposal. Assuming now that the α_i's generate $\mathbb{Z}_n^*/(\mathbb{Z}_n^*)^d$, an attacker picking some random values $\gamma \in \mathbb{Z}_n^*$, a_i's in $\{0, 1 \dots d-1\}$ and then computing $\gamma \cdot \prod_{i=1}^{s} \alpha_i^{a_i}$ is also able to simulate the source S. Thus, $\mathcal{A}_2$ can be replaced by an attacker $\mathcal{A}_3$ that possesses only the public key. We conclude by saying that any attacker of our scheme will be then considered as $\mathcal{A}_3$. Finally, notice that $\mathcal{A}_3$ is exactly in the situation that corresponds to the assumption of the problem $\mathbf{MOVA}^d$ (see section 3.) .

To prepare these security proofs we first need the following results.

Theorem 8. *Let $\varphi : G \longrightarrow \mathbb{Z}_d$ be a group homomorphism. If one can compute a f such that $\Pr_{x \in G}(f(x) \neq \varphi(x)) \leq \frac{\xi}{12}$ with a constant $\xi < 1$, then one can compute φ in a number of calls to f bounded by a polynomial in* $\log(\#G)$.

We have postponed the proof of this theorem to the appendix C.

Assuming that $\alpha_1 \dots \alpha_s$ span $\mathbb{Z}_n^*/(\mathbb{Z}_n^*)^d$ and using Theorem 8, we show that an entity that is able to confirm or deny a given signature must be able to compute the character, i.e. he possesses the secret key. Indeed, in these two protocols, it is requested to the Prover to evaluate the logarithm of the character on different values (e.g. δ). Passing these tests corresponds to the ability of the computation of $\log_\chi$. More precisely, in the confirmation protocol we can see the Prover as a function that takes on input the value δ depending of the a_i's and b_i's and computes $\log_\chi(\delta)$. We can see this process in one function that is defined on the Abelian group $\mathbb{Z}_n^*$ and whose values lie in $\mathbb{Z}_d$. We see that we can directly apply our above general results to this function, since it satisfies the properties of the function φ of Theorem 8. Thus, an entity that can evaluate this function with a small error probability is able to compute the character χ by Theorem 8.

Corollary 9 (Privacy of Confirmation). *Let Σ be a valid signature associated to a valid public key K_P. If $\mathbf{MOVA}^d$ is hard, then no fake prover can pass the confirmation but with a probability bounded by $(1 - \frac{\xi}{12})^k$ for any $\xi < 1$.*

This corollary protects a user against an impersonation during the confirmation protocol. So, an enemy is not able to confirm a message signed by a given person

without knowing his secret key. The case of the denial protocol is more subtle because the number of characters the prover has really to compute is not fixed. In fact, when $u = 1$ he has a huge probability to pass the test by answering at random. It can happen with probability $2^{-\ell}$, that the prover does not need to compute any character at all. In anyway, he will have to distinguish between $u = 0$ or $u = 1$ in order to pass the test. Thus the probability of success of the enemy is in anyway less than $2^{-\ell}$ since the prover cannot know the value u.

After this discussion and having exposed Theorem 8, we can obviously say that our scheme is resistant against existential forgery.

Corollary 10 (Hardness of existential forgery). *Assuming that* $\mathbf{MOVA}^d$ *is hard and that* G_2 *is a random oracle, then no attacker can forge a valid signature for a message* m *but with a probability bounded by* $(1 - \frac{\xi}{12})^t$ *for any* $\xi < 1$.

5.3 The Confirmation Protocol

We provide below some properties on the security of the confirmation protocol. From now on, $\mathrm{Sign}(m, K_P, P)$ denotes the signature of the message m of the user P possessing the public key K_P.

Proposition 11 (Confirmation protocol).

Completeness. *Let* $\Sigma = Sign(m, K_P, P)$ *be a valid signature. If* P *and* V *follow the Confirmation Protocol, then* V *always accepts the validity of the signature* Σ.

Soundness. *Let* $\Sigma \neq Sign(m, K_P, P)$ *be an invalid signature with respect to* K_P. *Then a cheating Prover* P *can confirm the signature* Σ *with a probability not better than* $\frac{1}{p^k}$, *where* p *is the smallest prime factor of* d.

Zero-Knowledge. *The confirmation protocol is zero-knowledge.*

Proof (Sketch). The completeness is obvious by looking at the protocol.

For the proof of the soundness, we investigate what the behavior of the cheater P should be in order to bypass the confirmation protocol. For sake of simplicity, assume also that the signature Σ differs to $\mathrm{Sign}(m, K_P, P)$ at only one component. W.l.o.g. assume that $c_1 \neq \log_\chi(\beta_1)$, where the term β_1 is the first term of $G_2(m)$. Passing one round of the confirmation protocol is equivalent to be able to find the value $v := \sum_{i=1}^{s} a_i e_i + \sum_{i=1}^{t} b_i c_i \bmod d$ knowing the e_i's, $\log_\chi(\beta_i)$'s and $\log_\chi(\delta)$. Since $v - \log_\chi(\delta) = b_1(c_1 - \log_\chi(\beta_1))$, we deduce that the cheater passes the test iff he can find the value b_1. This is not possible because the value δ can be generated in several different ways, i.e. for several different $\gamma \in \mathbb{Z}_n^*$, a_i's and b_i's. Thus, the d different distributions of the δ corresponding to the d different fixed values b_1 are indistinguishable when d is prime. Otherwise, the assertion remains true when we replace d by p in the worst case. Therefore, he cannot do better than supposing the correct v in a set of at least p elements.

Zero-knowledge: A honest verifier can easily simulate the transcript of the protocol. Since a dishonest verifier has a negligible probability to pass the protocol, our confirmation protocol is therefore zero-knowledge. □

5.4 The Denial Protocol

Proposition 12 (Denial protocol).

Completeness. *Let $\Sigma \neq Sign(m, K_P, P)$ be an invalid signature. If P and V follow the Confirmation Protocol, then V always concludes the invalidity of the signature Σ.*
Soundness. *Let $\Sigma = Sign(m, K_P, P)$ be a valid signature with respect to K_P. Then a cheating Prover P can deny the signature Σ with a probability not greater than $\frac{1}{2^\ell}$.*
Zero-Knowledge. *The denial protocol is zero-knowledge.*

Proof (Sketch). Completeness: It is obvious by examining the denial protocol.

Soundness: First, notice that a cheating prover can easily pass the denial protocol if he would be able to find when $u = 0$ or $u = 1$. Conversely, if he has not this ability, he cannot pass the denial protocol with a probability greater than $\frac{1}{2^\ell}$ if we assume that the soundness of confirmation protocol is perfect.

Zero-knowledge: For $u = 0$ a verifier can trivially simulate the transcript of the protocol (assuming that $< R >$ can be simulated). For $u = 1$ he can pick some a_{ij}'s and γ_i's at random then set $q_i := \sum_{j=1}^{s} a_{ij} e_j$ and $\delta_i := \gamma_i^d \cdot \prod_{j=1}^{s} \alpha_j^{a_{ij}}$. He can pick $R \neq Q$ at random then simulate the protocol. One can easily prove that the generated (δ, Q, R) have the same distribution as in the protocol. He then needs to simulate the confirmation protocol. □

5.5 Complexity

The complexity of the signature generation is the computation of t characters. For the confirmation protocol, the verifier needs about $k \cdot (s+t) \cdot (d-1)/d$ multiplications in $\mathbb{Z}_n^*$ assuming that the values α_i^2, β_i^2, α_i^3, β_i^3 mod n are precomputed. In the same protocol, the prover has to perform k character computations. The denial protocol requires about $\ell \cdot t \cdot (s+t) \cdot (s-1)/s$ modular multiplications and $k \cdot \ell/2$ character computations to the prover. The verifier has to compute $1/2 \cdot (\ell t + k) \cdot (s+t) \cdot (d-1)/d$ modular multiplications[4]. Note that character computation is asymptotically comparable to multiplication in terms of complexity i.e. $\mathcal{O}((\log n)^2)$.

The setup protocol requires the computation of s characters as well as finding the hard character. This step can be realized in two different ways. The first one requires the generation of two primes p,q with a complexity of $\mathcal{O}((\log n)^4)$. The second way (for $d = 3, 4$ only) requires $\mathcal{O}((\log n)^2)$ since we have to pick a large $\pi \in \mathbb{Z}[\theta]$ and compute $n = \pi\bar{\pi}$.

[4] Note that, it is possible to adapt the protocol of [10] in order to reduce the complexity of the denial protocol.

5.6 Key Setup Variants

Here, we discuss some variants of the setup allowing to reduce the size of s. As we have seen, in the first variant the signer selects his own key without any help. The consequence is that s has to be large to ensure the security.

In the second variant, we propose that the signer selects his own key online with the participation of a certificate authority. This allows to reduce the value of s since the signer is limited with the number of attempts. Note also that the complexity of this key setup is similar to the first variant, i.e. the complexity can be quadratic with $d = 3, 4$ and the second way for generating n as discussed in the previous section.

The last variant allows to have a s even lower but requires a greater complexity of the key setup since the signer needs to know the factorization of the modulus n. Here, the signer generates the key itself and proves its validity to the certificate authority or to the verifier. Below, we describe the protocol in which the prover (signer) convinces a verifier (authority) that the α_i's generate $\mathbb{Z}_n^*/(\mathbb{Z}_n^*)^d$.

Repeat m times:

1. The prover picks $\delta_1 \in \mathbb{Z}_n^*$ at random and runs COMMIT(δ_1). He sends $< \delta_1 >$ to the verifier.
2. The verifier picks $\delta_2 \in \mathbb{Z}_n^*$ at random and sends δ_2 to the prover.
3. The prover computes some coefficients $\gamma \in \mathbb{Z}_n^*$, $a_1, \ldots, a_s \in \{0, \ldots d-1\}$ that satisfy $\delta_1\delta_2 \equiv \gamma^d \cdot \prod_{j=1}^{s} \alpha_j^{a_j} \pmod n$ He sends δ_1, OPEN$_{\delta_1}$, $a_1, \ldots a_s$ to the verifier.
4. The verifier runs CHECK$(\delta_1, < \delta_1 >, \text{OPEN}_{\delta_1})$ and checks if $\delta_1 \in \mathbb{Z}_n^*$ and if the equality $\delta_1\delta_2 \equiv \gamma^d \cdot \prod_{j=1}^{s} \alpha_j^{a_j} \pmod n$ holds.

It can be shown that this protocol is complete, sound and zero-knowledge.

5.7 Parameters Choice

Note that our bounds are not tight and that we believe that $\frac{\xi}{12}$ can be replaced by $1 - \frac{1}{d}$ everywhere[5]. Hence, the probability of an impersonation is similar to that of soundness. Since an attacker cannot check the validity or invalidity of a signature offline, the minimal size of the suitable parameters should correspond to a probability of 2^{-20}. The signature can therefore have a length of 20 bits, i.e. $t = 20/(\log_2(d))$. The same probability for the soundness of the confirmation resp. denial protocol, implies that $k = 20/(\log_2(p))$ resp. $\ell = 20$. If the public key is generated offline (first variant of setup), we have to consider a probability of 2^{-80}. Hence, the value of s is 80 for $d = 2, 4$ and $80/(\log_2(3))$ for $d = 3$. Finally, the size of n should be as in RSA, i.e. 1024 bits. For $d = 3$ we get the following size: $s = 52$, $t = 13$, $k = 13$ and $\ell = 20$. If the α_i's are generated online (second variant of setup) which registering the public key to an authority, we

[5] At the time we are wrapping up this paper, we can prove that we can replace $\xi/12$ by $\xi/2$.

can reduce s to $s = 13$. If failure cases are strongly controlled by the authority we can even afford a security level of 2^{-10} and have $s = 6$. If we can further prove that the α_i's span $\mathbb{Z}_n^*/(\mathbb{Z}_n^*)^d$ to authority (third variant of setup) we can shorten s drastically to $s = 2$ using certificates.

For academic purposes, we can propose $d = 2$, $s = 2$, $t = 1$, $k = 20$, $\ell = 20$ (i.e. a signature of only one bit !). An enemy is able to forge a signature with a probability of 1/2 but he would not be able to confirm it. However, the true signer could not deny it.

6 Conclusion

We proposed a new undeniable signature and prove its security. Since the signature does not have to be an element of the size of a modulus, our scheme offers the advantage to sign with short signatures. Moreover, we can see that the complexity of the signature generation, the confirmation and denial protocol is quadratic in the size of n since the most costly operation is a character computation. Furthermore, some key setup variants allow to get quadratic complexity. Another nice property of our protocol is the possibility to confirm several signatures at the same time. For this batch verification, we only need to consider these signatures as a big one.

As a further research, we will extend our scheme to characters of higher order. It would be also worth studying if our scheme can be modified in order to offer some additional advanced properties such as the convertibility or the delegation. In our scheme, we already have a kind of delegation when $d = 3$ or 4. Indeed, the ability to sign, confirm an deny can be delegated by releasing one hard character (i.e. some $\pi \in \mathbb{Z}[\theta]$) to the proxy while the original signer can keep the complete list of characters (i.e. the factorization of n). This property holds for $d \neq 2$ since disclosing one π does not fully disclose the complete factorization of n. In the context of undeniable signature the delegation should not give the possibility for the proxy to sign but only to confirm or deny.

References

1. S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy, *Proof Verification and Hardness of Approximation Problems,* Proc. 33rd IEEE Symp. on Foundations of Computer Science, pp. 14-23, 1992.
2. L. Babai, L. Fortnow, L. Levin and M. Szegedy, *Checking Computations in Polylogarithmic Time,* Proc. 23rd ACM Symp. on Theory of Computing, pp. 21-31, 1991.
3. J. Boyar, D. Chaum, I. Damgård and T. Pedersen, *Convertible Undeniable Signatures,* Advances in Cryptology - Crypto '90, LNCS **537**, pp. 189-205, Springer, 1990.
4. D. Chaum, *Zero-Knowledge Undeniable Signatures,* Advances in Cryptology - Eurocrypt '90, LNCS **473,** pp. 458-464, Springer, 1990.
5. D. Chaum, *Designated Confirmer Signatures,* Advances in Cryptology - Eurocrypt '94, LNCS **950,** pp. 86-91, Springer, 1994.

6. D. Chaum and H. van Antwerpen, *Undeniable Signatures,* Advances in Cryptology - Crypto '89, LNCS **435,** pp. 212-217, Springer, 1989.
7. H. Cohen, *A Course in Computational Algebraic Number Theory,* Graduate Texts in Mathematics **138,** Springer, 2000.
8. I. Dåmgard and T. Pedersen, *New Convertible Undeniable Signatures Schemes,* Advances in Cryptology - Eurocrypt '96, LNCS **1070,** pp. 372-386, Springer, 1996.
9. Y. Desmedt and M. Yung, *Weaknesses of Undeniable Signature Schemes,* Advances in Cryptology - Crypto '91, LNCS **576,** pp. 205-220, Springer, 1991.
10. R. Gennaro, T. Rabin and H. Krawczyk, *RSA-Based Undeniable Signatures,* Journal of Cryptology, **13,** pp. 397-416, Springer, 2000.
11. S. Goldwasser and S. Micali, *Probabilistic Encryption,* Journal of Computer and System Sciences, **28**, pp. 270-299, 1984.
12. K. Ireland and M. Rosen, *A Classical Introduction to Modern Number Theory: Second Edition,* Graduate Texts in Mathematics **84,** Springer, 1990.
13. P. Landrock, *A New Concept in Protocols: Verifiable Computational Delegation,* Security of Protocols, LNCS **1550,** Springer 1998.
14. F. Lemmermeyer, *Reciprocity Laws,* Monographs in Mathematics, Springer, 2000.
15. M. Michels, H. Petersen and P. Horster, *Breaking and Repairing a Convertible Undeniable Signature,* In Proceedings of the 3rd ACM Conference on Computer and Communications Security, pp- 148-152, 1996.
16. P. Nguyen, *La Géométrie des Nombres en Cryptologie,* Thèse de Doctorat.
17. R. Scheidler, *A Public-Key Cryptosystem Using Purely Cubic Fields,* Journal of Cryptology, **11,** pp. 109-124, Springer, 1998.
18. R. Scheidler and H. Williams, *A Public-Key Cryptosystem Utilizing Cyclotomic Fields,* Design, Codes and Cryptography, **6,** pp. 117-131, Kluwer Academic Publishers, 1995.

A Proofs of Some Equivalence Problems

FACT and CYCLOFACT. The case $d = 2$ is trivial. The cases $d = 3$ and $d = 4$ are similar. We concentrate on $d = 3$ here.

$\mathbf{FACT} \leq \mathbf{CYCLOFACT}^3$: Suppose we are given an oracle $\mathcal{O}_{\mathbf{CYCLOFACT}^3}$ that solves the problem $\mathbf{CYCLOFACT}^3$. We compute the factorization of a $n \in \mathbb{Z}$ by calling $\mathcal{O}_{\mathbf{CYCLOFACT}^3}$ on the input n. We then obtain a decomposition of the form $n = u \cdot (1-\omega)^{2i} \cdot \pi_1\pi_2 \ldots \pi_k \cdot q_1 \cdot q_2 \ldots q_l$. By choosing the π_j's that have the same norm and by combining them with u we get some terms of the form $\pi_j\bar{\pi}_j = p_j$, where the p_j's are rational prime integers. Doing the same with $(1-\omega)^{2i}$, provides the term 3^i. After this process, only rational primes will remain in this decomposition, i.e. the factorization of n in $\mathbb{Z}$.

$\mathbf{CYCLOFACT}^3 \leq \mathbf{FACT}$: Here, we have access to the oracle $\mathcal{O}_{\mathbf{FACT}}$ and we have to factorize a $\sigma \in \mathbb{Z}[\omega]$. To this end, we compute $n = \sigma\bar{\sigma}$ and call the oracle $\mathcal{O}_{\mathbf{FACT}}$ on n to obtain the factorization $n = \prod p_i$. Since the rational prime numbers p_i congruent to 2 modulo 3 are also prime in $\mathbb{Z}[\omega]$, it suffices to find the nontrivial primes π_i of the form $\pi_i\bar{\pi}_i \equiv 1 \pmod 3$. To this purpose, we apply the algorithm of subsection 2.2 to the rational primes p_i's congruent to 1 modulo 3. Hence, we obtain the decomposition $p_i = \pi_i\bar{\pi}_i$ of those primes. It

remains to decide which one of π_i or $\bar{\pi}_i$ divides σ. This can be decided by an Euclidean division. Thus, all the non trivial prime divisors of σ are found and therefore its factorization.

FERMAT and ROOT. We can show that **FERMAT**^{3} is equivalent to solve the equation $n = s^2+3t^2$. Then, we can easily see that a solution of this equation gives a square root of –3 modulo n if $(t, n) = 1$, namely $s \cdot t^{-1}$. The converse assertion follows by the fact that a solution s, t is obtained by finding the shortest vector of the lattice $\{(s,t) \in \mathbb{Z}^2 | s \equiv tu \pmod{n}\}$. This can be done by a lattice reduction in dimension two using the reduction algorithm of Gauss (see [16]). Moreover, this algorithm has a polynomially complexity. □

B Probability of Generating $\mathbb{Z}_n^*/(\mathbb{Z}_n^*)^d$

We consider here a modulus of the form $n = pq$, where p and q are two rational primes s.t. $p \equiv q \equiv 1 \bmod d$. We study here the probability for s elements $\alpha_1 \ldots \alpha_s \in \mathbb{Z}_n^*$ picked at random to generate $\mathbb{Z}_n^*/(\mathbb{Z}_n^*)^d$. Observe that this group is isomorphic to $\mathbb{Z}_p^*/(\mathbb{Z}_p^*)^d \times \mathbb{Z}_q^*/(\mathbb{Z}_q^*)^d$ by Chinese Remainder Theorem. Finally, this is also isomorphic to $\mathbb{Z}_d \oplus \mathbb{Z}_d$. Thus, it suffices to compute the probability that s elements of $\mathbb{Z}_d \oplus \mathbb{Z}_d$ generate the whole group.

Case $d = 2$**.** First we observe that $\mathbb{Z}_2^2$ has 3 non trivial subgroups, namely $G_1 := \{(1,0),(0,0)\}$, $G_2 := \{(0,1),(0,0)\}$, $G_3 := \{(1,1),(0,0)\}$. The only possibility of elements to not generate the whole group is to stay always in exactly one of the above subgroup, i.e. to pick always the same nonzero elements and/or the zero elements. This probability is then $\Pr_2 = \frac{1}{4^s} + 3\left(\frac{1}{2^s} - \frac{1}{4^s}\right) = \frac{3}{2^s} - \frac{2}{4^s}$. The first term corresponds to the probability that all elements are equal to zero and the second corresponds that these elements lie in one of the three subgroup without being all equal to zero.

Case $d = 3$**.** This works similarly. The probability is $\frac{1}{9^s} + 4\left(\frac{1}{3^s} - \frac{1}{9^s}\right) = \frac{4}{3^s} - \frac{3}{9^s}$.

Case $d = 4$**.** Here, an exact computation would be more complicated, but the existence of subgroups of order 8 implies that the dominant term in the probability will be of magnitude $(\frac{8}{16})^s = 2^{-s}$. An example of subgroup of order 8 is < (1,2), (2,2) >= {(0,0), (1,2), (2,0), (3,2), (2,2), (3,0), (0,2), (1,0)}.

C Proof of Theorem 8

We first have the following theorem. Its proof is freely inspired from [1,2].

Theorem 13. *Let G be a finite Abelian group and $d|(\#G)$. Let $x_1, \ldots x_r \in G$, $y_1, \ldots y_r \in \mathbb{Z}_d$ and $f : G \longrightarrow \mathbb{Z}_d$. If*

$$\Pr_{\substack{a_1 \ldots a_r \in \mathbb{Z}_d \\ x \in G}} \left(f\left(d \cdot x + \sum_{i=1}^{r} a_i \cdot x_i\right) = \sum_{i=1}^{r} a_i \cdot y_i \right) = 1 - \varepsilon > \frac{1}{2},$$

then there exists a morphism $\varphi : G \longrightarrow \mathbb{Z}_d$ such that $\varphi(x_i) = y_i$ for all $1 \le i \le r$ and $\Pr_{x \in G}(f(x) = \varphi(x)) = 1 - \varepsilon$.

Proof. Let $H := \{(b_1 \dots b_r) \in \mathbb{Z}_d^r \text{ s.t. } \sum_{i=1}^r b_i \cdot x_i \in d \cdot G\}$. Let ε' be such that $\frac{1}{2} > \varepsilon' > \varepsilon > 0$ and let A be the set of all $(a_1 \dots a_r)$ in $\mathbb{Z}_d^r/H$ such that

$$\Pr_{\substack{b_1 \dots b_r \\ x \in G}} [f(d \cdot x + \sum_{i=1}^r (a_i + b_i) \cdot x_i) = \sum_{i=1}^r (a_i + b_i) \cdot y_i] \geq 1 - \varepsilon'\}.$$

We have

$$1 - \varepsilon = \mathop{\mathrm{E}}_{(a_1 \dots a_r) \in \mathbb{Z}_d^r/H} \left(\Pr_{\substack{(b_1 \dots b_r) \in H \\ x \in G}} [f(d \cdot x + \sum_{i=1}^r (a_i + b_i) \cdot x_i) = \sum_{i=1}^r (a_i + b_i) \cdot y_i] \right)$$
$$\leq \frac{\#A}{\#\mathbb{Z}_d^r/H} + (1 - \varepsilon')(1 - \frac{\#A}{\#\mathbb{Z}_d^r/H}).$$

From this, we deduce that $\varepsilon' - \varepsilon \leq \varepsilon' \cdot \frac{\#A}{\#\mathbb{Z}_d^r/H}$ and thus $A \neq \emptyset$.
Let $(a_1 \dots a_r)$ be in A. We have

$$\mathop{\mathrm{E}}_{x \in G} \left(\Pr_{b \in H} [f(d \cdot x + \sum_{i=1}^r a_i \cdot x_i) - \sum_{i=1}^r a_i \cdot x_i = \sum_{i=1}^r b_i \cdot y_i] \right) \geq 1 - \varepsilon'.$$

Hence, there exists a $x \in G$ such that $\Pr_{b \in H}[cste = \sum_{i=1} b_i \cdot y_i] \geq 1 - \varepsilon' > \frac{1}{2}$. Therefore, for all $b \in H$ there holds $\sum_{i=1}^r b_i \cdot y_i = 0$. Finally, we can define φ such that $\varphi(d \cdot x + \sum_{i=1}^r a_i \cdot x_i) = \sum_{i=1}^r a_i \cdot y_i$. □

Lemma 14. *Assume we are able to compute f s. t. $\Pr_{x \in G}(f(x) \neq \varphi(x)) \leq \varepsilon$. Then we can compute a function g such that $\Pr_{x \in G}(g(x) \neq \varphi(x)) \leq 12\varepsilon^2$ with at most 6 calls to f.*

Proof. For an $x \in G$, we compute the function g at x as follows:

1. Pick $y_1, y_2, y_3 \in G$.
2. Compute $f(x + y_i)$, $f(y_i)$ for $i = 1, 2, 3$.
3. If $f(x + y_1) - f(y_1) = f(x + y_2) - f(y_2)$, let this be $g(x)$. Otherwise, we set that $g(x) = f(x + y_3) - f(y_3)$.

Set $P_x := \Pr_{y \in G}(f(y) \neq \varphi(y) \text{ or } f(x + y) \neq \varphi(x + y))$. By definition, we have $P_x \leq 2\varepsilon$. We obtain $\Pr(g(x) \neq \varphi(x)) \leq 2P_x^2(1 - P_x) + P_x^2 \leq 12\varepsilon^2$. □

Proof (Theorem 8). By iterating n times, we get $\Pr(f(x) \neq \varphi(x)) \leq \frac{1}{12} \cdot (12\varepsilon)^{2^n} \leq \frac{1}{12} \cdot \xi^{2^n}$. For $n > \log_2(\frac{\log(\#G)}{\log(1/\xi)})$ we have $\Pr_{x \in G}(f(x) \neq \varphi(x)) < \frac{1}{\#G}$. Hence, this probability is equal to zero and the complexity is multiplied by a factor that is in the class $\text{poly}(\log(\#G))$. □

Efficient Extension of Standard Schnorr/RSA Signatures into Universal Designated-Verifier Signatures

Ron Steinfeld, Huaxiong Wang, and Josef Pieprzyk

Dept. of Computing, Macquarie University, North Ryde, Australia
{rons,hwang,josef}@comp .mq.edu.au
http://www.ics.mq.edu.au/acac/

Abstract. Universal Designated-Verifier Signature (UDVS) schemes are digital signature schemes with additional functionality which allows *any* holder of a signature to *designate* the signature to any desired *designated-verifier* such that the designated-verifier can verify that the message was signed by the signer, but is unable to convince anyone else of this fact. Since UDVS schemes reduce to standard signatures when no verifier designation is performed, it is natural to ask how to extend the classical Schnorr or RSA signature schemes into UDVS schemes, so that the existing key generation and signing implementation infrastructure for these schemes can be used without modification. We show how this can be efficiently achieved, and provide proofs of security for our schemes in the random oracle model.

1 Introduction

Universal Designated-Verifier Signature (UDVS) schemes introduced by Steinfeld et al [16] are digital signature schemes with additional functionality which allows *any* holder of a signature to *designate* the signature to any desired *designated-verifier* such that the designated-verifier can verify that the message was signed by the signer, but is unable to convince anyone else of this fact, because the verifier's secret key allows him to forge the designated-verifier signatures without the signer's cooperation. Such signature schemes protect the privacy of signature holders from dissemination of signatures by verifiers, and have applications in certification systems [16].

The previous work [16] has shown how to construct efficient deterministic UDVS schemes from Bilinear group-pairs. However, since UDVS schemes reduce to standard signatures when no verifier designation is performed, it is natural to ask how to extend the classical Schnorr [14] or RSA [12] signature schemes into UDVS schemes, so that the existing key generation and signing implementation infrastructure for these schemes can be used without modification — the UDVS functionality can be added to such implementations as an optional feature. In this paper we show how this can be efficiently achieved, and provide concrete proofs of security for our schemes in the random oracle model [2].

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 86–100, 2004.

As shown in [16], any secure efficient construction of an unconditionally-private UDVS scheme with *unique signatures* (e.g. fully deterministic UDVS schemes with unique secret keys) gives rise to a secure efficient ID-Based Encryption (IBE) scheme. Constructing secure and efficient IBE schemes from classical Diffie-Hellman or RSA problems is a long-standing open problem [3], and until this problem is solved we also cannot hope to construct unconditionally-private UDVS schemes with unique signatures based on classical problems. However, the results in this paper show that by giving up the unique signature requirement and allowing randomization in either the signing (in the case of Schnorr signatures) or designation (in the case of RSA) algorithms, one can construct efficient UDVS schemes from classical problems. Although the UDVS schemes presented in this paper do not have unique signatures, they still achieve perfect unconditional privacy in the sense of [16].

Due to space limitation, the proofs of all theorems in the paper are omitted. They are included in the full version of this paper [17].

1.1 Related Work

As pointed out in [16], the concept of UDVS schemes can be viewed as an application of the general idea of *designated-verifier proofs,* introduced by Jakobsson, Sako and Impagliazzo [8], where a prover non-interactively designates a proof of a statement to a verifier, in such a way that the verifier can simulate the proof by himself with his secret key and thus cannot transfer the proof to convince anyone else about the truth of the statement, yet the verifier himself is convinced by the proof. The distinctive feature of UDVS schemes is *universal* designation: *anyone* who obtains a signature can designate it.

Two of our proposed UDVS schemes (namely $\mathsf{SchUDVS}_2$ and RSAUDVS) make use of the paradigm in [8] of using a trapdoor commitment in a non-interactive proof of knowledge to achieve verifier designation. Since the underlying construction techniques used in these schemes is known, we view our main contribution here is in providing a concrete security analysis which bounds the insecurity of these schemes in terms of the underlying primitives. Our third proposed scheme $\mathsf{SchUDVS}_1$ shows an alternative and more efficient approach than the paradigm of [8], for extending the Schnorr signature scheme into a UDVS scheme, using the Diffie-Hellman function. It is an analogoue of the the bilinear-based approach for constructing UDVS schemes proposed in [16].

Besides providing UDVS schemes based on classical problems, another contribution of this paper is in defining a stronger unforgeability notion for UDVS schemes, which allows the forger access to the attacked designated verifier's verification oracle, as well as to the signer's signing oracle (whereas the model in [16] only allows access to the signing oracle). We analyse our schemes in this stronger model.

Further related work to UDVS schemes is discussed in [16].

2 Preliminaries

2.1 Algorithms and Probability Notation

We say that a function $f : \mathbb{N} \to \mathbb{R}$ is a *negligible* function if, for any $c > 0$, there exists $k_0 \in \mathbb{N}$ such that $f(k) < 1/k^c$ for all $k > k_0$. We say that a probability function $p : \mathbb{N} \to \mathbb{R}$ is *overwhelming* if the function $q : \mathbb{N} \to \mathbb{R}$ defined by $q(k) = 1 - p(k)$ is a negligible function. For various algorithms discussed, we will define a sequence of integers to measure the *resources* of these algorithms (e.g. running-time plus program length, number of oracle queries to various oracles). All these resource parameters can in general be functions of a *security parameter* k of the scheme. We say that an algorithm A with resource parameters $RP = (r_1, \ldots, r_n)$ is *efficient* if each resource parameter $r_i(k)$ of A is bounded by a polynomial function of the security parameter k, i.e. there exists a $k_0 > 0$ and $c > 0$ such that $r_i(k) < k^c$ for all $k > k_0$.

2.2 Discrete-Log and Diffie-Hellman Problems

Our schemes use the following known hard problems for their security. For all these problems GC denotes an algorithm that on input a security parameter k, returns an instance (D_G, g) of a multiplicative group G of prime order q with generator g (the description string D_G determines the group and contains the group order q).

1 Discrete-Log Problem (DL) [4]: Given $(D_G, g) = \mathsf{GC}(k)$ and $y_1 = g^{x_1}$ for uniformly random $x_1 \in \mathbb{Z}_q^*$, compute x_1. We say that DL is *hard* if the success probability $\mathbf{Succ}_{\mathsf{A},\mathsf{DL}}(k)$ of any efficient DL algorithm A with run-time $t(k)$ is upper-bounded by a negligible function $\mathbf{InSec}_{\mathsf{DL}}(t)$ of k.

2 Computational Diffie-Hellman Problem (CDH) [4]: Given $(D_G, g) = \mathsf{GC}(k)$, $y_1 = g^{x_1}$ and $y_2 = g^{x_2}$ for uniformly random $x_1, x_2 \in \mathbb{Z}_q^*$, compute $\mathsf{CDH}_g(g^{x_1}, g^{x_2}) \stackrel{\text{def}}{=} g^{x_1 x_2}$. We say that CDH is *hard* if the success probability $\mathbf{Succ}_{\mathsf{A},\mathsf{CDH}}(k)$ of any efficient CDH algorithm A with run-time $t(k)$ is upper-bounded by a negligible function $\mathbf{InSec}_{\mathsf{CDH}}(t)$ in k.

3 Strong Diffie-Hellman Problem (SDH) [1, 10]: Given $(D_G, g) = \mathsf{GC}(k)$, $y_1 = g^{x_1}$ and $y_2 = g^{x_2}$ for uniformly random $x_1, x_2 \in \mathbb{Z}_q^*$, compute $g^{x_1 x_2}$ given access to a restricted Decision Diffie-Hellman (DDH) oracle $\mathsf{DDH}_{x_1}(.,.)$, which on input $(w, K) \in G \times G$, returns 1 if $K = w^{x_1}$ and 0 else. We say that SDH is *hard* if the success probability $\mathbf{Succ}_{\mathsf{A},\mathsf{SDH}}(k)$ of any efficient SDH algorithm A with run-time $t(k)$ and which makes up to $q(k)$ queries to $\mathsf{DDH}_{x_1}(.,.)$, is upper-bounded by a negligible function $\mathbf{InSec}_{\mathsf{SDH}}(t, q)$ in k.

We remark that the Strong Diffie-Hellman problem (SDH) as defined above and in [1] is a potentially harder variant of the Gap Diffie-Hellman (GDH) problem as defined in [10]. The difference between the two problems is in the DDH oracle: In the GDH problem the DDH oracle accepts *four* inputs (h, z_1, z_2, K)

from the attacker and decides whether $K = \mathsf{CDH}_h(z_1, z_2)$, whereas in the SDH problem the attacker can only control the (z_2, K) inputs to the DDH oracle and the other two are fixed to the values $h = g$ and $z_1 = y_1$ (we call this weaker oracle a *restricted* DDH oracle).

2.3 Trapdoor Hash Functions

Some of our proposed UDVS schemes make use of a general cryptographic scheme called a *trapdoor hash function.* We recall the definition and security notions for such schemes [15]. A trapdoor hash function scheme consists of three efficient algorithms: a *key generation* algorithm GKF, a *hash function evaluation* algorithm *F,* and a *collision solver* algorithm CSF. On input a security parameter k, the (randomized) key-gen. algorithm $\mathsf{GKF}(k)$ outputs a secret/public-key pair (sk, pk). On input a public-key *pk,* message $m \in M$ and random $r \in R$ (Here M and R are the message and randomness spaces, respectively), the hash function evaluation algorithm outputs a hash string $h = F_{pk}(m; r) \in H$ (here H is the hash string space). On input a key-pair (sk, pk), a message/randomizer pair $(m_1, r_1) \in M \times R$ and a second message $m_2 \in M$, the collision solver algorithm outputs a second randomizer $r_2 = \mathsf{CSF}((sk, pk), (m_1, r_1), m_2) \in R$ such that (m_1, r_1) and (m_2, r_2) constitute a collision for F_{pk}, i.e. $F_{pk}(m_1; r_1) = F_{pk}(m_2; r_2)$.

There are two desirable security properties for a trapdoor hash function scheme TH = (GKF, F, CSF). The scheme TH is called *collision-resistant* if the success probability $\mathbf{Succ}^{\mathrm{CR}}_{A,\mathsf{TH}}$ of any efficient attacker A in the following game is negligible. A key-pair $(sk, pk) = \mathsf{GKF}(k)$ is generated, and A is given k and the public-key pk. A can run for time t and succeeds if it outputs a collision (m_1, r_1) and (m_2, r_2) for F_{pk} satisfying $F_{pk}(m_1, r_1) = F_{pk}(m_2, r_2)$ and $m_1 \neq m_2$. We denote by $\mathbf{InSec}^{\mathrm{CR}}_{\mathsf{TH}}(t)$ the maximal success probability in above game over all attackers A with run-time plus program length at most t. The scheme TH is called *perfectly-trapdoor* if it has the following property: for each key-pair $(sk, pk) = \mathsf{GKF}(k)$ and message pair $(m_1, m_2) \in M \times M$, if r_1 is chosen uniformly at random from R, then $r_2 \stackrel{\mathrm{def}}{=} \mathsf{CSF}((sk, pk), (m_1, r_1), m_2) \in R$ has a uniform probability distribution on R.

3 Universal Designated-Verifier Signature (UDVS) Schemes

We review the definition of UDVS schemes and their security notions [16]. For unforgeability we also introduce a stronger notion of security than used in [16].

A Universal Designated Verifier Signature (UDVS) scheme DVS consists of seven algorithms and a 'Verifier Key-Registration Protocol' $\mathsf{P_{KR}}$. All these algorithms may be randomized.

1. **Common Parameter Generation** GC — on input a security parameter k, outputs a string consisting of common scheme parameters *cp* (publicly shared by all users).

2. **Signer Key Generation** GKS — on input a common parameter string *cp,* outputs a secret/public key-pair (sk_1, pk_1) for *signer.*
3. **Verifier Key Generation** GKV — on input a common parameter string *cp,* outputs a secret/public key-pair (sk_3, pk_3) for *verifier.*
4. **Signing** S — on input signing secret key sk_1, message m, outputs *signer*'s publicly-verifiable (PV) signature σ.
5. **Public Verification** V — on input *signer*'s public key pk_1 and message/PV-signature pair (m, σ), outputs verification decision $d \in \{Acc, Rej\}$.
6. **Designation** CDV — on input a *signer*'s public key pk_1, a *verifier*'s public key pk_3 and a message/PV-signature pair (m, σ), outputs a designated-verifier (DV) signature $\widehat{\sigma}$.
7. **Designated Verification** VDV — on input a *signer*'s public key pk_1, *verifier*'s secret key sk_3, and message/DV-signature pair $(m, \widehat{\sigma})$, outputs verification decision $d \in \{Acc, Rej\}$.
8. **Verifier Key-Registration** $\mathsf{P_{KR}} = (\mathsf{KRA}, \mathsf{VER})$ — a protocol between a 'Key Registration Authority' (KRA) and a 'Verifier' (VER) who wishes to register a verifier's public key. On common input *cp,* the algorithms KRA and VER interact by sending messages alternately from one to another. At the end of the protocol, KRA outputs a pair $(pk_3, Auth)$, where pk_3 is a verifier's public-key, and $Auth \in \{Acc, Rej\}$ is a key-registration authorization decision. We write $P_{KR}(\mathsf{KRA}, \mathsf{VER}) = (pk_3, Auth)$ to denote this protocol's output.

Verifier Key-Reg. Protocol. The purpose of the 'Verifier Key-Registration' protocol is to force the verifier to 'know' the secret-key corresponding to his public-key, in order to enforce the non-transferability privacy property. In this paper we assume, following [16], the *direct* key reg. protocol, in which the verifier simply reveals his secret / public key to the KRA, who authorizes the public-key only if the provided secret-key matches the public key.

3.1 Unforgeability

In the case of a UDVS scheme there are actually two types of unforgeability properties to consider. The first property, called called 'PV-Unforgeability', is just the usual existential unforgeability notion under chosen-message attack [6] for the standard PV signature scheme $D = (\mathsf{GC}, \mathsf{GKS}, \mathsf{S}, \mathsf{V})$ induced by the UDVS scheme (this prevents attacks to fool the *designator*). The second property, called 'DV-Unforgeability', requires that it is difficult for an attacker to forge a DV-signature $\widehat{\sigma}^*$ by the signer on a 'new' message m^*, such that the pair $(m^*, \widehat{\sigma}^*)$ passes the DV-verification test with respect to a given designated-verifier's public key pk_3 (this prevents attacks to fool the designated verifier, possibly mounted by a dishonest designator). As pointed out in [16], it is sufficient to prove the DV unforgeability of a UDVS scheme, since the 'DV-unforgeability' property implies the 'PV-unforgeability' property.

In this paper we introduce a stronger version of DV-unforgeability than used in [16], which we call ST-DV-UF. This model allows the forger also access to

the verification oracle of the designated-verifier (this oracle may help the forger because it uses the designated-verifier's secret key, which in turn can be used to forge DV signatures, as required by the privacy property). Note that the model in [16] does not provide this oracle. We believe it is desirable for UDVS schemes to be secure even under such attacks, and place no restrictions on the attacker in accessing the verifier's oracle — in particular the attacker can control both the message/DV sig. pair as well as the signer's public key in accessing this oracle. We remark (proof omitted) that the *strong* DV-unforgeability of the UDVS scheme in [16] follows (in the random-oracle model) from the hardness of a *gap* version of the Bilinear Diffie-Hellman (BDH) problem, in which the attacker has access to a BDH decision oracle (whereas just hardness of BDH suffices for this scheme to achieve the weaker DV-unforgeability notion in [16]).

Definition 1 (Strong DV-Unforgeability). *Let* $\mathsf{DVS} = (\mathsf{GC}, \mathsf{GKS}, \mathsf{GKV}, \mathsf{S}, \mathsf{V}, \mathsf{CDV}, \mathsf{VDV}, \mathsf{P_{KR}})$ *be a UDVS scheme. Let* A *denote a forger attacking the unforgeability of* DVS. *The Strong DV-Unforgeability notion ST-UF-DV for this scheme is defined as follows:*

1. ***Attacker Input:*** *Signer and Verifier's public-keys* (pk_1, pk_3) *(where* $(sk_1, pk_1) = \mathsf{GKS}(cp)$, $(sk_3, pk_3) = \mathsf{GKV}(cp)$ ***and*** $cp = \mathsf{GC}(k)$*).*
2. ***Attacker Resources*:** *Run-time plus program-length at most* t, *Oracle access to signer's signing oracle* $\mathsf{S}(sk_1, .)$ *(*q_s *queries), oracle access to designated-verifier's verification oracle* $\mathsf{VDV}(., sk_3, ., .)$ *(*q_v *queries) and, if scheme* DVS *makes use of* n *random oracles* $RO_1, \ldots, RO_n$, *allow* q_{RO_i} *queries to the* i*th oracle* RO_i *for* $i = 1, \ldots, n$. *We write attacker's Resource Parameters (RPs) as* $RP = (t, q_s, q_v, q_{RO_1}, \ldots, q_{RO_n})$.
3. ***Attacker Goal:*** *Output a forgery message/DV-signature pair* $(m^*, \widehat{\sigma}^*)$ *such that:*
   (1) The forgery is valid, i. e. $\mathsf{VDV}(pk_1, sk_3, m^*, \widehat{\sigma}^*) = Acc$.
   (2) Message m^* *is 'new', i.e. has not been queried by attacker to* S.
4. ***Security Notion Definition*:** *Scheme is said to be* unforgeable *in the sense of ST-UF-DV if, for any efficient attacker* A, *the probability* $\mathbf{Succ}^{\mathrm{ST-UF-DV}}_{\mathsf{A,DVS}}(k)$ *that* A *succeeds in achieving above goal is a negligible function of* k. *We quantify the insecurity of* DVS *in the sense of ST-UF-DV against arbitrary attackers with resource parameters* $RP = (t, q_s, q_v, q_{RO_1}, \ldots, q_{RO_n})$ *by the probability*

$$\mathbf{InSec}^{\mathrm{ST-UF-DV}}_{\mathsf{DVS}}(t, q_s, q_v, q_{RO_1}, \ldots, q_{RO_n}) \stackrel{\mathrm{def}}{=} \max_{\mathsf{A} \in AS_{RP}} \mathbf{Succ}^{\mathrm{ST-UF-DV}}_{\mathsf{A,DVS}}(k),$$

where the set AS_{RP} *contains all attackers with resource parameters RP.*

3.2 Non-transferability Privacy

Informally, the purpose of the privacy property for a UDVS scheme is to prevent a designated-verifier from using the DV signature σ_{dv} on a message m to produce evidence which convinces a third-party that the message m was signed by the

signer. The privacy is achieved because the designated-verifier can forge DV signatures using his secret-key, so even if the designated-verifier reveals his secret key to the third-party, the third-party cannot distinguish whether a DV signature was produced by the designator or forged by the designated-verifier.

We review the privacy model from [16]. The attacker is modelled as a pair of interacting algorithms $(\mathsf{A}_1, \mathsf{A}_2)$ representing the designated-verifier (DV) and Third-Party (TP), respectively. Let $\widehat{\mathsf{A}_1}$ denote a forgery strategy. The goal of A_2 is to distinguish whether it is interacting with A_1 who has access to designated signatures (game yes) or with $\widehat{\mathsf{A}_1}$, who doesn't have access to designated signatures (game no). More precisely, the game yes runs in two stages as follows.

Stage 1. $(\mathsf{A}_1, \mathsf{A}_2)$ are run on input pk_1, where $(sk_1, pk_1) = \mathsf{GKS}(cp)$ and $cp = \mathsf{GC}(k)$. In this stage, A_1 has access to: (1) signing oracle $\mathsf{S}(sk_1, .)$, (2) KRA key-reg. oracle to register verifier public keys pk via P_{KR} interactions, (3) A_2 oracle for querying a message to A_2 and receiving a response. At end of stage 1, A_1 outputs a message m^* not queried to S during the game (m^* is given to A_2). Let $\sigma^* = \mathsf{S}(sk_1, m^*)$.

Stage 2. A_1 continues to make S,KRA and A_2 queries as in stage 1, but also has access to a designation oracle $\mathsf{CDV}(pk_1, ., m^*, \sigma^*)$ which it can query with any verifier public-key pk which was answered *Acc* by a previous KRA key-reg. query. At end of stage 2, A_2 outputs a decision $d \in \{\mathsf{yes}, \mathsf{no}\}$.

The game no is defined in the same way except that (1) A_1 is replaced by $\widehat{\mathsf{A}_1}$, (2) $\widehat{\mathsf{A}_1}$ receives as input pk_1 and the program for A_1, (3) $\widehat{\mathsf{A}_1}$ cannot make any designation queries, (4) $\widehat{\mathsf{A}_1}$ makes same number of sign queries as A_1 (possibly 0).

Let P_{yes} and P_{no} denote the probability that A_2 outputs yes in games yes and no, respectively. We let $C_{\widehat{\mathsf{A}_1}}(\mathsf{A}_1, \mathsf{A}_2) \stackrel{\text{def}}{=} |P_{\mathsf{yes}} - P_{\mathsf{no}}|$ denote A_2's distinguishing advantage.

Definition 2. *A UDVS scheme is said to achieve* complete and perfect unconditional privacy *(PR notion) if there exists an efficient forgery strategy* $\widehat{\mathsf{A}_1}$ *such that* $C_{\widehat{\mathsf{A}_1}}(\mathsf{A}_1, \mathsf{A}_2) = 0$ *for any efficient* A_1 *and computationally unbounded* A_2.

4 Two Extensions of Schnorr Signature Scheme into UDVS Schemes

We will present two UDVS schemes which are both extensions of the Schnorr [14] signature scheme (that is, the signer key-generation, signing and public-verification algorithms in both schemes are identical to those of the Schnorr signature). The first UDVS scheme $\mathsf{SchUDVS}_1$ has an efficient and deterministic designation algorithm and its unforgeability relies on the Strong Diffie-Hellman (SDH) assumption. The second UDVS scheme $\mathsf{SchUDVS}_2$ has a less efficient randomized designation algorithm, but its unforgeability follows from the weaker Discrete-Logarithm (DL) assumption (in the random-oracle model).

4.1 First Scheme: $\mathsf{SchUDVS}_1$

Our first UDVS scheme $\mathsf{SchUDVS}_1$ is defined as follows. Let $\{0,1\}^{\leq \ell}$ denote the message space of all bit strings of length at most ℓ bits. The scheme makes use of a cryptographic hash function $H : \{0,1\}^{\leq \ell} \times \{0,1\}^{l_G} \rightarrow \{0,1\}^{l_H}$, modelled as a random-oracle [2] in our security analysis. We assume that elements of the group G output by algorithm GC are represented by bit strings of length $l_G \geq l_q$ bits, where $l_q \stackrel{\text{def}}{=} \lfloor \log_2 q \rfloor + 1$ is the bit length of q.

1. **Common Parameter Generation** GC. (Identical to Schnorr). Choose a group G of prime order $q > 2^{l_H}$ with description string D_G (e.g. if G is a subgroup of $\mathbb{Z}_p^*$, the string D_G would contain (p,q)), and let $g \in G$ denote a generator for G. The common parameters are $cp = (D_G, g)$.
2. **Signer Key Generation** GKS. (Identical to Schnorr). Given the common parameters cp, pick random $x_1 \in \mathbb{Z}_q^*$ and compute $y_1 = g^{x_1}$. The public key is $pk_1 = (cp, y_1)$. The secret key is $sk_1 = (cp, x_1)$.
3. **Verifier Key Generation** GKV. Given the common parameters cp, pick random $x_3 \in \mathbb{Z}_q^*$ and compute $y_3 = g^{x_3}$. The public key is $pk_3 = (cp, y_3)$. The secret key is $sk_3 = (cp, x_3)$.
4. **Signing** S. (Identical to Schnorr). Given the signer's secret key (cp, x_1), and message m, choose a random $k \in \mathbb{Z}_q$ and compute $u = g^k$, $r = H(m,u)$ and $s = k + r \cdot x_1 \pmod q$. The PV signature is $\sigma = (r,s)$.
5. **Public Verification** V. (Identical to Schnorr). Given the signer's public key y_1 and a message/PV sig. pair $(m,(r,s))$, accept if and only if $H(m,u) = r$, where $u = g^s \cdot y_1^{-r}$.
6. **Designation** CDV. Given the signer's public key y_1, a verifier's public key y_3 and a message/PV-signature pair $(m,(r,s))$, compute $u = g^s \cdot y_1^{-r}$ and $K = y_3^s$. The DV signature is $\widehat{\sigma} = (u,K)$.
7. **Designated Verification** VDV. Given a signer's public key y_1, a verifier's secret key x_3, and message/DV-sig. pair $(m,(u,K))$, accept if and only if $K = (u \cdot y_1^r)^{x_3}$, where $r = H(m,u)$.

Unforgeability. The PV-Unforgeability of $\mathsf{SchUDVS}_1$ is equivalent to the unforgeability of the Schnorr signature, which in turn is equivalent to the Discrete-Logarithm (DL) assumption in G, assuming the random-oracle model for $H(.)$[11]. However, for the DV-Unforgeability of $\mathsf{SchUDVS}_1$, it is clear that the stronger 'Computational Diffie-Hellman' (CDH) assumption in G is certainly *necessary* — an attacker can forge a DV signature (u,K) on a message m by choosing a random $u \in G$, computing $r = H(m,u)$ and then $K = \mathsf{CDH}_g(u \cdot y_1^r, y_3)$ (indeed this is the idea behind the proof of the privacy of $\mathsf{SchUDVS}_1$ — see below). Moreover, in the strong DV-unforgeability attack setting, the even stronger 'Strong Diffie-Hellman' (SDH) assumption in G is necessary. This is because the forger's access to the verifier's VDV oracle allows him to simulate the fixed-input DDH oracle $\mathsf{DDH}_{x_3}(w,K)$ which decides whether $K = w^{x_3}$ or not (see Sec. 2.2), namely we have $\mathsf{DDH}_{x_3}(w,K) = \mathsf{VDV}(y_1', x_3, m, (u,K))$ with $y_1' = (w \cdot u^{-1})^{r^{-1} \bmod q}$ and $r = H(m,u)$. Note that this does not rule out the possibility that there may

be another attack which even bypasses the need to break SDH. Fortunately, the following theorem shows that this is not the case and SDH is also a *sufficient* condition for Strong DV-Unforgeability of $\mathsf{SchUDVS}_1$, assuming the random-oracle model for $H(.)$. The proof uses the forking technique, as used in the proof in [11] of PV-Unforgeability of the Schnorr signature.

Theorem 1 (Strong DV-Unforg. of $\mathsf{SchUDVS}_1$). *If the Strong Diffie-Hellman problem (SDH) is hard in groups generated by the common-parameter algorithm GC, then the scheme $\mathsf{SchUDVS}_1$ achieves Strong DV-unforgeability (ST-UF-DV notion) in the random-oracle model for $H(.)$. Concretely, the following insecurity bound holds:*

$$\mathbf{InSec}^{\mathrm{ST-UF-DV}}_{\mathsf{SchUDVS}_1}(t, q_s, q_v, q_H) \leq 2\left[(q_H + q_v)\mathbf{InSec}_{\mathsf{SDH}}(t[S], q[S])\right]^{1/2} + \frac{q_s(q_H + q_s + q_v) + 2(q_H + q_v) + 1}{2^{l_H}},$$

where $t[S] = 2t + 2(q_H + q_s + q_v + 1)(T_S + O(l_H)) + (q_s + 1)O(l_q T_g) + O(l_q^2)$, where $T_S = O(\log_2(q_H + q_s + q_v) \cdot (\ell + l_G))$ and $q[S] = 2q_v$. Here we denote by T_g the time needed to perform a group operation in G.

Privacy. The privacy of $\mathsf{SchUDVS}_1$ follows from the existence of an algorithm for forging DV signatures (with identical probability distribution as that of real DV signatures) using the verifier's secret key, which is a trapdoor for solving the CDH problem on which the DV-Unforgeability relies.

Theorem 2 (Privacy of $\mathsf{SchUDVS}_1$). *The scheme $\mathsf{SchUDVS}_1$ achieves complete and perfect unconditional privacy (PR notion).*

4.2 Second Scheme: $\mathsf{SchUDVS}_2$

Our second UDVS scheme $\mathsf{SchUDVS}_2$ trades off efficiency for a better provable unforgeability security guarantee. Rather than using the Diffie-Hellman trapdoor function to achieve privacy, we instead get the designator to produce a Schnorr proof of knowledge of the PV signature (r, s). This proof of knowledge is made non-interactive in the random-oracle model using the Fiat-Shamir heuristic [5], but using a *trapdoor hash function* [9,15] $F_{y_3}(.;.)$ composed with a random oracle $J(.)$ in producing the 'verifier random challenge' $\widehat{r}$ for this proof of knowledge. The designated-verifier's secret key consists of the trapdoor for the hash function F_{y_3}, which suffices for forging the DV signatures, thus providing the privacy property. We remark that a similar technique was used by Jakobsson Sako and Impagliazzo [8], who used a trapdoor commitment scheme in constructing a designated-verifier undeniable signature scheme. Our scheme can use *any* secure trapdoor hash function.

The resulting scheme is defined as follows. Let $\{0,1\}^{\leq \ell}$ denote the message space of all bit strings of length at most ℓ bits. The scheme makes use of two cryptographic hash functions $H : \{0,1\}^{\leq \ell} \times \{0,1\}^{l_G} \rightarrow \{0,1\}^{l_H}$ and

$J : \{0,1\}^{\leq \ell} \times \mathbb{Z}_{2^{l_H}} \times \{0,1\}^{l_G} \times \{0,1\}^{l_F} \rightarrow \{0,1\}^{l_J}$, both modelled as random-oracles [2] in our security analysis. We also use a trapdoor hash function scheme TH = (GKF, F, CSF) with $F_{y_3} : \{0,1\}^{l_G} \times R_F \rightarrow \{0,1\}^{l_F}$ (we refer the reader to Section 2 for a definition of trapdoor hash function schemes). We assume that elements of the group G output by algorithm GC are represented by bit strings of length $l_G \geq l_q$ bits, where $l_q \stackrel{\text{def}}{=} \lfloor \log_2 q \rfloor + 1$ is the bit length of q.

1. **Common Parameter Generation** GC. (Identical to Schnorr). Choose a group G of prime order q with description string D_G (e.g. if G is a subgroup of $\mathbb{Z}_p^*$, the string D_G would contain (p, q)), and let $g \in G$ denote a generator for G. The common parameters are $cp = (k, D_G, g)$ (k is the security parameter).
2. **Signer Key Generation** GKS. (Identical to Schnorr). Given the common parameters cp, pick random $x_1 \in \mathbb{Z}_q$ and compute $y_1 = g^{x_1}$. The public key is $pk_1 = (cp, y_1)$. The secret key is $sk_1 = (cp, x_1)$.
3. **Verifier Key Generation** GKV. Given the common parameters $cp = k$, run TH's key-gen. algorithm to compute $(sk, pk) = \mathsf{GKF}(k)$. The public key is $pk_3 = (cp, pk)$. The secret key is $sk_3 = (cp, sk, pk)$.
4. **Signing** S. (Identical to Schnorr). Given the signer's secret key (cp, x_1), and message m, choose a random $k \in \mathbb{Z}_q$ and compute $u = g^k$, $r = H(m, u)$ and $s = k + r \cdot x_1 \pmod q$. The PV signature is $\sigma = (r, s)$.
5. **Public Verification** V. (Identical to Schnorr). Given the signer's public key y_1 and a message/PV sig. pair $(m, (r, s))$, accept if and only if $H(m, u) = r$, where $u = g^s \cdot y_1^{-r}$.
6. **Designation** CDV. Given the signer's public key y_1, a verifier's public key $pk_3 = (cp, pk)$ and a message/PV-signature pair $(m, (r, s))$, compute $u = g^s \cdot y_1^{-r}$, $\widehat{u} = g^{\widehat{k}}$ for a random $\widehat{k} \in \mathbb{Z}_q$, $\widehat{h} = F_{pk}(\widehat{u}; \widehat{r}_F)$ for a random $\widehat{r}_F \in R_F$, $\widehat{r} = J(m, r, u, \widehat{h})$ and $\widehat{s} = \widehat{k} + \widehat{r} \cdot s \bmod q$. The DV signature is $\widehat{\sigma} = (u, \widehat{r}_F, \widehat{r}, \widehat{s})$.
7. **Designated Verification** VDV. Given a signer's public key y_1, a verifier's secret key $sk_3 = (cp, sk, pk)$, and message/DV-sig. pair $(m, (u, \widehat{r}_F, \widehat{r}, \widehat{s}))$, accept if and only if $J(m, r, u, \widehat{h}) = \widehat{r}$, where $r = H(m, u)$, $\widehat{h} = F_{pk}(\widehat{u}; \widehat{r}_F)$ and $\widehat{u} = g^{\widehat{s}} \cdot (u \cdot y_1^r)^{-\widehat{r}}$.

Unforgeability. The idea behind the DV-Unforgeability of $\mathsf{SchUDVS}_2$, is that the DV signature is effectively a proof of knowledge of the s portion of the PV Schnorr signature (r, s) by the signer on m. Namely, using the forking technique we can use a forger for $\mathsf{SchUDVS}_2$ to extract s and hence forge a Schnorr PV signature for some unsigned message m, or alternately to break the collision-resistance of the trapdoor hash scheme TH. We have the following concrete result. Note that we need only assume that $J(.)$ is a random-oracle in proving this result, but we provide a count of $H(.)$ queries to allow the use of our reduction bound in conjunction with known results on the unforgeability of the Schnorr signature which assume the random-oracle model for $H(.)$.

Theorem 3 (Strong DV-Unforg. of $\mathsf{SchUDVS}_2$). *If $\mathsf{SchUDVS}_2$ is PV-unforgeable (UF-PV notion) and* TH *is collision-resistant (CR notion) then*

$\mathsf{SchUDVS_2}$ *achieves Strong DV-unforgeability (ST-UF-DV notion) in the random-oracle model for J(.). Concretely, the following insecurity bound holds:*

$$\mathbf{InSec}^{\mathrm{ST-UF-DV}}_{\mathsf{SchUDVS_2}}(t, q_s, q_v, q_J, q_H) \leq$$
$$2[(q_J + q_v)q_s]^{1/2}\left[\mathbf{InSec}^{\mathrm{UF-PV}}_{\mathsf{SchUDVS_2}}(t[S], q_s[S], q_H[S]) + \mathbf{InSec}^{\mathrm{CR}}_{\mathsf{TH}}(t[T])\right]^{1/2}$$
$$+ \frac{2(q_J + q_v)q_s + 1}{2^{l_J}},$$

where $t[S] = t[T] = 2t + O((q_J + q_v)(\ell + l_F + l_G) + l_q T_g + l_q^2)$, $q_s[S] = 2q_s$ *and* $q_H[S] = 2q_H$. *Here we denote by* T_g *the time needed to perform a group operation in G.*

Privacy. The privacy of $\mathsf{SchUDVS_2}$ follows from the existence of an algorithm for forging DV signatures (with identical probability distribution as that of real DV signatures) using the verifier's secret key, which is a trapdoor for solving collisions in TH. In particular we need here the *perfectly-trapdoor* property of TH. This result holds in the standard model (no random-oracle assumptions).

Theorem 4 (Privacy of $\mathsf{SchUDVS_2}$). *If the scheme* TH *is perfectly-trapdoor then* $\mathsf{SchUDVS_2}$ *achieves complete and perfect unconditional privacy (PR notion).*

5 RSA-based Scheme: RSAUDVS

The idea for the construction of an RSA-based UDVS scheme is analogous to the second Schnorr-based scheme $\mathsf{SchUDVS_2}$, and is described as follows. The PV RSA signature known to the designator is the eth root $\sigma = h^{1/e} \bmod N$ of the message hash h, where (N, e) is the signer's RSA public key. To produce a DV signature on m, the designator computes a zero-knowledge proof of knowledge of the PV signature σ (made non-interactive using Fiat-Shamir method [5]), which is forgeable by the verifier. The Guilliou-Quisquater ID-based signature [7] is based on such a proof and is applied here for this purpose. To make the proof forgeable by the verifier, we use a trapdoor hash function in the computation of the challenge, as done in the $\mathsf{SchUDVS_2}$ scheme. We note that a restriction of the GQ proof that we use is that the random challenge r must be smaller than the public exponent e. To allow for small public exponents and achieve high security level, we apply α proofs in 'parallel', where α is chosen to achieve a sufficient security level — see security bound in our security analysis (a similar technique is used in the Fiat-Shamir signature scheme [5]).

The resulting scheme is defined as follows. Let $\{0,1\}^{\leq \ell}$ denote the message space of all bit strings of length at most ℓ bits. The scheme makes use of two cryptographic hash functions $H : \{0,1\}^{\leq \ell} \times R_S \rightarrow \{0,1\}^{l_H}$ and $J : \{0,1\}^{\leq \ell} \times \mathbb{Z}_{l_N}^{\alpha} \times \{0,1\}^{l_F} \rightarrow \mathbb{Z}_{2^{l_J/\alpha}}^{\alpha}$. Note that we only need to assume that $J(.)$ is a random-oracle model in our security analysis, and that we allow randomized RSA signatures with hash generation $h = H(m; s)$ for random s. The corresponding

verification is to check if $R(h, m) = Acc$ or not, where $R(.)$ is a binary relation function that outputs Acc if h is a valid hash of message m and outputs Rej else. Thus by a suitable choice of $H(.,.)$ and $R(.,.)$ our scheme can instantiated with any of the standardised variants of RSA signatures such as RSASSA-PSS or RSASSA-PKCS1-v15, as specified in the PKCS1 standard [13]. We also use a trapdoor hash function scheme TH = (GKF, F, CSF) with $F_{y_3} : \{0,1\}^{l_G} \times R_F \rightarrow \{0,1\}^{l_F}$ (we refer the reader to Section 2 for a definition of trapdoor hash function schemes). Here l_N denotes the length of RSA modulus N of the signer's public key.

1. **Common Parameter Generation** GC. (Identical to RSA). The comm. pars. are $cp = k$ (k is the security parameter).
2. **Signer Key Generation** GKS. (Identical to RSA). Given the common parameters cp, choose a prime $e > 2^{l_J/\alpha}$. Pick random primes p and q such that $N = pq$ has bit-length l_N and $\gcd(e, \phi(N)) = 1$, where $\phi(N) = (p-1)(q-1)$. Compute $d = e^{-1} \bmod \phi(N)$. The public key is $pk_1 = (cp, N, e)$. The secret key is $sk_1 = (cp, N, e, d)$.
3. **Verifier Key Generation** GKV. Given the comm. pars. $cp = k$, run TH's key-gen. algorithm to compute $(sk, pk) = \mathsf{GKF}(k)$. The public key is $pk_3 = (cp, pk)$. The secret key is $sk_3 = (cp, sk, pk)$.
4. **Signing** S. (Identical to RSA). Given the signer's secret key (cp, N, e, d), and message m, choose a random $s \in R_S$ and compute $h = H(m, s)$ and $\sigma = h^d \bmod N$. The PV signature is σ.
5. **Public Verification** V. (Identical to RSA). Given the signer's public key (cp, N, e) and a message/PV sig. pair (m, σ), accept if and only if $R(m, h) = Acc$, where $h = \sigma^e \bmod N$.
6. **Designation** CDV. Given the signer's public key (cp, N, e), a verifier's public key $pk_3 = (cp, pk)$ and a message/PV-signature pair (m, σ), choose α random elements $k_i \in \mathbb{Z}_N^*$ and compute $\widehat{u} = (\widehat{u}_1, \ldots, \widehat{u}_\alpha)$, where $\widehat{u}_i = k_i^e \bmod N$ for $i = 1, \ldots, \alpha$. Compute $\widehat{h} = F_{pk}(\widehat{u}; \widehat{r}_F)$ for random $\widehat{r}_F \in R_F$. Compute $\widehat{r} = (\widehat{r}_1, \ldots, \widehat{r}_\alpha) = J(m, h, \widehat{h})$, where $h = \sigma^e \bmod N$ and $\widehat{r}_i \in \mathbb{Z}_{2^{l_J/\alpha}}$ for $i = 1, \ldots, \alpha$. Compute $\widehat{s} = (\widehat{s}_1, \ldots, \widehat{s}_\alpha)$, where $\widehat{s}_i = k_i \cdot \sigma^{\widehat{r}_i} \bmod N$ for all $i = 1, \ldots, \alpha$. The DV signature is $\widehat{\sigma} = (h, \widehat{r}_F, \widehat{r}, \widehat{s})$.
7. **Designated Verification** VDV. Given a signer's public key (cp, N, e), a verifier's secret key $sk_3 = (cp, sk, pk)$, and message/DV-sig. pair $(m, (h, \widehat{r}_F, \widehat{r}, \widehat{s}))$, accept if and only if $J(m, h, \widehat{h}) = \widehat{r}$ and $R(m, h) = Acc$, where $\widehat{h} = F_{pk}(\widehat{u}; \widehat{r}_F)$ with $\widehat{u} = (\widehat{u}_1, \ldots, \widehat{u}_\alpha)$ and $\widehat{u}_i = \widehat{s}_i^e \cdot h^{-\widehat{r}_i} \bmod N$ for $i = 1, \ldots, \alpha$.

Unforgeability. Similar to the scheme $\mathsf{SchUDVS}_2$, thanks to the soundness of the GQ proof of knowledge of RSA inverses, we can prove the DV unforgeability of RSAUDVS assuming the PV-unforgeability of RSAUDVS (i.e. the existential unforgeability under chosen-message attack of the underlying standard RSA signature (GKS, S, V)) and the collision-resistance of the trapdoor hash TH. The concrete result is the following.

Theorem 5 (Strong DV-Unforg. of RSAUDVS**).** *If* RSAUDVS *is PV-unforgeable (UF-PV notion) and* TH *is collision-resistant (CR notion) then*

RSAUDVS *achieves Strong DV-unforgeability (ST-UF-DV notion) in the random-oracle model for J(.). Concretely, the following insecurity bound holds:*

$$\mathbf{InSec}^{\mathrm{ST-UF-DV}}_{\mathsf{RSAUDVS}}(t, q_s, q_v, q_J, q_H) \leq$$
$$2[(q_J + q_v)q_s]^{1/2}\left[\mathbf{InSec}^{\mathrm{UF-PV}}_{\mathsf{RSAUDVS}}(t[S], q_s[S], q_H[S]) + \mathbf{InSec}^{\mathrm{CR}}_{\mathsf{TH}}(t[T])\right]^{1/2}$$
$$+ \frac{2(q_J + q_v)q_s + 1}{2^{l_J}},$$

where $t[S] = t[T] = 2t + O((q_J + q_v)(l_F + l_N) + l_e^2 + l_e T_N)$, $q_s[S] = 2q_s$ *and* $q_H[S] = 2q_H$. *Here we denote by* T_N *the time needed to perform a multiplication in* $\mathbb{Z}_N^*$ *and* $l_e = \log_2(e)$.

Privacy. The privacy of RSAUDVS is unconditional, assuming the perfectly-trapdoor property of the trapdoor hash scheme TH.

Theorem 6 (Privacy of RSAUDVS**).** *If the scheme* TH *is perfectly-trapdoor then* RSAUDVS *achieves complete and perfect unconditional privacy (PR notion).*

6 Scheme Comparison

The following tables compare the security and performance features of the proposed schemes (also shown for comparison is an entry for the bilinear-based UDVS scheme DVSBM [16]). It is evident that $\mathsf{SchUDVS}_1$ is more computationally efficient than $\mathsf{SchUDVS}_2$ but its security relies on a stronger assumption and it also produces slightly longer DV signatures. The RSA-based scheme RSAUDVS has a disadvantage of long DV signature length, assuming a low public exponent. However, the computation is about the same as in the Schnorr-based schemes.

Scheme	Extended Sig.	Hard Problem	Det. Desig?	DV Sig. Length (typ)
$\mathsf{SchUDVS}_1$	Schnorr	SDH	Yes	2.0 kb
$\mathsf{SchUDVS}_2$	Schnorr	DL	No	1.5 kb
RSAUDVS	RSA	RSA	No	11.6 kb
DVSBM	BLS	BDH	Yes	1.0 kb

Table 1. Comparison of UDVS Schemes. The column 'Det Desig?' indicates if the schemes designation algorithm is deterministic. Refer to [17] for assumptions used to compute typical DV sig. lengths.

7 Conclusions

We have shown how to efficiently extend the standard Schnorr and RSA signature schemes into Universal Designated-Verifier Signature schemes, and provided a

Scheme	Desig. Time	Ver. Time
SchUDVS$_1$	2 exp.	1 exp.
SchUDVS$_2$	2 exp. + TH	1 exp. + TH
RSAUDVS	$2(\lceil l_J/\log_2(e)\rceil + 1)$ exp. + TH	$\lceil l_J/\log_2(e)\rceil + 1$ exp. + TH
DVSBM	1 pairing	1 pairing + 1 exp.

Table 2. Comparison of UDVS Schemes Approximate Computation Time. Here we count the cost of computing a product $a^x b^y c^z$ as equivalent to a single exponentiation (exp.) in the underlying group. For RSAUDVS exponent lengths are all $\log_2(e)$. TH denotes the cost of evaluating the trapdoor hash function F_{pk} (typ. 1 exp.).

concrete security analysis of the resulting schemes. One problem of our RSA scheme is that the length of designated signatures is larger than standard RSA signatures by a factor roughly proportional to $k/\log_2(e)$, where k is the security parameter and e is the public exponent. An interesting open problem is to find an RSA based UDVS scheme with designated signatures only a constant factor longer than standard RSA signatures, independent of e.

Acknowledgments. The work of Ron Steinfeld, Huaxiong Wang and Josef Pieprzyk was supported by ARC Discovery Grant DP0345366. Huaxiong Wang's work was also supported in part by ARC Discovery Grant DP0344444.

References

[1] M. Abdalla, M. Bellare, and P. Rogaway. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In *Topics in Cryptology - CT-RSA 2001,* volume 2020 of *LNCS,* pages 143–158, Berlin, 2001. Springer-Verlag. See full paper available at `www-cse.ucsd.edu/users/mihir`.

[2] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proc. 1-st ACM Conf. on Comp. and Comm. Security,* pages 62–73, New York, November 1993. ACM.

[3] D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. In *CRYPTO 2001,* volume 2139 of *LNCS,* pages 213–229, Berlin, 2001. Springer-Verlag.

[4] W. Diffie and M. Hellman. New Directions in Cryptography. *IEEE Trans. on Information Theory,* 22:644–654, 1976.

[5] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions of Identification and Signature Problems. In *CRYPTO'86,* volume 263 of *LNCS,* pages 186–194, Berlin, 1987. Springer-Verlag.

[6] S. Goldwasser, S. Micali, and R. Rivest. A Digital Signature Scheme Secure against Adaptively Chosen Message Attacks. *SIAM Journal on Computing,* 17(2):281–308, 1988.

[7] L.C. Guillou and J.J. Quisquater. A "Paradoxical" Identity-Based Signature Scheme Resulting from Zero-Knowledge. In *CRYPTO '88,* volume 403 of *LNCS,* pages 216–231, Berlin, 1990. Springer-Verlag.

[8] M. Jakobsson, K. Sako, and R. Impagliazzo. Designated Verifier Proofs and Their Applications. In *Eurocrypt '96,* volume 1070 of *LNCS,* pages 143–154, Berlin, 1996. Springer-Verlag.

[9] H. Krawczyk and T. Rabin. Chameleon Signatures. In *NDSS 2000,* 2000. Available at http://www.isoc.org/isoc/conferences/ndss/2000/proceedings/.

[10] T. Okamoto and D. Pointcheval. The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes. In *PKC2001,* volume 1992 of *LNCS,* pages 104–118, Berlin, 2000. Springer-Verlag.

[11] D. Pointcheval and J. Stern. Security Arguments for Digital Signatures and Blind Signatures. *J. of Cryptology,* 13(3):361–396, 2000.

[12] R. L. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM,* 21(2):120–128, 1978.

[13] RSA Laboratories. *PKCS#1 v. 2.1: RSA Cryptography Standard,* 2002.

[14] C. P. Schnorr. Efficient Identification and Signatures for Smart Cards. In *CRYPTO'89,* volume 435 of *LNCS,* pages 239–251, Berlin, 1990. Springer-Verlag.

[15] A. Shamir and Y. Tauman. Improved Online/Offline Signature Schemes. In *CRYPTO 2001,* volume 2139 of *LNCS,* pages 355–367, Berlin, 2001. Springer-Verlag.

[16] R. Steinfeld, L. Bull, H. Wang, and J. Pieprzyk. Universal Designated-Verifier Signatures. In *Asiacrypt 2003,* volume 2894 of *LNCS,* pages 523–542, Berlin, 2003. Springer-Verlag. Full version available at http://www.comp.mq.edu.au/~rons.

[17] R. Steinfeld, H. Wang, and J. Pieprzyk. Efficient Extension of Standard Schnorr/RSA signatures into Universal Designated-Verifier Signatures. Cryptology ePrint Archive, Report 2003/193, 2003. http://eprint.iacr.org/.

Constructing Committed Signatures from Strong-RSA Assumption in the Standard Complexity Model

Huafei Zhu

Department of Infocomm Security, Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore 19613
huafei@i2r.a-star.edu.sg

Abstract. In this paper, we provide the first committed signature provably secure in the standard complexity model based on the strong RSA assumption. The idea behind the construction is that given any valid partial signature of message m, if a co-signer with its auxiliary input is able to generate variables called the resolution of message m such that the distribution of the variables is indistinguishable from those generated by the primary signer alone from the point views of the verifier/arbitrator, then from which a committed signature can be derived.

Keywords: Committed signatures, fair exchange protocols, strong RSA assumption

1 Introduction

In PODC 2003, Park, Chong, Siegel and Ray [15] provided a novel method of constructing fair exchange protocol by distributing the computation of RSA signature. This approach avoids the design of verifiable encryption scheme at the expense of having co-signer store a piece of prime signer's secret key (please refer to [1], [4], [2], [3] for more details). Based on Park et.al's study, Dodis and Reyzin [10] presented a unified model for non-interactive fair exchange protocols which results in a new primitive called committed signatures later. Committed signatures are the following thing: Alice can produce a partial signature to Bob; upon receiving what she needs from Bob, she can convert it to a full signature. If she refuses, the trusted third party Charlie can do it for her upon receipt of partial signature and proper verification that Bob fulfilled his obligation to Alice.

Park, Chong, Siegel and Ray's fair exchange protocol is actually a committed signature scheme since the mechanism of the non-interactive fair exchange is the same thing as a committed signature. Unfortunately this committed signature is totally breakable in the registration phase [10]. Dodis and Reyzin [10] then presented a remedy scheme by utilizing Boldyreva's non-interactive two-party multi-signature scheme [5]. Therefore Dodis and Reyzin's committed signature is the first committed signature provably secure under the Gap Diffie-Hellman assumption in the random oracle paradigm.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 101–114, 2004.

Security in the random oracle model does not imply security in the real world. The existence of committed signature is obvious in the standard complexity model provided the underlying signature schemes are provably secure in the standard complexity model as two signatures with keys (pk_1, sk_1), (pk_2, sk_2), and let $PK = (pk_1, pk_2)$, $SK = (sk_1, sk_2)$ and $\sigma = (\sigma_1, \sigma_2)$ are sufficient to build a secure committed signature. Therefore the challenge problem is to construct a committed signature consistent with a stand-alone signature scheme in the standard complexity model. In this paper, we are able to provide the first committed signature based on the strong RSA assumption. The idea behind the construction is that given any valid partial signature of message m, if a co-signer with its auxiliary input is able to generate variables called the resolution of message m such that the distribution of the variables is indistinguishable from those generated by the primary signer alone from the point views of the verifier/arbitrator, then from which a committed signature can be derived.

The rest of paper is organized as follows: in Section 2, we formalize the security definition of committed signatures, and a committed signature is fully described in the Subsection 3.1, the proof of its security is presented in Subsection 3.2. In Section 4, we construct committed signatures from the point views of real world by providing two efficient schemes with random strings reusing. Finally the conclusion is presented in Section 5.

2 Notions and Definitions

The following definition of committed signatures is formalized the SAME thing as non-interactive fair exchanges introduced by Park, Chong, Siegel and Ray [15] and [10]. Therefore, the committed schemes presented in this report should be viewed as the actual fair exchange protocols working in the real world.

Definition 1 A committed signature involves a primary signer Alice, a verifier Bob and a co-signer (or arbitrator) Charlie, and is given by the following efficient procedures:

- Key generator *KG*: This is an interactive protocol between a primary signer and a co-signer, by the end of which either one of the parties aborts, or the primary signer learns her secret signing key *SK,* the co-signer learns his secret key *ASK,* and both parties agree on the primary signer's public key *PK* and partial verification key *APK;*
- Fully signing algorithm *Sig* and its correspondent verification algorithm *Ver*: These are conventional signing and verification algorithms. $Sig(m, SK)$ run by the primary signer, outputs a full signature σ on m, while $Ver(m, \sigma, PK)$ run by any verifier, outputs 1 (accept) or 0 (reject);
- Partially signing algorithm *PSig* and the correspondent verification algorithm *PVer*: These are partial signing and verification algorithms, which are similar to ordinary signing and verification algorithms, except they can depend on the public arbitration key *APK*. $PSig(m, SK, PK, APK)$, run by the primary signer, outputs a partial signature σ', while $PVer(m, \sigma' PK, APK)$, run by any verifier, outputs 1 (accept) or 0 (reject);

-Resolution algorithm *Res*: This is a resolution algorithm run by the co-signer (arbitrator) in case the primary signer refuses to open her signature σ to the verifier, who in turn possesses a valid partial signature σ' on m and a proof that he fulfilled his obligation to the primary signer. In this case, $Res(m, \sigma', ASK, PK)$ should output a valid full signature of m.

Correctness of committed signatures states that: (1) $Ver(m, Sig(m, SK), PK)$=1;(2) $PVer(m, PSig(m, SK, PK, APK), PK, APK)$=1;*and*(3)Ver(m, $Res(PSig(m, SK, PK, APK), ASK, APK, PK), PK)$=1.

2.1 Security of Committed Signatures

Recall that a committed signature is formalized the same thing as a non-interactive fair exchange. The security of committed signature scheme should consist of ensuring three aspects: security against a primary signer Alice, security against a verifier Bob, and security against a co-signer/abitrator Charlie.

Security against a primary signer Intuitively, a primary signer Alice should not provide a partial signature which is valid both from the point views of a verifier and a co-signer but which will not be opened into the primary signer's full signature by the honest co-signer. More formally:

Let P be an oracle simulating the partial signing procedure *PSig,* and R be an oracle simulating the resolution procedure *Res.* Let k be system security parameter. We require that any probabilistic polynomial time *Adv* succeeds with at most negligible probability in the following experiment.

Experiment 1 (security against primary signer):

1.1: Key generation: $(SK^*, PK, ASK, APK) \leftarrow KG^*(1^k)$, where KG^* denotes the run of key generator KG with the dishonest primary signer by the adversary, and SK^* denotes the adversary's states.

1.2: *Res* oracle query: In this phase, for each adaptively chosen message m_j, the adversary computes its partial signature $\sigma_j{}'$ for m_j. Finally the adversary forward $\sigma_j{}'$ to the oracle R to obtain the full signature σ_j of message m_j, where $1 \leq j \leq p(k)$, and $p(\cdot)$ is a polynomial. At the end of R oracle query, the adversary produces a message and its full signature pair (m, σ), i.e., $(m, \sigma') \leftarrow Adv^R(SK^*, PK, APK)$, $\sigma \leftarrow Adv(m, \sigma', SK^*, APK, PK)$, where $m \neq m_j$, $1 \leq j \leq p(k)$.

1.3. Success of $Adv := [PVer(m, \sigma', APK, PK) = 1 \wedge Ver(m, \sigma, PK) = 0]$.

Definition 2 A committed signature scheme is secure against primary signer attack, if any probabilistic polynomial time adversary *Adv* associated with Resolution oracle, succeeds with at most negligible probability, where the probability takes over coin tosses in $KG(\cdot)$, $PSig(\cdot)$ and $R(\cdot)$.

Security against verifier We consider the following scenario: suppose a primary signer Alice and a verifier Bob are trying to exchange signature in a fair way. Alice wants to commit to the transaction by providing her partial signature. Of course, it should be computationally infeasible for Bob to compute the full signature from the partial signature. More formally, we require that any

probabilistic polynomial time adversary *Adv* succeeds with at most negligible probability in the following experiment:

Experiment 2 (security against verifier):

2.1 Key generation: $(SK, PK, ASK, APK) \leftarrow KG(1^k)$, where *KG* is run by the honest primary signer and honest co-signer. Adversary *Adv* are admitted to make queries to the two orales *P* and *R*.

2.2 *P* and *R* oracle query: For each adaptively chosen message m_j, the adversary obtains the partial signature $\sigma_j{}'$ of message m_j by querying the partial signing oracle *P*. Then the adversary forward $\sigma_j{}'$ to the resolution oracle *R* to obtain the full signature σ_j of message m_j, where $1 \leq j \leq p(k)$, and $p(\cdot)$ is a polynomial. At the end of oracle both *P* and *R* queries, the adversary produces a message-full signature pair $(m, \sigma) \leftarrow Adv^{P,R}(PK, APK)$.

2.3 Success of adversary $Adv := [Ver(m, \sigma, PK) = 1 \wedge m \notin Query(Adv, R)]$, where *Query*(*Adv, R*) is the set of valid queries the adversary *Adv* asked to the resolution oracle *R,* i.e., (m, σ') such that $PVer(m, \sigma') = 1$.

Definition 3 A committed signature scheme is secure against verifier attack, if any probabilistic polynomial time adversary *Adv* associated with partial signing oracle *P* and the resolution oracle *R,* succeeds with at most negligible probability, where the probability takes over coin tosses in $KG(\cdot)$, $P(\cdot)$ and $R(\cdot)$.

Security against co-signer/arbitrator This property is crucial. Even though the co-signer (arbitrator) is semi-trusted, the primary signer does not want this co-signer to produce a valid signature which the primary signer did not intend on producing. To achieve this goal, we require that any probabilistic polynomial time adversary *Adv* associated with partial signing oracle *P,* succeeds with at most negligible probability in the following experiment:

Experiment 3 (security against co-signer / arbitrator):

3.1 Key generation: $(SK, PK, ASK^*, APK) \leftarrow KG^*(1^k)$, where $KG^*(1^k)$ is run by the dishonest co-signer or arbitrator. Adversary *Adv* are admitted to make queries to the partial signing orale *P*.

3.2 *P* oracle query: For each adaptively chosen message m_j, the adversary obtains the partial signature $\sigma_j{}'$ for m_j from the oracle *P,* where $1 \leq j \leq p(k)$, and $p(\cdot)$ is a polynomial. At the end of the partial partial signing oracle query, the adversary produces a message-full signature pair (m, σ), i.e., $(m, \sigma) \leftarrow Adv^P(ASK^*, PK, APK)$.

3.3 Success of adversary $Adv := [Ver(m, \sigma, PK) = 1 \wedge m \notin Query(Adv, P)]$, where *Query*(*Adv, P*) is the set of valid queries *Adv* asked to the partial oracle *P*, i.e., (m, σ') such that $PVer(m, \sigma') = 1$.

Definition 4 A committed signature scheme is secure against co-signer attack, if any probabilistic polynomial time adversary *Adv* associated with partial signing oracle *P*, succeeds with at most negligible probability, where the probability takes over coin tosses in $KG(\cdot)$, $P(\cdot)$.

Definition 5 A committed signature scheme is secure if it is secure against primary signer attack, verifier attack and co-signer attack.

3 Constructing Committed Signatures from Strong RSA Assumption

3.1 Our Committed Signature Scheme

We utilize Zhu's signature as primary building block to construct committed signature scheme [16]. We remark that the use of Zhu's signature is not essential. The Cramer-Shoup's signature including trapdoor hash signature [9], Camenisch and Lysyanskaya [7] and Fischlin's signature scheme [11] are all suitable for our purposes. Nevertheless, among the signatures mentioned above, Zhu's signature is the most efficient.

Zhu's signature scheme Zhu's signature scheme is defined as follows [16]:

- Key generation algorithm: Let p, q be two large safe primes (i.e., $p-1 = 2p'$ and $q-1 = 2q'$, where p', q' are two primes with length $(l'+1)$). Let $n = pq$ and QR_n be the quadratic residue of Z_n^*. Let $X, g, h \in QR_n$ be three generators chosen uniformly at random. The public key is (n, g, h, X, H), where H is a collision free hash function with output length l. The private key is (p, q).
- Signature algorithm: To sign a message m, a $(l+1)$-bit prime e and a string $t \in \{0,1\}^l$ are chosen at random. The equation $y^e = Xg^t h^{H(m)} \bmod n$ is solved for y. The corresponding signature of the message m is (e, t, y).
- Verification algorithm: Given a putative triple (e, t, y), the verifier checks that e is an $(l+1)$-bit odd number. Then it checks the validity of $X = y^e g^{-t} h^{-H(m)} \bmod n$. If the equation is valid, then the signature is valid. Otherwise, it is rejected.

Strong RSA assumption: Strong RSA assumption was introduced by Baric and Pfitzmann [6] and Fujisaki and Okamoto [12]: The strong RSA assumption is that it is hard, on input an RSA modulus n and an element $z \in Z_n^*$, to compute values $e > 1$ and y such that $y^e = z \bmod n$. More formally, we assume that for all polynomial time circuit families A_k, there exists a negligible function $\nu(k)$ such that:

$$\Pr[n \leftarrow G(1^k), z \leftarrow Z_n^*, (e, y) \leftarrow A_k(n, z) : e > 1 \wedge y^e = z \bmod n] = \nu(k)$$

The following lemma, due to Guillou-Quisquater [14], is useful to prove the security of the committed signature scheme.

Guillou-Quisquater lemma Suppose $w^e = z^b$ and $d = \gcd(e, b)$. Then there exists an efficient algorithm computing the (e/d)-th root of z.

Zhu's signature scheme is immune to adaptive chosen-message attack in the sense of Goldwasser, Micali and Rivest [13], under joint assumptions of the strong RSA problem as well as the existence of collision free hash function. Please refer to the appendix for details. Based on Zhu's signature scheme, we are ready to describe the new committed signature below.

Key generation algorithm: We choose two safe primes $p = 2p' + 1$, $q = 2q' + 1$ and compute $N = pq$. Denote the quadratic residue of Z_N^* by QR_N. Let

x, h_1, h_2 be elements chosen uniformly at random from the cyclic group QR_N. Let *PriG* be a prime generator. On input 1^k, it generates $2s+1$ primes, each with bit length $(l+1)$. The prime pair $\{e_{i,1}, e_{i,2}\}$ is indexed by some $i \in I$ $(1 \leq i \leq s)$. The public key (X, g_1, g_2) is computed from x, h_1, h_2 and $(e_{1,2}, e_{2,2}, \cdots e_{s,2})$ as follows:

$$X \leftarrow x^{e_{1,2}e_{2,2}\cdots e_{s-1,2}e_{s,2}} \mathrm{mod} N$$

$$g_1 \leftarrow h_1^{e_{1,2}e_{2,2}\cdots e_{s-1,2}e_{s,2}} \mathrm{mod} N$$

$$g_2 \leftarrow h_2^{e_{1,2}e_{2,2}\cdots e_{s-1,2}e_{s,2}} \mathrm{mod} N$$

Denote a subset of index set in which each index i has been used to sign some message by I_{used}. We then build a public accessible prime list table *PriT* as follows. On input $i \in I_{used}$, *PriT* outputs $\{e_{i,1}, e_{i,2}\}$.

The primary signer's public key *PK* is $(N, X, g_1, g_2,$ *H,PriT,* $I_{used})$. The private key *SK* is $\{x, h_1, h_2, p, q, (e_{i,1}, e_{i,2}), 1 \leq i \leq s)\}$, where H is a publicly known collision-free hash function.

The *APK* of the co-signer is $(N,$ *X,* $g_1, g_2,$ *H, PriT,* $I_{used})$. The secret key of the co-signer *ASK* is $\{x, h_1, h_2, (e_{1,2}, e_{2,2}, \cdots, e_{s,2})\}$.

Partial signing algorithm *PSig* **and correspondent verification algorithm** *PVer*: To sing a message m, we choose $i \in I \setminus I_{used}$ and a random string $t_{i,1} \in \{0,1\}^l$. The equation:

$$y_{i,1}^{e_{i,1}} = X g_1^{t_{i,1}} g_2^{H(m)} \mathrm{mod} N$$

is solved for $y_{i,1}$.

We then update the index I_{used} by accumulating

$$I_{used} \leftarrow I_{used} \bigcup \{i\}$$

The partial signature of message m is $\sigma' = (i, e_{i,1}, t_{i,1}, y_{i,1})$.

On upon receiving a putative partial signature $\sigma' = (i, e_{i,1}, t_{i,1}, y_{i,1})$, the verification algorithm checks whether $i \in I_{used}$ or not, if $i \notin I_{used}$, then it outputs 0, otherwise, it runs *PriT,* on input i to obtain a prime pair $(e_{i,1}, e_{i,2})$, and it outputs 1, i.e., $PVer(m, \sigma') = 1$ if $\sigma'(m)$ satisfies the equation:

$$X = y_{i,1}^{e_{i,1}} g_1^{-t_{i,1}} g_2^{-H(m)} \mathrm{mod} N$$

Full signing algorithm *Sig* **and correspondent verification algorithm** *Ver*: To fully sign the message m, for the given i, we obtain the prime pair $\{e_{i,1}, e_{i,2}\}$ by running *PriT* on input $i \in I_{used}$. Then we choose a random string $t_{i,2} \in \{0,1\}^l$ uniformly at random and compute $y_{i,2}$ from the equation:

$$y_{i,2}^{e_{i,2}} = X g_1^{t_{i,2}} g_2^{H(t_{i,1}||m)} \mathrm{mod} N$$

The corresponding full signature σ of the message m is defined below:

$$\sigma := (i, e_{i,1}, e_{i,2}, t_{i,1}, t_{i,2}, y_{i,1}, y_{i,2})$$

To verify the correctness of full signature scheme σ, the verification algorithm checks whether $i \in I_{used}$ or not, if $i \notin I_{used}$, then it outputs 0, otherwise, it runs *PriT,* on input i to obtain a prime pair $(e_{i,1}, e_{i,2})$. Finally it tests whether the following equations are valid:

$$X = y_{i,1}^{e_{i,1}} g_1^{-t_{i,1}} g_2^{-H(m)} \mathrm{mod} N$$

and

$$X = y_{i,2}^{e_{i,2}} g_1^{-t_{i,2}} g_2^{-H(t_{i,1}||m)} \mathrm{mod} N$$

If both equations are valid, then the verification function outputs $Ver(m, \sigma) = 1$, otherwise, it outputs 0;

Resolution algorithm *Res:* Given a partial signature $\sigma' = (i, e_{i,1}, t_{i,1}, y_{i,1})$ of message m, the co-signer runs the prime list table *PriT* on input $i \in I_{used}$ to obtain the pair of primes $(e_{i,1}, e_{i,2})$, and checks whether $e_{i,1}$ is a component of partial signature σ' (such a prime $e_{i,1}$ is called a valid prime). If it is valid then the co-signer checks the valid of the following equation:

$$y_{i,1}^{e_{i,1}} = X g_1^{t_{i,1}} g_2^{H(m)} \mathrm{mod} N$$

If it is valid, the co-signer then computes:

$$X_i \leftarrow x^{e_{1,2} \cdots e_{i-1,2} e_{i+1,2} \cdots e_{s,2}}$$

$$g_{i,1} \leftarrow h_1^{e_{1,2} \cdots e_{i-1,2} e_{i+1,2} \cdots e_{s,2}}$$

and

$$g_{i,2} \leftarrow h_2^{e_{1,2} \cdots e_{i-1,2} e_{i+1,2} \cdots e_{s,2}}$$

Finally, the co-signer chooses a random string $t'_{i,2} \in \{0,1\}^l$ and computes $y_{i,2}$ from the following equation:

$$y_{i,2} = X_i g_{i,1}^{t'_{i,2}} g_{i,2}^{H(t_{i,1}||m)} \mathrm{mod} N$$

The output of the resolution algorithm is $(i, e_{i,1}, e_{i,2}, t_{i,1}, t'_{i,2}, y_{i,1}, y_{i,2})$
Obviously,

$$X = y_{i,2}^{e_{i,2}} g_1^{-t'_{i,2}} g_2^{-H(t_{i,1}||m)} \mathrm{mod} N$$

-We remark that the choice of random string $t'_{i,2} \in \{0,1\}^l$ in the resolution phase does not dependent on the random string $t_{i,2}$ in the full signature algorithm. If we insist on the same string used in the resolution algorithm *Res,* then the random pair $(t_{i,1}, t_{i,2})$ can be listed as public known random string set which is also indexed by the set *I*.

-We remark that the number of signature is bounded by s, where $s(\cdot)$ is a polynomial of security parameter k. This is an interesting property as a primary signer can specify the number of signatures for each certificate during its validity duration.

-We also remark that the scheme requires both the signer and co-signer to be stateful to keep count $i \in I_{used}$ and so never reuse primes. And the used index set I_{used} updated after each signature generation is apparently assumed to be accessible to the verifier and co-signer.

3.2 The Proof of Security

Theorem 6: The committed signature is secure under the strong RSA assumption and the assumption that H is collision resistant in the standard complexity model.

Proof: Security against the primary signer Alice is trivial since the co-signer holds *ASK* in the protocol.

Security against the verifier Bob: Assume that protocol is not secure against the verifier attack. That is, there is an adversary playing the role of verifier in the actually protocol, who is able to forge a full signature σ of a message m ($m \neq m_i$, $1 \leq i \leq f$) with non-negligible probability after it has queried partial signing oracle and resolution oracle of messages $m_1, \cdots, m_f$, each is chosen adaptively by the adversary. Let $(i, e_{i,1}, e_{i,2}, t_{i,1}, t'_{i,2}, y_{i,1}, y_{i,2})$ be the full signature provided by the partial signing oracle and the resolution oracle corresponding to a set of messages m_i $(1 \leq i \leq f)$. We consider three types of forgeries as that in [9]: 1) for some $1 \leq j \leq f$, $e_{k,2} = e_{j,2}$ and $t'_{k,2} = t'_{j,2}$, where $k \notin \{1, \cdots, f\}$; 2) for some $1 \leq j \leq f$, $e_{k,2} = e_{j,2}$ and $t'_{k,2} \neq t'_{j,2}$, where $k \notin \{1, \cdots, f\}$; 3) for all $1 \leq j \leq f$, $e_{k,2} \neq e_{j,2}$, where $k \notin \{1, \cdots, f\}$. We should show that any forgery scheme of the three types will lead to a contradiction to the assumptions of the theorem. This renders any forgery impossible. By the security definition, the adversary can query the types of oracles: partial signing oracle and resolution oracle. Therefore we should describe the two oracles in the following simulation according to the forgery types defined above.

Type 1 forgery: On input (z, e), where $z \in Z_N^*$, e is a $(l+1)$-bit prime, we choose $(2f-1)$ primes $(e_{i,1}, e_{i,2})$ for $1 \leq i \neq j \leq f$, each with length $(l+1)$-bit. The j-th prime pair is defined by $(e_{j,1}, e_{j,2})$. We compute PK and APK by choosing $z_1, z_2 \in Z_N^*$ uniformly at random and computing

$$g_1 \leftarrow z_1^{2e_{1,1}e_{1,2}\cdots e_{f,1}e_{f,2}} z^{2e_{1,1}e_{1,2}\cdots e_{j-1,1}e_{j-1,2}e_{j,1}e_{j+1,1}e_{j+1,2}\cdots e_{f,1}e_{f,2}}$$

$$g_2 \leftarrow z^{2e_{1,1}e_{1,2}\cdots e_{j-1,1}e_{j-1,2}e_{j,1}e_{j+1,1}e_{j+1,2}\cdots e_{f,1}e_{f,2}}$$

$$X \leftarrow z_2^{2\beta e_{1,1}e_{1,2}\cdots e_{f,1}e_{f,2}} z^{2e_{1,1}e_{1,2}\cdots e_{j-1,1}e_{j-1,2}e_{j,1}e_{j+1,1}e_{j+1,2}\cdots e_{f,1}e_{f,2}(-\alpha)}$$

where $\alpha \in \{0,1\}^{l+1}$ and $\beta \in Z_N$ are chosen uniformly at random.

Since the simulator knows each $e_{i,1}$ $(1 \leq i \leq f)$, therefore it is easy to compute the partial signing oracle of message m_i $(1 \leq i \leq f)$. And it is also easy to compute the resolution of i-th message $i \neq j$ queried to resolution oracle query *Res*. What we need to show is how to simulate the j-th resolution oracle query. This can be done as follows:

$$y_{j,2}{}^{e_{j,2}} = X g_1{}^{t'_{j,2}} g_2{}^{H(t_{j,1}||m_j)}$$

$$= z_2{}^{2\beta \prod_{1,\cdots f}(e_{i,1}e_{i,2})} z_1{}^{2t'_{j,2}\prod_{1,\cdots f}(e_{i,1}e_{i,2})} \times$$

$$z^{2e_{1,1}e_{1,2}\cdots e_{j-1,1}e_{j-1,2}e_{j,1}e_{j+1,1}e_{j+1,2}\cdots e_{f,1}e_{f,2}(-\alpha+t'_{j,2}+H(t_{j,1}||m_j))}$$

Now we set $-\alpha + t'_{j,2} + H(t_{j,1}||m_j) = 0$, i.e., $t'_{j,2} = \alpha - H(t_{j,1}||m_j)$. To show that the simulation is not trivial, we should show that $t'_{j,2}$ is uniformly distributed over $\{0,1\}^l$ with non-negligible amount. Since $\alpha \in \{0,1\}^{l+1}$ is chosen uniformly at random, the probability that $t'_{j,2}$ belongs to the correct interval and it does so with the correct uniform distribution can be computed as follows:

$$\frac{(2^{l+1} - 1 - H(t_{j,1}||m_j) - 2^l + 1) + H(t_{j,1}||m_j)}{(2^{l+1} - 1 - H(t_{j,1}||m_j)) - (-H(t_{j,1}||m_j)) + 1} = 1/2$$

Suppose the adversary is able to forge a faking signature of message m_k, denoted by $(k, e_{k,1}, e_{k,2}, t'_{k,1}, t'_{k,2}, y_{k,1}, y_{k,2})$, where $e_{k,2} = e_{j,2}$ and $t'_{k,2} = t'_{j,2}$, $k \notin \{1, \cdots, f\}$. We can not assume that $e_{k,2} = e_{j,2}$, $t'_{k,2} = t'_{j,2}$ and $y_{k,2} = y_{j,2}$ as H is a collision free hash function. Now we have two equations:

$$y_{k,2}{}^{e_{k,2}} = X g_1{}^{t'_{k,2}} g_2{}^{H(t_{k,1}||m_k)}$$

And

$$y_{j,2}{}^{e_{j,2}} = X g_1{}^{t'_{j,2}} g_2{}^{H(t_{j,1}||m_j)}$$

It follows that

$$(\frac{y_{j,2}}{y_{k,2}})^{e_{j,2}} = g_2{}^{H(t_{j,1}||m_j) - H(t_{k,1}||m_k)}$$

$$= z^{2e_{1,1}e_{1,2}\cdots e_{j-1,1}e_{j-1,2}e_{j,1}e_{j+1,1}e_{j+1,2}\cdots e_{f,1}e_{f,2}(H(t_{j,1}||m_j) - H(t_{k,1}||m_k))}$$

where $e_{j,2} = e$. Consequently, one is able to extract the e-th root of z with non-negligible probability. It contradicts the standard RSA assumption.

Type 2 forgery: On input z and e, where $z \in Z_N^*$, e is a $(l+1)$-bit prime, we choose $(2f-1)$ primes $(e_{i,1}, e_{i,2})$ for $1 \leq i \neq j \leq f$. The j-th prime pair is defined by $(e_{j,1}, e)$. We compute PK and APK by choosign $z_1, z_2 \in Z_N^*$ uniformly at random and computing

$$g_1 \leftarrow z^{2e_{1,1}e_{1,2}\cdots e_{j-1,1}e_{j-1,2}e_{j,1}e_{j+1,1}e_{j+1,2}\cdots e_{f,1}e_{f,2}}$$

$$g_2 \leftarrow z_1{}^{2e_{1,1}e_{1,2}\cdots e_{j-1,1}e_{j-1,2}e_{j,1}e_{j,2}e_{j+1,1}e_{j+1,2}\cdots e_{f,1}e_{f,2}}$$

$$X \leftarrow g_1^{-\alpha} z_2{}^{2e_{1,1}e_{1,2}\cdots e_{j-1,1}e_{j-1,2}e_{j,1}e_{j,2}e_{j+1,1}e_{j+1,2}\cdots e_{f,1}e_{f,2}}$$

where $z_1, z_2 \in Z_N$ and $\alpha \in \{0,1\}^l$ are chosen uniformly at random. Since QR_N is a cyclic group, we can assume that g_1, g_2 are generators of QR_N with overwhelming probability.

Since $e_{i,1}$ for $1 \leq i \leq f$ are known therefore, the partial signing oracle is perfect from the point views of the adversary. To simulate the i-th message m_i $(i \neq j)$ to the resolution oracle, we select a random string $t'_{i,2} \in \{0,1\}^l$ and computes:

$$y_{i,2}^{e_{i,2}} = X g_1{}^{t'_{i,2}} g_2{}^{H(t_{i,1}||m_i)}$$

$$= ((z_1{}^{H(t_{i,1}||m_i)} z_2)^{2e_{1,1}e_{1,2}\cdots e_{i-1,1}e_{i-1,2}e_{i,1}e_{i+1,1}e_{i+1,2}\cdots e_{f,1}e_{f,2}} z^{2e_{i,1}(t'_{i,2}-\alpha)\prod_{s\neq i,j} e_{s,1}e_{s,2}})^{e_{i,2}}$$

The output of resolution oracle is $(i, e_{i,2}, y_{i,2}, t'_{i,2})$.

To sign the j-th message m_j, the signing oracle sets $t'_{j,2} \leftarrow \alpha$ and computes:

$$y_{j,2}{}^{e_{j,2}} = ((z_1{}^{H(t_{j,1}||m_i)} z_2)^{2e_{j,1}\prod_{s\neq j} e_{s,1}e_{s,2}})^{e_{j,2}}$$

where $e_{j,2} = e$.

Let $Res(m_k) = (k, e_{k,2}, y_{k,2}, t'_{k,2})$ be a legal signature generated by the adversary of message $m_k \neq m_i$ for all $1 \leq i \leq f$. By the assumption, we know that

$$y_{k,2}{}^{e_{k,2}} = X g_1{}^{t'_{k,2}} g_2{}^{H(t'_{k,1}||m_k)}$$

and

$$y_{j,2}{}^{e_{j,2}} = X g_1{}^{t'_{j,2}} g_2{}^{H(t'_{j,1}||m_j)}$$

Consequently, we have the following equation:

$$(\frac{y_{k,2}}{y_{j,2}})^{e_{j,2}} = g_1{}^{t'_{k,2}-t'_{j,2}} g_2{}^{H(t'_{k,1}||m_k)-H(t'_{j,1}||m_j)}$$

Equivalently,

$$z^{2(\alpha-t'_{k,2})e_{j,1}\prod_{i\neq j} e_{i,1}e_{i,2}} = (z_1{}^{2e_{j,1}(H(t'_{j,1}||m_j)-H(t'_{k,1}||m_k))\prod_{i\neq j} e_{i,1}e_{i,2}})^{e_{j,2}}$$

Since $t'_{j,2} = \alpha$ and $t_{k,2} \neq t'_{j,2}$, it follows that $\alpha - t'_{k,2} \neq 0$. We then apply Guillou-Quisquater lemma to extract the e-th root of z. This contradicts the standard RSA assumption.

Type 3 forgery: On input z, where $z \in Z_N^*$, we choose $2f$ primes $(e_{i,1}, e_{i,2})$ for $1 \leq i \leq f$ and compute the PK and ASK as follows:

$$g_1 \leftarrow z^{2e_{1,1}e_{1,2}\cdots e_{f,1}e_{f,2}}$$

and

$$g_2 \leftarrow g_1^a, X \leftarrow g_1^b$$

where $a, b \in \{1, n^2\}$.

Since the simulator knows all prime pairs, it follows it can simulate both partial signing and resolution queries. Let $Res(m_k) = (k, e_{k,2}, y_{k,2}, t'_{k,2})$ be a legal signature generated by the adversary of message $m_k \neq m_i$ for all $1 \leq i \leq f$. It yields the equation

$$y_{k,2}{}^{e_{k,2}} = X g_1^{t'_{k,2}} g_2{}^{H(t_{k,1}||m_k)} = z^E$$

where $E = 2(b + t'_{k,2} + aH(t_{k,1}||m_k))e_{1,1}e_{1,2}\cdots e_{f,1}e_{f,2}$

Since we are able to compute the $\frac{e}{E}$-th root of z provided e is a not a divisor of E according to the lemma of Guillou and Qusiquater [14], it is sufficient to show that e is not a divisor of E with non-negligible probability. Due to the the fact that $\gcd(e, e_{1,1}e_{1,2}\cdots e_{f,1}e_{f,2}) = 1$, it is sufficient to show that e is not a divisor of $b + t + aH(t_{k,1}||m_k))$ with non-negligible probability. Since $b \in (1, n^2)$, it follows that one can write $b = b'p'q' + b''$. Therefore, the probability that $b + t + aH(m) \equiv 0 \bmod e$ is about $1/e$.

Security against the co-signer/arbitrator Charlie: Even though the co-signer (arbitrator) is semi-trusted, the primary signer does not want this co-signer to produce valid signature which the primary signer did not intend on producing. In other words, if the co-signer is able to forge a partial signature of a message m, then we make use of Charlie as a subroutine to break the strong RSA assumption. Since Bob holds the correspondent *ASK,* therefore we can assume that Bob succeeds in forging a valid partial signature with non-negligible probability. The simulation is the same as the proof of Zhu's signature, therefore omitted.

4 Conclusion

In this report, we provide the first committed signature from the strong RSA assumption based on Zhu's signature scheme. As the committed signature formalized the same thing as the fair exchange protocol, our scheme is actually a fair exchange protocol which is provably secure in the standard complexity model. We should admit that the scheme does not quite achieve the consistency with Zhu's signature scheme with a stand-alone signature fully. How to construct a compactly specified one is our further research.

References

1. G. Ateniese, Efficient Verifiable Encryption (and Fair Exchange) of Digital Signatures. In 6th ACM Conference on Computer and Communications Security (ACM CCS'99), 138-146.
2. N. Asokan, M. Schunter, M. Waidner: Optimistic Protocols for Fair Exchange. ACM Conference on Computer and Communications Security 1997: 7-17.
3. N. Asokan, V. Shoup, M. Waidner: Optimistic Fair Exchange of Digital Signatures (Extended Abstract). EUROCRYPT 1998: 591-606.
4. F. Bao, R. Deng, W. Mao, Efficient and Practical Fair Exchange Protocols, Proceedings of 1998 IEEE Symposium on Security and Privacy, Oakland, pp. 77-85, 1998.
5. A. Boldyreva. Efficient threshold signatures, multisigntaures and blind signatures based on the Gap Diffie Helman group signature scheme. PKC 2003, LNCS 2567.
6. N. Braic and B. Pfitzmann. Collision free accumulators and fail-stop signature scheme without trees. Eurocrypt'97, 480-494, 1997.
7. J.Camenisch, A. Lysyanskaya. A Signature Scheme with Efficient Protocols. SCN 2002: 268-289.
8. Jan Camenisch, Markus Michels: Proving in Zero-Knowledge that a Number Is the Product of Two Safe Primes. EUROCRYPT 1999:107-122
9. R. Cramer and V. Shoup. Signature scheme based on the Strong RAS assumption. 6th ACM Conference on Computer and Communication Security, Signapore, ACM Press, November 1999.
10. Y.Dodis, L. Reyzin. Breaking and Repairing Optimistic Fair Exchange from PODC 2003, ACM Workshop on Digital Rights Management (DRM), October 2003.

11. Marc Fischlin: The Cramer-Shoup Strong-RSASignature Scheme Revisited. Public Key Cryptography, 2003: 116-129.
12. E. Fujisaki, T. Okamoto. Statistical zero-knowledge protocols to prove modular polynomial relations. Crypto'97, LNCS 1294, Springer-verlag, 1997.
13. S. Goldwasser, S. Micali, R. Rivest: A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. SIAM J. Comput. 17(2): 281-308, 1988.
14. L. Guillou, J. Quisquater. A practical zero-knowledge protocol fitted to security microprocessors minimizing both transmission and memory. Eurocrypto'88, 123-128, 1988.
15. J. M. Park, E. Chong, H. Siegel, and I. Ray. Constructing Fair-Exchange Protocols for E-Commerce Via Distributed Computation of RSA Signatures, PODC 2003, 172-181.
16. Huafei Zhu. New Digital Signature Scheme Attaining Immunity to Adaptive Chosen-message attack. Chinese Journal of Electronics, Vol.10, No.4, Page 484-486, Oct, 2001.

Appendix: A Formal Proof of Zhu's Signature Scheme

Claim: Zhu's signature scheme is immune to adaptive chosen-message attack under the strong RSA assumption and the assumption that H is a collision resistant.

Proof: Assume that the signature scheme is NOT secure against adaptive chosen message attack. That is, there is an adversary, who is able to forge the signature (e, t, y) of a message $m(m \neq m_i, 1 \leq i \leq f)$ with non-negligible probability after it has queried correspondent signature of each message $m_1, \cdots, m_f$, which is chosen adaptively by the adversary. Let $(e_1, t_1, y_1), \cdots, (e_f, t_f, y_f)$ be signatures provided by the signing oracle corresponding to a set of messages $m_1, \cdots, m_f$. We consider three types of forgeries: 1) for some $1 \leq j \leq f$, $e = e_j$ and $t = t_j$; 2) for some $1 \leq j \leq f$, $e = e_j$ and $t \neq t_j$; 3) for all $1 \leq j \leq f$, $e \neq e_j$. We should show that any forgery scheme of the three types will lead to a contradiction to the assumptions of the theorem. This renders any forgery impossible.

Type 1-Forger : We consider an adversary who chooses a forgery signature such that $e = e_j$ for a fixed j: $1 \leq j \leq f$, where f is the total number of the queries to the signing oracle. If the adversary succeeds in a signature forgery as type1 with non-negligible probability then given n, we are able to compute $z^{1/r}$ with non-negligible probability, where r is a $(l+1)$-bit prime. This contradicts to the assumed hardness of the standard RSA problem. We state the attack in details as follows: given $z \in Z_n^*$ and r, we choose a set of total $f-1$ primes with length $(l+1)$-bit $e_1, ...e_{j-1}, e_{j+1}, ..., e_f$ uniformly at random. We then create the correspondent public key (X, g, h) of the simulator as follows: given $z \in Z_n^*$ and r, we choose a set of total $f-1$ primes with length $(l+1)$-bit $e_1, ...e_{j-1}, e_{j+1}, ..., e_f$ uniformly at random. We choose $w, v \in Z_n$ uniformly at random, and compute $h = z^{2e_1 ... e_{j-1} e_{j+1} ... e_f}$, $g = v^{2e_1 \cdots e_f} z^{2e_1 ... e_{j-1} e_{j+1} \cdots e_f}$ and

$X = w^{2\beta e_1 \cdots e_f} z^{2e_1 \ldots e_{j-1} e_{j+1} \ldots e_f (-\alpha)}$, where $\alpha \in \{0,1\}^{l+1}$ and $\beta \in Z_n$ are chosen uniformly at random.

Since the simulator knows each e_i, therefore it is easy to compute the i-th signing query. What we need to show is how to simulate the j-th signing query. This can be done as follows:

$$y_j^{e_j} = X g^{t_j} h^{H(m_j)} = (w^{\beta} v^{t_j})^{2e_1 \cdots e_f} z^{2e_1 \ldots e_{j-1} e_{j+1} \ldots e_f (-\alpha + t_j + H(m_j))}$$

Now we set $-\alpha + t_j + H(m_j) = 0$, i.e, $t_j = \alpha - H(m_j)$.

To show the simulation above is non-trivial, we should show t_i is uniformly distributed over $\{0,1\}^l$ with non-negligible amount. Since $\alpha \in \{0,1\}^{l+1}$ is chosen uniformly at random, i.e., $0 \leq \alpha \leq 2^{l+1} - 1$, the probability t_j belongs to the correct interval and it does so with the correct uniform distribution can be computed as follows:

$$\frac{(2^{l+1} - 1 - H(m_j) - 2^l + 1) + H(m_j)}{(2^{l+1} - 1 - H(m_j)) - (-H(m_j)) + 1} = 1/2$$

Suppose the adversary is able to forge a faking signature of message m, denoted by (e, y, t), such that $e_j = e(= r)$, $t_j = t$. Notice that one can not assume that $e_j = e$, $t_j = t$ and $y_j = y$, since H is a collision free hash function. Now we have two equations: $y_j^e = X g^t h^{H(m_j)}$ and $y^e = X g^t h^{H(m)}$. Consequently, we obtain the equation:

$$(\frac{y_j}{y})^e = h^{H(m_j) - H(m)} = z^{2e_1, \ldots e_{j-1}, e_{j+1}, \ldots, e_f (H(m_j) - H(m))}$$

It follows that one can extract the e-th root of z with non-negligible probability. Therefore, we arrive at the contradiction of the standard hardness of RSA assumption.

Type 2-Forger: We consider an adversary who succeed in forging a valid signature such that $e = e_j$, $t \neq e_j$ for a fixed j: $1 \leq j \leq f$, where f is the total number of the queries to the signing oracle. If the adversary succeeds in a signature forgery as type1 with non-negligible probability then given n, we are able to compute $z^{1/r}$ with non-negligible probability for a given z and r, where r is a $(l+1)$-bit prime. This contradicts to the assumed hardness of the standard RSA problem. We state the attack in details as follows: given $z \in Z_n^*$ and r, we choose a set of total $f - 1$ primes with length $(l+1)$-bit $e_1, \ldots e_{j-1}, e_{j+1}, \ldots, e_f$ at random. We then create the correspondent public key (X, g, h) of the simulated signature scheme as follows: $g = z^{2e_1 \ldots e_{j-1} e_{j+1} \ldots e_f}$, $h = v^{2e_1 \ldots e_f}$ and $X = g^{-\alpha} w^{2e_1 \cdots e_f}$, where $w, v \in Z_n$ and α is a l-bit random string. Since QR_n is a cyclic group, we can assume that g, h are generators of QR_n with overwhelming probability. To sign the i-th message $m_i (i \neq j)$, the signing oracle selects a random string $t_i \in \{0,1\}^l$, and computes:

$$y_i^{e_i} = ((w v^{H(m_i)})^{2e_1 \ldots e_{i-1} e_{i+1} \ldots e_f} z^{2(t_i - \alpha) \Pi_{s \neq i, s \neq j} e_s})^{e_i}$$

The output of the signing oracle is a signature of message m_i, denoted by $\sigma(m_i) = (e_i, y_i, t_i)$.

To sign the j-th message m_j, the signing oracle, sets $t_j \leftarrow \alpha$ and computes:

$$y_j{}^{e_j} = ((wv^{H(m_j)})^{2\Pi_{s\neq j}e_s})^{e_j}$$

The output of the signing oracle is a signature of message m_j, denoted by $\sigma(m_j) = (e_j, y_j, t_j)$.

Let $\sigma(m) = (e, y, t)$ be a valid signature forged by the adversary of message m. By assumption, we know that $y^e = Xg^t h^{H(m)}$. Consequently, we have the following equation:

$$g^{t_j} h^{H(m_j)} y_j{}^{e_j} = g^t h^{H(m)} y^e$$

Equivalently

$$z^{2(\alpha-t)\Pi_{i\neq j}e_i} = (v^{2(H(m)-H(m_j))\Pi_{i\neq j}e_i} \frac{y}{y_j})^{e_j}$$

Since $t_j = \alpha$ and $t \neq t_j$ by assumption, it follows that $t \neq \alpha$. We then apply Guillou-Quisquater lemma to extract the r-th root of z, where $r = e_j$.

Type 3-Forger: We consider the third type of the attack: the adversary forgery is that for all $1 \leq j \leq f$, $e \neq e_j$. If the adversary succeeds in forgery with non-negligible probability, then given n, a random $z \in Z_n^*$, we are able to compute $z^{1/d}$ $(d > 1)$ with non-negligible probability, which contradicts to the assumed hardness of strong RSA assumption. We state our attack in details as follows: we generate g and h with the help of z. We define $g = z^{2e_1 \cdots e_f}$ and $h = g^a$, where $a \in (1, n^2)$, is a random element. We can assume that g is a generator of QR_n with overwhelming probability. Finally, we define $X = g^b$, where $b \in (1, n^2)$. Since the simulator knows the all e_j, the signature oracle can be perfectly simulated. Let (e, t, y) be a forgery signature of message m. It yields the equation $y^e = Xg^t h^{H(m)} = z^E$, where $E = (b + t + aH(m))2e_1 ... e_f$. Since we are able to compute (e/E)-th root of z provided e is a not a divisor of E according to the lemma of Guillou and Qusiquater, it is sufficient to show that e is not a divisor of E with non-negligible probability. Due to the the fact that $\gcd(e, e_1 e_2 \cdots e_f) = 1$, it is sufficient to show that e is not a divisor of $b+t+aH(m)$ with non-negligible probability. Since $b \in (1, n^2)$, it follows that one can write $b = b'p'q' + b''$. Therefore, the probability that $b + t + aH(m) \equiv 0 \bmod e$ is about $1/e$.

Constant Round Authenticated Group Key Agreement via Distributed Computation*

Emmanuel Bresson[1] and Dario Catalano[2]

[1] Cryptology Department, CELAR, 35174 Bruz Cedex, France
Emmanuel.Bresson@polytechnique.org
[2] Dépt. d'informatique, École normale supérieure, 75230 Paris Cedex 05, France.
Dario.Catalano@ens.fr

Abstract. A group key agreement protocol allows a set of users, communicating over a public network, to agree on a private session key. Most of the schemes proposed so far require a linear number (with respect to the number of participants) of communication rounds to securely achieve this goal. In this paper we propose a new constant-round group key exchange protocol that provides efficiency and privacy under the Decisional Diffie-Hellman assumption. Our construction is practical, conceptually simple and it is obtained by taking advantage of the properties of the El-Gamal encryption scheme combined with standard secret sharing techniques.

1 Introduction

Group key agreement protocols allow several parties to come up with a common secret from which a session key can be derived. Hence, these protocols are likely to be used in numerous group-oriented scenarios, such as video conferencing, collaborative applications, secure replicated database, in order to achieve secure multicasting network layer among the parties. Typically, the parties hold some long-term keys that enable them to communicate privately from point to point, or to authenticate messages. Another setting considers password-based authentication, which we are not dealing with in this paper. The final goal of group key exchange protocols is to efficiently implement "secure" multicast channels. In order to specify what "efficiently" and "secure" mean, one may consider some desirable properties for a group key agreement protocol. *Efficiency,* while not measuring security, is to be considered as a crucial property when designing key agreement protocols and it is quantified as the number of communication rounds, as well as the space and computing resources required to agree on the final key. Indeed, limiting the number of rounds can be of prime importance in many real-life applications. Consider for example the case of a group where some (or all the) participants have a slow network connection. In such a situation the efficiency of the entire protocol can be severely degraded even if the "slow guys"

* Extended abstract. A full version of this paper can be found at www.di.ens.fr/~catalano.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 115–129, 2004.

constitute a very small minority of the group. Other scenarios where reducing the number of rounds is important are all those applications where many players are involved, or where many keys have to be exchanged.

Combining efficiency and security is not a trivial task. One of the most basic security property that is required to a group key agreement protocol is the so-called *contributory:* all the parties are ensured to properly contribute to the final secret value and to its distribution — in other words, no party should be able to impose the value of the session key which should be uniformly distributed over the session key space. On top of that a key agreement scheme should guarantee some *privacy* property for final session key, i.e. that no eavesdropper should be able to gain information (at least in some computational sense) about the key after having seen the messages exchanged between the parties being involved in the protocol. In general, however, a group key agreement should preserve privacy even in the case on which the network is under control of some malicious adversary that may try to modify any message sent among the players. Thus the main features we would like to find in a Group Key Agreement scheme are security and efficiency, in the presence of an *active* adversary. The typical approach to the problem [1,5,8,9] requires some data to go through the complete set of parties, which by sequentially adding some private contribution, "build" the actual key in a linear number of rounds of communication. The main problem with this approach is, of course, that it may lead to very slow protocols. To improve on communication complexity the natural solution is to try to devise a scheme that allows for simultaneous sending of contributions.

So, at the very end, the basic problem of group key agreement can be simplified as follows: we want a set of players to agree on some random value that they will later, privately, reconstruct to use as shared key. Written in this way the problem seems to be quite similar to the standard multi-party computation goal where a group of players wants to compute the output of a public function when the input is shared among the participants. There is a crucial difference however: in the multi-party computation setting the output of the function is, in general, kept *shared* and may be publicly reconstructed if some conditions are met. In the group key agreement setting, on the other hand, we want the players to be able to *privately* reconstruct the secret. In other words, the goal of the key agreement is to establish a random value that at the end of the protocol should be disclosed to the players only. In this paper we basically combine standard secret sharing techniques [30] with the use of El-Gamal cryptosystem [20] to make this goal possible.

Related work – Some formal models for studying security of the session key were initiated by Bellare and Rogaway [5,6] and further refined by Blake-Wilson *et al.* [7,8]. Another formal model is based on the multi-party simulatability technique and was initiated by Bellare, Canetti and Krawczyk [2], and refined by Shoup [31]. Some classical examples of group key agreement protocols dealing with privacy are the generalizations of the original Diffie-Hellman paper [16], whose first proposals can be traced back to Ingemarsson *et al.* [23]. Some more sophisticated schemes [10,32] rely on the so-called *group Diffie-Hellman assump-*

tions, for which some reductions can be found in [11], while others are based on more heuristic, quite non-standard[3] assumptions [18]. Let us also mention some proposed schemes that are based on elliptic curve cryptography: Joux [24] proposed a single round method for 3-party Diffie-Hellman key agreement using pairings. However, a generalization based on multi-linear forms is still an open problem [19].

As we said, a major issue of such protocols consists in efficiency, and this is especially true when considering large groups or dynamic peer group key agreement. Some protocols offering provable security have been recently analyzed by Bresson *et al.* [9,10]; they are essentially derived from an article by Steiner *et al.* [32]. However they require a linear number of communication rounds. In [12], Burmester and Desmedt proposed a very efficient, elegant protocol that needs only two rounds (three rounds when considering the confirmation step). The main advantage of constant round protocols is that the impact of "slow guys" is reduced, in the sense that 1 or several slow connections have essentially the same impact on efficiency. Burmester and Desmedt provide a security proof that reduces the privacy (one-wayness) of the session key to the (computational) Diffie-Hellman problem. However no proof of security (in the stronger sense of semantic security [22]) is provided in the original paper. Only recently, Katz and Yung [25] proposed a more general framework that provides a formal proof of security for this protocol, based on the DDH assumption. An interesting contribution of their paper is a scalable compiler that transforms any group key exchange protocol secure against a passive adversary into one that is secure against an active adversary, controlling the network.

In 1999, Li and Pieprzyk [26] proposed a key agreement protocol based on secret sharing techniques. They use the well-known polynomial secret sharing *à la* Shamir [30] to reconstruct a session key. While their work leads to a constant-round protocol and may appear quite similar to ours, it is actually less efficient. First of all they adopt an $(n+1)$-out-of-$2n$ sharing scheme and need to resort to secure channel to guarantee secrecy. In our case, on the other hand, we can use an n-out-of-n secret sharing scheme and no additional assumption is required. Furthermore in [26] to recover the secret the parties are required to perform Lagrange interpolation on the exponents. We emphasize that working in the exponents implies a relatively inefficient scheme, requiring $O(3n)$ exponentiations per player.

Our contributions – In this work, we propose a constant round key exchange protocol, based on secret sharing techniques, and using an asynchronous network. Our scheme is very efficient in terms of communication between the players (only two rounds of communications — plus a confirmation additional round — are required) and provides security (even with respect to parallel executions) under the well known Decisional Diffie-Hellman assumption. As noted above, only very few schemes proposed so far offer both authentication and privacy under standard assumptions [10,25]. We emphasize that our solution achieves comparable

[3] These assumptions make use of "multiple-decker" exponents, and are not easily related to DH.

bandwidth (in terms of the number of total bit per player exchanged) with respect to all previously proposed schemes. Also, if preprocessing is possible our protocol requires only - roughly - 2 exponentiations per player. Moreover we believe that our proposal allows for a more general approach to the problem. Indeed almost all previously suggested solutions are somehow generalizations of the basic Diffie-Hellman key exchange protocol [16], and thus are inherently related to the underlying (computational or decisional) assumptions. Finding alternative to existing solutions is not only a common practice in cryptography but a line of research of fundamental importance in practice. In this sense, in our case, the reduction to the decisional Diffie-Hellman assumption comes *solely* from the fact that we are adopting the El Gamal cryptosystem as underlying encryption primitive (to take advantage, in terms of efficiency, of its nice properties). However, we stress here, that up to some loss in efficiency, it remains possible to successfully implement our protocol using a different semantically secure cryptosystem, relying on alternative intractability assumptions. One could even imagine a scheme in which the data are encrypted point-to-point using a symmetric encryption scheme (the drawback being there the number of secret keys).

2 The Model

Players and network – We consider a network of n players $P_1, \ldots, P_n$, that are connected by point-to-point channels. We assume that each channel can be authenticated by the use of an underlying secure signature scheme. Thus, as already said, we consider an existing PKI and do not deal with password-authentication. We are based on an asynchronous network, in which messages can be delivered in arbitrary order. Moreover, and unless explicitly mentioned, we assume that each player can send several messages at the same time (multi-send property); this does not imply that all receivers will get the same message (broadcast). By saying that player A sends a message privately to B we intend A sending an encrypted message (with respect to B's public key) to B.

Adversary – The network is likely to be faulty, that is, not reliable because of attacks. To take into account such attacks, including those by "malicious" adversaries, we consider an active adversary $\mathcal{A}$ that has full control over the network. In particular we model this adversary as able to read, delete, and modify any message sent on the network. We stress that, as in previously proposed schemes, $\mathcal{A}$ does not have *any* control on the players themselves, and in particular, can not read their private memory[4].

Rushing attacks – We will assume that no player significantly deviates from the protocol, however we enable some players (but not all of them) to choose their contribution to the key according to some arbitrarily biased distribution

[4] More precisely, $\mathcal{A}$ cannot access the storage of the session key; when considering forward-secrecy, one may consider that $\mathcal{A}$ *partially* corrupts the private memory of a player, and gets the long-term key.

(however we assume the adversary *does not* have any knowledge of such bias). Note that this allows for some player to adopt a *rushing* behavior by which he waits to receive the messages of the remaining parties (in a given round of communication) before sending his own. We stress, however, that this does not mean that rushing players do not follow the intructions nor that they follow instructions in a different order; it just means they choose their nonces non-uniformly, and if possible after the others. Moreover, we will assume that at least one player is *completely honest.*

Security notions – Our goal is to provide protocols allowing a pool of players to jointly agree on a common secret session key, in a presence of a malicious adversary (which includes the framework of a faulty network). We consider the following security notions, most of them are defined in [1].

Completeness means that, if the adversary is completely passive, the protocol terminates with each player holding a session key, which is the same for all of them.

Privacy means that whatever the adversary does, it cannot gain any information about the session key, if such a key is set (that is, if the protocol does not abort). In particular, it means that nobody outside of the group is able to compute the session key (*implicit authentication*).

Contributory means that each player is ensured to contribute equally to the final value of the key, and in particular, nobody can bias the distribution of the key.

Confirmation property encompasses the fact that a given player can be ensured a message has been delivered to other players. However, note that the receiver is not ensured that its confirmation has been delivered to the sender (unless using a confirmation again, which leads to infinite recursion). Such a network model thus needs to use time-out methods to abort a protocol if needed. Confirmations are used to achieve *explicit authentication,* by which every player has proof the group holds the same key.

Notations – Let ℓ be a security parameter. In the following we denote with $\mathbb{N}$ the set of natural integers and with $\mathbb{R}^+$ the set of positive real numbers. We say that a function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}^+$ is *negligible* if for every polynomial $\rho(\ell)$ there exists a $\ell_0 \in \mathbb{N}$ s.t. for all $\ell > \ell_0$, $\mathsf{negl}(\ell) \leq 1/\rho(\ell)$. For $a, b \in \mathbb{N}$ we write $a|b$ if a divides b. If A is a set, then $a \leftarrow A$ indicates the process of selecting a at random and uniformly over A (which in particular assumes that A can be sampled efficiently).

2.1 The Formal Model

Players – We consider multiple, potentially parallel executions of the protocol; each player involved in the group has thus many *instances,* also called *oracles* running parallel sessions. The instances are seen as processes running on a given machine: some data (long-term key, public parameters) are shared, some data are specific to a process (eg, the session key). We assume that all signed messages are headed with sessions IDs, which uniquely identify each parallel run. We

denote by sk_i^t the session key computed by player P_i in session whose ID is t. We consider a group of players whose membership is fixed, ie, there is no "Join" or "Remove" manipulations.

Adversarial capabilities – The adversary $\mathcal{A}$ is formalized through several *queries* describing possible interactions with oracles. Following [9], we define four types of queries: the Send-query is used to send arbitrary messages to an oracle; the Reveal-query is used to deal with known-key attacks, by revealing to $\mathcal{A}$ the session sk_i^t key hold by an oracle; the Corrupt-query leaks the long-term data L_i and allows to consider forward-secrecy; finally the Test-query is used to model the semantic security: it returns either the session key, or a random string, $\mathcal{A}$ being to guess which of the two cases.

Necessary conditions – A few straightforward conditions must be satisfied in order to properly use this model. These conditions are used in [1,9] to define the *Freshness* of a session key. First, a Reveal-query makes sense only if an oracle has already *accepted* a session key. Second a Test -query can be asked only once in the entire attack. Third, a Test-query must be asked before *any* Corrupt-query (asked to the Test-ed oracle or not). Four, a Test-query cannot be asked on a session for which some oracles have accepted and have been Reveal-ed. The last three requirements ensure that the Test makes sense, that is, the session key is not "obviously" known by the adversary through basic means.

Definitions – We say that $\mathcal{A}$ *disrupts* an (instance of) player if it does not honestly relay the messages sent by this oracle (i.e., $\mathcal{A}$ is an active adversary generating faults on the network), but is still unable to access this player's internal data. When dealing with forward-secrecy, we say that $\mathcal{A}$ *corrupts* a player if it can get his long-term key. We denote L_i the long-term key used by player P_i.

We say that a Group Key Agreement Protocol, is secure if for any adversary $\mathcal{A}$ controlling the network the following four conditions are met.

Completeness: If $\mathcal{A}$ does not disrupt any oracle in a session t, then at the end of the protocol, there exists sk^t (which is efficiently computable) such that for all $i \in \{1, \ldots, n\}$ we have $sk_i^t = sk^t$.

Contributory (uniformity): If $\mathcal{A}$ does not disrupt any oracle in a session, then sk is uniformly distributed in the key space $\mathcal{K}$.

Privacy: We formalize this property as follows. Let $\mathcal{B}$ be a challenger facing the adversary and that runs the protocol, controlling all the players involved in the key exchange protocol being attacked. Let κ_1 be the corresponding session key computed by the members. On a Test-query, $\mathcal{B}$ chooses a random string κ_0 in the key space $\mathcal{K}$. Then it gives to the adversary either κ_0 or κ_1 (with equal probability). When terminating its attack, $\mathcal{A}$ should output a "guess" for the hidden bit b. We say that the protocol establishes a *private* session key sk if there exists a negligible function negl such that for sufficiently large ℓ, we have:

$$\mathsf{Adv}(\mathcal{A}) = 2\left(\Pr\left[\mathcal{A}(\mathcal{V}, \kappa_b) = b \,\middle|\, \begin{array}{c} \kappa_0 \leftarrow \mathcal{K};\quad \kappa_1 = sk \\ b \leftarrow \{0,1\} \end{array}\right] - \frac{1}{2}\right) = \mathsf{negl}(\ell)$$

Assumption 1 (DDH Assumption) *Let p and q two primes such that $|q| = \ell$ and $q|p-1$ and g an element of order q in $\mathbb{Z}_p^\star$. Let $Q = \langle g \rangle$. There exists a negligible function* negl *such that for sufficiently large ℓ, for any probabilistic polynomial-time distinguisher Δ, we have:*

$$\mathsf{Adv}^{\mathsf{ddh}}(\Delta) = \left| \Pr_{x,y}[\Delta(g, g^x, g^y, g^{xy}) = 1] - \Pr_{x,y,z}[\Delta(g, g^x, g^y, g^z) = 1] \right| = \mathsf{negl}(\ell)$$

Informally this assumption states that given the two elements $X = g^x \bmod p$ and $Y = g^y \bmod p$ the value $Z = g^{xy} \bmod p$ is indistinguishable from a random one in Q (see [22] for a definition of computational indistinguishability).

3 The Proposed Scheme

We start with an informal description of our protocol. The goal here is to highlight the main ideas underlying our construction without going too much into technical details.

Overview of the protocol – We will assume that each player P_i holds a pair of matching private/public key (x_i, h_i), where $h_i = g^{x_i} \bmod p$. We denote by $C_{i,j}(m, \alpha)$ an El-Gamal encryption of a message m under key h_j, using random α. Intuitively $C_{i,j}$ can be seen as an encrypted message sent from player P_i to player P_j.

The proposed protocol goes as follows. Every player P_i uniformly chooses a random value a_i as his own contribution to the key exchange protocol and a randomizer pad r_i. P_i proceeds by encrypting a_i, under the public key of every remaining player, and sends the ciphertext $C_{i,j}$ to player P_j. Moreover P_i randomly chooses a polynomial $f_i(z)$ of degree $n-1$ in $\mathbb{Z}_q^\star$ such that $f_i(0) = r_i$ and sends to player P_j the value $f_i(j)$. Once this first stage is over every player P_j sums the received $f_i(j)$ and multiplies the received ciphertexts (that is, corresponding to all indices but its own). Let us call C_j the resulting product and let δ_j be the plaintext corresponding to C_j. Note that, because of the homomorphic properties of the El-Gamal encryption scheme, the quantity $\delta_j \cdot a_j \bmod p$, is exactly $a = a_1 \cdots a_n \bmod p$, and, of course, it is the same for all the players involved in the protocol. Similarly the quantity $f(i)$, obtained by summing up all the $f_j(i)$'s, will be a share of a unique polynomial $f(z)$ such that $f(0) = r$ where $r = r_1 + \ldots + r_n \bmod q$. So to conclude the protocol the parties compute r, by interpolating $f(z)$ over $\mathbb{Z}_q^\star$ and set their session key $sk = a \cdot g^r$.

Dealing with rushing scenarios – One may wonder why we need to distribute encryptions of a_i, shares of r_i and then define the session key as $sk = \prod_{i=1}^n a_i \cdot g^r \bmod p$ rather than simply distribute encryptions of a_i and set the final key as $sk = \prod_{i=1}^n a_i \bmod p$. As a matter of fact, this second solution may be possible in a non-rushing scenario where all the parties are assumed to maintain a *completely* honest behavior, and using *un-biaised* pseudo-random generators. In our case, as sketched in section 2 a player may decide to choose his contribution *after* having received all those of the remaining parties. Thus he could,

Authenticated Group Key Agreement Protocol

Public Parameters: Two primes p, q such that $q|p-1$. A subgroup $Q = \langle g \rangle$ of order q. An hash function $\mathcal{H}$ modeled as a random oracle, and $\mathcal{ID}$ be the current session ID.

Public inputs: The players'public keys h_i, for $i = 1, \ldots, n$.

Private input (for player i): A value x_i such that $h_i = g^{x_i} \bmod p$.

In a preprocessing stage player P_i runs a signature generation algorithm `SigGen` to obtain a couple of matching signing and verification keys (SK_i, VK_i).

First Round – Each player P_i does the following:

1. Choose $a_i \leftarrow Q$
2. Choose $r_i, b_{i,1}, \ldots, b_{i,n-1} \leftarrow \mathbb{Z}_q$.
3. Define $f_i(z) = r_i + b_{i,1}z + \ldots + b_{i,n-1}z^{n-1} \bmod q$
4. For each $j = 1 \ldots n$ $(j \neq i)$
   Choose $k \leftarrow \mathbb{Z}_q$ and set $C_{i,j} = (A_{i,j}, B_{i,j}) = (g^k \bmod p, h_j^k a_i \bmod p)$.
5. Send to player P_j the values $C_{i,j}$, $f_i(j)$ and $\sigma_{i,j} = \mathtt{Sign}_{SK_i}(C_{i,j}||f_i(j)||\mathcal{ID})$.

Second Round – Once having received all the values above each player P_i does the following: (if P_i receives less than $n-1$ triplets $(C_{j,i}, f_j(i), \sigma_{j,i})$ he aborts the protocol)

1. Check the authentication (signature) of all received values. If the check fails the player aborts the protocol.
2. Multiply the received ciphertexts: let $A_i = \prod_{j \neq i} A_{j,i} \bmod p$ and $B_i = a_i \cdot \prod_{j \neq i} B_{j,i} \bmod p$.
3. Decrypt the result to define the value $a_{(i)} = B_i / A_i^{x_i}$.
4. Compute
$$f_i = f_i(i) + \sum_{j \neq i} f_j(i) \bmod q$$
as his share of a $(n-1)$-degree polynomial $f(z)$ whose free term we indicate with r.
5. Send to other players the values f_i and $\omega_i = \mathtt{Sign}_{SK_i}(f_i||\mathcal{ID})$.

Third Round –

1. The players interpolate $f(z)$ and retrieve r.
2. Player P_i defines its session seed as
$$sk_{(i)} = a_{(i)} \cdot g^r \bmod p.$$

Confirmation Step: Compute $s_i = \mathcal{H}(sk_{(i)}||\mathcal{ID})$ and broadcast this value together with its signature $\gamma_i = \mathtt{Sign}_{SK_i}(s_i||\mathcal{ID})$.
If the n broadcasted values are all the same, set the final key as
$$sk = \mathcal{H}(sk_{(i)})$$

Fig. 1. Pseudo-code description for the Group Key Agreement Protocol

arbitrarily, set the value for the final key. In order to avoid such a situation, in our protocol we distinguish two stages: during the first one every player sends encryptions of a_i and waits for all the other guys to do the same. Then, once he has received all the shares he proceeds to the second stage by disclosing his $f(i)$. Such such a two round separation has the effect of forcing the players to choose their r_i without having any clue (in a strong information-theoretic sense) about the r_i's chosen by the remaining players. In this way the produced key is uniformly distributed if at least one of the players chooses his contributions uniformly and at random (and we stress that in our model we assume that at least one player mantains a fully honest behavior).

In practice, we implement this idea by assuming the second round starts when each player has received *all* the $n-1$ contributions from the remaining parties. The underlying intuition is that if a player has received $n-1$ ciphertexts, then he can safely start the second round, because he is ensured that every other party has already chosen his own share. Interestingly this approach allows for some modularity in error detection. Indeed, if at least one player aborted after round one (for instance, because he did not correctly receive all the expected ciphertexts) such a situation can be efficiently detected in round two as follows. If after round one some party aborted (i.e. quit) the protocol then the remaining players cannot reconstruct the polynomial $f(\cdot)$ — simply because not enough shares are available — and the protocol can be immediately aborted. On the other hand the fact of receiving all the expected shares in round two, can be seen as a "so far, so good" guarantee: if a player sends his share of the polynomial, it means that he must have been happy with what he has received so far.

Disclosing the polynomial's shares – We notice that in the first round (step 5), a player can safely send his "own" shares $f_i(j)$ (the shares for his private polynomial f_i), without encrypting them, since this does not reveal any information at all about his randomizer $r_i = f_i(0)$: in fact the value $f_i(i)$ is *never* disclosed at any time. Moreover, and for the same reason, it is important to note that the entire transcript of the first round does not reveal anything about the "global" shares $f(i)$ neither. More precisely, recall that the "global" share for player P_i (i.e., his share of $f(\cdot)$) is defined as $f(i) = \sum_{j=1}^{n} f_j(i)$, but only the values $f_{j\neq i}(i)$ are disclosed. Thus, until the second round, step 5, all "global" shares $f(i)$ are still *information-theoretically* hidden to the adversary, and each player P_i knows exactly $f(i)$, that is, no more at all about other $f(j)$'s. During the second round, the shares will be disclosed (keep in mind that is done once the contributions a_i's have been received by each player).

Key confirmatory – There remains one final problem to discuss in our protocol, because of which we yet need a confirmation step at the very end of round two. Actually, to be sure that all parties correctly recover the session key, we use the following technique, known as *key confirmatory,* that allows each player to check that the remaining participants have computed the key. This additional confirmation step makes use of an asynchronous broadcast, which means that we need to assume that the network is equipped with such primitive. Using a

broadcast, either everybody received the confirmation messages, or nobody did and the protocol aborts.

We can obtain the key confirmatory property by having each player computing and broadcasting an additional value to other players. Every player sends a "receipt" which is computed from the session key, thus playing the role of an authenticator. The technique is described with more details in [9] and requires the assumption of random oracle[5] in order for the authenticator not to leak any information about the session key. In particular, such an authenticator, should not be computed directly from the final session key, but rather from an intermediate common secret (otherwise, an eavesdropper would be able to gain some partial information about *sk* — for instance the hash of *sk* — and to distinguish it from a random string).

4 Security of the Scheme

In this section we prove the following security theorem.

Theorem 1. *The protocol presented in figure 1 is a secure Authenticated Group Key Agreement protocol, achieving completeness, contributory and privacy (under the DDH assumption).*

Completeness – Obvious by inspection.

Privacy – We consider a simulator $\mathcal{S}$ that *emulates* the protocol to the adversary in such way that the simulation is indistinguishable from the real attack. Formally the simulator goes as follows. It receives as input a triplet (X, Y, Z), for which it has to decide whether it is a Diffie-Hellman triplet or not. Let $\mathcal{Q}$ be the total number of interactions the adversary is likely to make. $\mathcal{S}$ starts by choosing at random an integer q_0 in $[1, \mathcal{Q}]$, hoping the q_0-th interaction will be the attacked session. Then it chooses uniformly and at random an index i_0 in $[1, n]$. After that, it initializes the protocol by choosing n random exponents ξ_1 through ξ_n. It sets the public key $h_i = Y g^{\xi_i} \bmod p$ for every player P_i. Finally it gives g and all the h_i's to the adversary $\mathcal{A}$, as the public input of the protocol.

To take into account the rushing scenarios, we consider two different pseudo-random generators $\mathcal{R}$ and $\mathcal{R}^\star$, assuming the latter is biased. In particular, $\mathcal{R}^\star$, when called by a player, takes as input all previous data used by this player. We denote, to formalize our simulation, by $\mathcal{R}_j$ the pseudo-random generator used by P_j, and we set $\mathcal{R}_{i_0} = \mathcal{R}$ and $\mathcal{R}_j = \mathcal{R}^\star$ for all $j \neq i_0$.

Then (for each parallel sessions), the simulator $\mathcal{S}$ simulates the players in the first round as follows. On receiving a $\mathsf{Send}(U_j, \mathtt{start})$-query, the simulator chooses a secret contribution a_j and n coefficients $r_j, (b_{j,k})_{1 \leq k \leq n-1}$, using the pseudo-random generator $\mathcal{R}_j$. The ciphertexts in step 4 are computed straightforwardly, except if $j = i_0$ and $q = q_0$. In that later case, the simulator chooses

[5] Actually the random oracle is considered for efficiency reasons only and it is not necessary for the [9] technique to work. In particular the random oracle can be replaced by a *pseudo-random* function [21].

(uniformly) $n-1$ random values ρ_j (for $j=1,\ldots,n$ but $j \neq i_0$) and computes $A_{i_0,j} = Xg^{\rho_j}$ and $B_{i_0,j} = ZY^{\rho_j}X^{\xi_j}g^{\rho_j\xi_j}a_{i_0}$ as an encryption of a_{i_0}. The query is answered with the $n-1$ (signed) flows to be sent to others.

The second round starts (for player U_j) after having received $n-1$ queries, from $n-1$ other players (within a given concurrent session). Before that, the simulator just stores the received flows. The simulator checks the authenticity of the received flows, then defines $a_{(j)}$ as the product of all a_i. Note, in particular, $\mathcal{S}$ does not perform the multiplication of ciphertexts (step 2), nor the decryption (step 3), since it does not know the private key $x_j = \log_g h_j$. Steps 4 and 5 of round 2 are performed straightforwardly, and the query is finally answered by f_j, together with its signature.

Round 3 is simulated as in the real protocol. The confirmation step is processed straightforwardly. After the third Round, a Reveal-query is answered straightforwardly, except if asked to U_{i_0} within the q_0-session. In that case $\mathcal{S}$ aborts.

If the Test -query does not occur within the q_0-th session, the simulator aborts. This happens with probability at most $(Q-1)/Q$. Otherwise, it is processed as follows. Let $\kappa = a \cdot g^r = \prod_{i=1}^n a_i \cdot g^{\sum_{i=1}^n r_i}$. When the Test-query occurs, the simulator flips a private coin β and set $\kappa_0 \leftarrow \mathcal{K}, \kappa_1 = \kappa$, where $\mathcal{K}$ is the session key space (and in our case $\mathcal{K} = Q$). Then it gives κ_β to the adversary. The interaction might continue then; at the end of the attack, the adversary answers with a bit b', that the simulator relays back as its own guess. The theorem then follows immediately from the following two claims.

Claim 1 *If the simulator's input is a Diffie-Hellman triplet (that is $b=1$) the adversary's view is perfectly indistinguishable from the real protocol.*

It is easy to see that, in this case, the simulation is perfectly identical to the real protocol with player P_i using private contribution a_i, and thus the value κ is actually the session key *sk*. This means that an infinitely powerful adversary, which would be able to recover all plaintexts, would necessarily lead to $sk = \kappa$. Indeed, the secret key of player P_j is implicitly $y + \xi_j$, where $y = \log_g Y$. And any ciphertext $C_{i_0,j}$ is an honest encryption of a_{i_0}, using randomness $x + \rho_j$, where $x = \log_g X$. Of course, any other $C_{i,j}$ is an encryption of a_i under public key h_j.

Then we have ($\mathcal{A}_\mathcal{V}$ denotes the adversary together with its view):

$$\begin{aligned}
\Pr[\mathcal{A}_\mathcal{V}(\kappa_\beta)=1|\beta=1 \wedge b=1] &= \Pr\left[\mathcal{A}_\mathcal{V}(\kappa_\beta)=1 \,\middle|\, \beta=1 \wedge \kappa_1 = sk\right] \\
&= \Pr\left[\mathcal{A}_\mathcal{V}(\kappa_1)=1 \,\middle|\, \kappa_1 = sk\right] \qquad (1)\\
\Pr[\mathcal{A}_\mathcal{V}(\kappa_\beta)=1|\beta=0 \wedge b=1] &= \Pr\left[\mathcal{A}_\mathcal{V}(\kappa_\beta)=1 \,\middle|\, \beta=0 \wedge \kappa_0 \leftarrow \mathcal{K}\right] \\
&= \Pr\left[\mathcal{A}_\mathcal{V}(\kappa_0)=1 \,\middle|\, \kappa_0 \leftarrow \mathcal{K}\right] \qquad (2)
\end{aligned}$$

Then using the fact that $\Pr[\beta=1] = \Pr[\beta=0] = 1/2$, we have:

$$\Pr[\mathcal{A}_\mathcal{V}(\kappa_\beta)=1|b=1] = \frac{1}{2}\Pr[\mathcal{A}_\mathcal{V}(\kappa_1)=1|\kappa_1=sk] + \frac{1}{2}\Pr[\mathcal{A}_\mathcal{V}(\kappa_0)=1|\kappa_0 \leftarrow \mathcal{K}]$$

Claim 2 *If the simulator's input is a random triplet (that is $b = 0$) the adversary's view is independent from a_{i_0}.*

In such a case, all the values are correctly computed, except that the ciphertexts $C_{i_0,j}$ encrypt random values. More precisely, the value computationally hidden in $C_{i_0,j}$ under public key $h_j = Yg^{\xi_j}$ is (implicitly):

$$\hat{a}_{i_0,j} = \frac{B_{i_0,j}}{(A_{i_0,j})^{y+\xi_j}} = \frac{ZY^{\rho_j}X^{\xi_j}g^{\rho_j\xi_j}a_{i_0}}{(Xg^{\rho_j})^{y+\xi_j}} = \frac{g^{z+y\rho_j+x\xi_j+\rho_j\xi_j}a_{i_0}}{(g^{x+\rho_j})^{y+\xi_j}} = g^{z-xy}a_{i_0}$$

where $z = \log_g Z$. Note that this value does not depend from the index j of the receiver. This is due to the fact we use the *additive* random self-reducibility property of the Diffie-Hellman problem.

Consequently, the plaintext that an infinitely powerful adversary would recover by decrypting all the ciphertexts is (for any j): $\hat{a}_{(j)} = g^{z-xy}\prod_k a_k = g^{z-xy}a_{(j)}$; thus, the adversary learns no information at all about $a_{(j)}$ when eavesdropping the messages. According to adversary's view, the session key sk associated to this simulated execution of the protocol is thus

$$\hat{a} \cdot g^{\sum_{j=1}^{n} r_j} = g^{z-xy}a \cdot g^r.$$

On the other hand, the simulation makes all players setting their key to $a_{(j)}$. Then, the value "recovered" (according to the simulation) by every player P_i, including P_{i_0}, is $\kappa_1 = ag^r$; moreover a_{i_0} and, thus κ_1, is uniformly distributed over Q, exactly as κ_0 is. Consequently, the value of β is information-theoretically hidden to $\mathcal{A}$.

$$\Pr[\mathcal{A}_{\mathcal{V}}(\kappa_\beta) = 1 | b = 0] = \frac{1}{2}\Pr[\mathcal{A}_{\mathcal{V}}(\kappa_1) = 1 | \kappa_1 \leftarrow \mathcal{K}] + \frac{1}{2}\Pr[\mathcal{A}_{\mathcal{V}}(\kappa_0) = 1 | \kappa_0 \leftarrow \mathcal{K}]$$

By subtraction, we get:

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ddh}}(\mathcal{S}) &= \Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0] \\
&= \Pr[\mathcal{A}_{\mathcal{V}}(\kappa_\beta) = 1 | b = 1] - \Pr[\mathcal{A}_{\mathcal{V}}(\kappa_\beta) = 1 | b = 0] \\
&= \frac{1}{2}\left(\Pr\left[\mathcal{A}_{\mathcal{V}}(\kappa_1) = 1 \middle| \kappa_1 = sk\right] - \Pr\left[\mathcal{A}_{\mathcal{V}}(\kappa_1) = 1 \middle| \kappa_1 \leftarrow \mathcal{K}\right]\right) + \frac{1}{2} \times 0 \\
&= \frac{\mathsf{Adv}(\mathcal{A})}{2}
\end{aligned}$$

Assuming the DDH assumption, this quantity is a negligible amount.

In fact, we have conditioned by the fact the Test-query has been correctly guessed, so we must divide by $1/\mathcal{Q}$.

Contributory – Contributory trivially follows from the fact that every player is forced to choose his share a_i having no information at all (in an information theoretic sense!) about the actual value of the randomizer r.

5 Comments, Optimizations, and Variants

Efficiency of the protocol – Our protocol is very efficient both in terms of bandwidth and in term of number of rounds required. The number of bits sent by each player is bounded by $3|p|n$ plus $n+2$ times the size of the employed signature scheme (used for the authentication). The protocol requires 2 rounds of communication, one asynchronous broadcast for the confirmation step and roughly $2n$ exponentiations per player (plus the cost of computing the signatures). If precomputations are possible (in a context where, for example, the participants public keys are all known in advance), all the exponentiations in Round 1 can be done off-line and the number of total exponentiations (per player) reduces to 2 (plus the cost of the signatures, and the cost of multiplying the received ciphertexts of course). To our knowledge, in this case (and for this specific aspect) our scheme is one of the most efficient group key agreement solutions known. Moreover, being a constant round protocol, it has the property that the number of "slow guys" is not a major efficiency issue. Indeed, a n round protocol is like a token ring network: a player does its work then passes the token to the next one; hence the delays induced by slow parties go cumulating. In our case, everybody works in parallel so we have the same delay whatever the number of slow guys is (more precisely, the delay is essentially that of the slowest guy).

Considering forward-secrecy – The forward-secrecy property [29,17,10] encompasses that the privacy (semantic security) of the session key is not compromised even in case of a further leakage of the long-term El-Gamal key. In other words, if the adversary learns a private key x_i at some time, then the knowledge of x_i, as well as the view of previous session key establishments, does not help him to get information about these previously established session keys. We state informally that our protocol provides forward-secrecy if at most one private key, say x_1, is revealed. Indeed, if an adversary $\mathcal{A}$knows x_1, it can decrypt all ciphertexts sent to P_1, thus learning all contributions $a_2, \ldots, a_n$. However, P_1's contribution, namely a_1, is never encrypted under $h_1 = g^{x_1}$, and then, remains (computationally) hidden to $\mathcal{A}$, and so does the session key. In order to cover larger scenarios, we have to consider forward-secure public-key encryption schemes [14].

Resistance to known-key attacks – A key exchange protocol is said to be resistant to known-key attacks [32] if the exposure of a session key gives no advantage to an adversary for breaking the privacy of future session keys. This property takes some importance in dynamic groups, in which future session keys are computed from private data among which is the current session key. Our protocol trivially provides resistance to such attacks, since all values are one-time used and picked ("fresh") at the beginning of a key exchange.

A General Solution – Up to some loss in efficiency it is possible to generalize our construction in order to obtain a constant round authenticated group key agreement scheme provably secure under the sole assumption that trapdoor functions exist (indeed, this assumption ensures that a semantically secure en-

cryption scheme and a secure signature scheme exist). Details will appear in the final version of this paper.

6 Conclusions

In this paper we presented a new protocol that achieves strong properties of efficiency and security under standard assumptions. The protocol is efficient both in communication rounds and in bandwidth: the number of communication rounds is constant, and the bandwidth is comparable with that of previously proposed schemes. Our scheme is provably secure under the Decisional Diffie-Hellman assumption, and enjoys several additional properties such as forward-secrecy or an increased efficiency when preprocessing is allowed. An intriguing, still open, research problem is to establish a secure key agreement scheme that provides some kind of "resistance" with respect to active adversaries (i.e. for example a protocol that allows to the non corrupted players to eliminate the bad guys and to agree on a key.

Acknowledgments. We wish to thank Jacques Stern for valuable comments on a early version of this paper. We thank David Pointcheval for a number of helpful discussions we had.

References

1. G. Ateniese, M. Steiner and G. Tsudik. New multi-party authentication services and key agreement protocols. *IEEE Selected Areas in Communications,* **18**(4): 628–639, 2000.
2. M. Bellare, R. Canetti and H. Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols. In *STOC '98,* pp. 419–428, 1998.
3. M. Bellare, D. Pointcheval and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In *Eurocrypt '00, LNCS* 1807, pp. 139–155. Springer, 2000.
4. M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *ACM CCS '93,* pp. 62–73. 1993.
5. M. Bellare and P. Rogaway. Entity authentication and key distribution. In *Crypto '93, LNCS* 773, pp. 232–249. Springer, 1993.
6. M. Bellare and P. Rogaway. Provably secure session key distribution: The three party case. In *STOC '95,* pp. 57–66, 1995.
7. S. Blake-Wilson, D. Johnson and A. Menezes. Key agreement protocols and their security analysis. In *Conf. on Cryptography and Coding, LNCS* 1355, pp. 30–45. Springer, 1997.
8. S. Blake-Wilson and A. Menezes. Authenticated Diffie-Hellman key agreement protocols. In *SAC '98, LNCS* 1556, pp. 339-361. Springer, 1998.
9. B. Bresson, O. Chevassut and D. Pointcheval. Provably authenticated group Diffie-Hellman key exchange – the dynamic case. In *Asiacrypt '01, LNCS,* pp. 290–309.
10. E. Bresson, O. Chevassut and D. Pointcheval. Dynamic group Diffie-Hellman key exchange under standard assumptions. In *Eurocrypt '02, LNCS* 2332, pp. 321–336.

11. E. Bresson, O. Chevassut and D. Pointcheval. The group Diffie-Hellman problems. In *SAC '02,* LNCS 2595, pp. 325–338. Springer, 2002.
12. M. Burmester and Y. G. Desmedt. A secure and efficient conference key distribution system. In *Eurocrypt '94, LNCS* 950, pp. 275–286. Springer, 1995.
13. C. Cachin, K. Kursawe, F. Petzold, and V. Shoup. Secure and efficient asynchronous broadcast protocols. In *Crypto '01, LNCS* 2139, pp. 524–541. Springer, 2001.
14. R. Canetti, S. Halevi and J. Katz. A forward-secure public-key encryption scheme. In *Eurocrypt '2003, LNCS* 2656, pp. 255–271, Springer, 2003.
15. R. Cramer V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Crypto '98, LNCS* 1462, pp. 13–25. Springer.
16. W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory,* vol. 22(6): pp. 644–654, November 1976.
17. W. Diffie, P. van Oorschot and W. Wiener. Authentication and authenticated key exchange. In *Designs, Codes and Cryptography,* vol. 2(2), pp. 107–125, June 1992.
18. W. Diffie, D. Steer, L. Strawczynski and M. Wiener. A secure audio teleconference system. In *Crypto '88, LNCS* 403, pp. 520–528. Springer, 1988.
19. R. Dupont and A. Enge. Practical non-interactive key distribution based on pairings. Cryptology eprint archive, `http://eprint.iacr.org`, May 2002.
20. T. ElGamal. A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE IT*-31(4):469–472, 1985.
21. O.Goldreich, S. Goldwasser and S. Micali. How to Construct Random Functions. *JACM* **33**(4):792-807, 1986.
22. S. Goldwasser and S. Micali. Probabilistic Encryption. *JCSS,* 28(2):270–299, April 1984.
23. I. Ingemarsson, D. T. Tang and C. K. Wong. A conference key distribution system. *IEEE Transactions on Information Theory,* vol. IT-28(5): pp. 714–720, September 1982.
24. A. Joux. A one-round protocol for tripartite Diffie-Hellman. In *Proc. of ANTS-4 Conference,* vol. 1838 of *LNCS,* pp. 385–394, Springer-Verlag, 2000.
25. J. Katz and M. Yung. A fully scalable Protocol for Authenticated Group Key Exchange. In *Crypto '03,* to appear.
26. C.-H. Li and J. Pieprzyk. Conference key agreement from secret sharing. In *Proc. of ACISP '99,* vol. 1587 of *LNCS,* pages 64–76. Springer-Verlag, 1999.
27. D. Malkhi, M. Merrit and O. Rodeh. Secure Reliable Multicast Protocols in a WAN. In *Proc. of International Conference on Distributed Computing Systems,* pp. 87–94, 1997.
28. D. Malkhi and M. Reiter. A High-Throughput Secure Reliable Multicast Protocol. *J. of Computer Security,* vol. 5, pp. 113-127, 1997.
29. A. J. Menezes, P. V. Oorschot and S. A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1997. `http://cacr.math.uwaterloo.ca/hac/`.
30. A. Shamir. How to share a secret. In *CACM,* **22**:612–613. November 1979.
31. V. Shoup. On formal models for secure key exchange. Tech. Rep. RZ 3120, IBM Zürich Research Lab, November 1999.
32. M. Steiner, G. Tsudik and M. Waidner. Key agreement in dynamic peer group. *IEEE Transactions on Parallel and Distributed Systems,* vol. 11(8): pp. 769–780, August 2000.

Efficient ID-based Group Key Agreement with Bilinear Maps

Kyu Young Choi, Jung Yeon Hwang, and Dong Hoon Lee

Center for Information Security Technologies (CIST),
Korea University, Seoul, Korea
{young,videmot}@cist.korea.ac.kr
donghlee@korea.ac.kr

Abstract. In modern collaborative and distributed applications, authenticated group key agreement (GKA) is one of important issues. Recently identity (ID)-based authenticated GKA has been increasingly researched because of the simplicity of a public key management. In this paper, we present a formal treatment on ID-based authenticated GKA, which extends the standard GKA model. We present two GKA protocols which use a bilinear-based cryptography: one is a bilinear variant of Burmester and Desmedt protocol [13] and the other is ID-based authenticated protocol based on the former protocol. Our protocols are scalable and 2-round protocols with forward secrecy. In particular, the ID-based authenticated GKA protocol provides a batch verification technique, which verifies the validity of transcripts from other group players simultaneously and improves computational efficiency. We then prove their securities under the decisional bilinear DH and computational DH assumptions.

1 Intoduction

BACKGROUND. In many modern collaborative and distributed applications such as multicast communication, audio-video conference and collaborative tools, scalable and reliable group communication is one of the critical problems. A group key agreement (GKA) protocol allows a group of users to share a key which may later be used to achieve some cryptographic goals. In addition to this basic tool an authentication mechanism provides an assurance of key-sharing with intended users. A protocol achieving these two goals is called an authenticated group key agreement (AGKA) protocol.

Among various authentication flavors, asymmetric techniques such as certificate based PKI (public key infrastructure) or ID-based system are commonly used to provide authentication. In a typical PKI deployed system, a user should obtain a certificate of a long-lived public key from the certifying authority and this certificate be given to a partner to authenticate the user. Whereas in a ID-based system, the partner just has to know the public identity of the user such as e-mail address. Thus, compared to certificate-based PKI system, ID-based authenticated systems simplify the key agreement (management) procedures.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 130–144, 2004.

Several papers have attempted to establish ID-based authenticated key agreement protocol. But the results in [19,22,24] only present informal analysis for the security of the proposed protocols and some of these protocols subsequently found to be flawed [19]. Joux [15] proposed a single round tripartite key agreement using Weil and Tate pairings but unauthenticated. Authenticated versions of this protocol were presented in [1,24]. Unfortunately, Joux's method does not seem possible to be extended to larger groups consisting of more than three parties since the method is based on the *bilinearity* itself. Recently, an ID-based group (n-party) key agreement protocol which uses the one-way function trees and a pairing is firstly proposed by Reddy, et al. [18] with informal security analysis. Barua, et al. [2] proposed an ID-based multi party key agreement scheme which uses ternary trees. The two protocols above have $\mathcal{O}(\lg n)$ communication rounds.

CONTRIBUTION. In this paper we formally present efficient ID-based authenticated group key agreement, which uses a bilinear-based cryptography. The protocol is a contributory key agreement in which generating a group key is the responsibility not only of the group manager, but also of every group member. Hence it does not impose a heavy computational burden on a particular party, which may cause bottle-neck.

To construct our ID-based AGKA protocol, we first present underlying 2-round GKA protocol, which is a bilinear version of the Burmester and Desmedt (BD) protocol [13]. We should be careful of the conversion since the trivial conversion of the BD protocol into a bilinear setting by simply substituting generators does not provide security even against a passive adversary. This security degradation stems from the *gap* property of a certain group where DDH problem is easy but CDH problem hard.

We then make an ID-based authentication method by combining this method and the former GKA protocol. In fact the presented ID-based authentication method can be naturally transformed into a normal ID-based signature scheme. Moreover the method provides a batch verification technique, which verifies the validity of transcripts simultaneously, to greatly improve computational efficiency. Like the underlying GKA protocol, our ID-based AGKA protocol is 2-round. Our ID-based AGKA protocol is most efficient in computational and communicational costs as compared to other previous known ID-based AGKA protocols.

We prove the security of both protocols under the intractability of CDH and DBDH (Decisional Bilinear DH) problems in the *random oracle* model. The protocols achieve *forward secrecy* in the sense that exposure of user's long-lived secret keys does not compromise the security of previous session keys.

RELATED WORK. Since the original two party Diffie-Hellman key agreement protocol has been presented in [14], authenticated key agreement problems have been extensively researched. In particular, Bellare and Rogaway adapted so-called *provable security* to a key exchange and firstly provided formal framework

in two and three party setting [5,6]. Based on that model, many subsequent works have identified concrete cryptographic problems.

Only recently, provably secure solutions for the authenticated group key agreement problem was presented in works of Bresson, et al. [12,10,11], which extended the results in [5,6,4]. Despite of the initial formal step, these protocols, based on the protocols of Steiner, et al. [23], require (relatively) expensive computational cost and the number of round is linear in the number of users participating in a session. Boyd, et al. [8] presented very efficient GKA protocol with a security proof in the random oracle model but did not provide forward secrecy. Katz, et al. [16] presented the constant-round and scalable AGKA protocol with forward secrecy, which is proven secure in the standard model. They took a modular approach and used a signature-based compiler that transforms any GKA protocol secure against a passive adversary to one secure against a stronger active adversary.

ORGANIZATION. Our paper is organized as follows. We define our security model in Section 2. We review cryptographic assumptions needed in Section 3. We present our GKA and ID-based AGKA protocols and prove the security in Section 4. We finally compare our protocol with other ID-based AGKA protocols and conclude in Section 5.

2 The Model

The model described in this section extends one of Bresson, et al. [10] which follows the approach of Bellare and Rogaway [5,6].

In our protocol, we assume broadcast network in which the users can *broacast* messages to others. Our broadcast network will neither provide authenticity nor guarantee that all user receive identical messages. I.e. we allow the possibility that a malicious adversary may read the broadcast messages and substitute some of them.

2.1 Security Model

Participants. We assume that each user U_i has a unique identity ID_i from $\{0,1\}^\ell$ and all identities are distinct. We also assume for simplicity a fixed set of protocol users $\mathcal{U} = \{U_1, \ldots, U_n\}$ where the number of users is polynomial in the security parameter k.

In the model we allow each user $U_i \in \mathcal{U}$ to execute the protocol many times with different users. *Instances* of a user U_i model distinct, but possibly concurrent executions of the protocol. We denote instance s of a user U_i, called an oracle, by Π_i^s for an integer $s \in \mathbb{N}$.

Initialization. During this phase, each user $U \in \mathcal{U}$ gets public and private keys. ID-based GKA protocol requires the following initialization phase.

- The master secret key msk and global parameters params are generated by algorithm Setup : $\mathsf{params} \leftarrow \mathsf{Setup}(1^k, \ell)$ where ℓ is the identity length.
- Each user U_i gains the long term secret key S_i from algorithm Ext : $S_i \leftarrow \mathsf{Ext}_{\mathsf{msk}}(ID_i)$.

The public parameters params and identities $\mathcal{ID} = \{ID_1, \ldots, ID_n\}$ are known by all users (and also by adversary).

Adversarial model. Normally, the security of a protocol is related to the adversary's ability. The abilities are formally modeled by queries issued by adversaries. We assume that a probabilistic polynomial time (PPT) adversary $\mathcal{A}$ controls the communications completely and can make queries to any instance. The list of queries that $\mathcal{A}$ can make is summarized below:

- $\mathsf{Extract}(ID_U)$: This query allows the adversary to get the long-term private key corresponding to ID_U where $ID_U \notin \mathcal{ID}$.
- $\mathsf{Execute}(\mathcal{ID})$: This query models passive attacks, where the adversary eavesdrops an executions of the protocol. $\mathcal{A}$ gets back the complete transcripts of an honest execution between the users in $\mathcal{ID}$. The number of group members are chosen by the adversary.
- $\mathsf{Send}(\Pi_i^s, M)$: This query allows the adversary to make the user ID_i run the protocol normally. This sends message M to instance Π_i^s which returns the reply generated by this instance.
- $\mathsf{Reveal}(\Pi_i^s)$: This query models the adversary's ability to find session group keys. If an oracle Π_i^s has *accepted,* holding a session group key K, then K is returned to the adversary.
- $\mathsf{Corrupt}(ID_i)$: This query models the attacks revealing the long-term secret key S_i. This does not outputs any internal data of ID_i.
- $\mathsf{Test}(\Pi_i^s)$: This query models the semantic security of a session key. This query is allowed only once by the adversary $\mathcal{A}$. A random bit b is chosen; if $b = 1$ then the session key is returned, otherwise a random value is returned.

In the model we consider two types of adversaries according to their attack types. The attack types are simulated by the queries issued by adversaries. A *passive adversary* is allowed to issue Execute, Reveal, Corrupt, and Test queries, while an *active adversary* is additionally allowed to issue Send and Extract queries. Even though Execute query can be simulated using Send queries repeatedly, we use the Execute query for more exact analysis.

2.2 Security Notions

Session IDS and Partnering. Following [16], we defines session IDS and partnering. The session IDS (SIDS) for an oracle Π_i^s is defined as $\mathrm{SIDS}(\Pi_i^s) = \langle SID_{ij} \rangle$, where SID_{ij} is the concatenation of all messages sent and received by an oracle Π_i^s during the execution. The partner ID for an oracle Π_i^s, denoted by $\mathrm{PIDS}(\Pi_i^s)$, is a set of the identities of the users with whom Π_i^s intends to establish a session key. Instances Π_i^s and Π_j^t are *partnered* if and only if $\mathrm{PIDS}(\Pi_i^s) = \mathrm{PIDS}(\Pi_j^t)$

and $\mathrm{SIDS}(\Pi_i^s)=\mathrm{SIDS}(\Pi_j^t)$. The presented notion of parting is simple since all messages are sent to all other users taking part in the protocol. We say that an oracle Π_i^s *accepts* when it has enough information to compute a session key.

Freshness. An oracle Π_i^s is said *fresh* (or hold a *fresh* key K) if:
- Π_i^s has accepted a session key $K \neq \mathrm{NULL}$ and neither Π_i^s nor one of its partners has been asked for a Reveal query,
- No Corrupt query has been asked before a query of the form $\mathsf{Send}(\Pi_i^s,*)$ or $\mathsf{Send}(\Pi_j^t,*)$, where Π_j^t is one of Π_i^s's partners.

Definitions of Security. We define the security of the protocol by following game between the adversary $\mathcal{A}$ and an infinite set of oracles Π_i^s for $ID_i \in \mathcal{ID}$ and $s \in \mathbb{N}$.

1. The long-term keys are assigned to each user through the initialization phase related to the security parameter.
2. Run adversary $\mathcal{A}$ who may issue some queries and get back the answers by the corresponding oracles.
3. At some stage during the execution a Test query is issued by the adversary to a *fresh* oracle. The adversary may continue to make other queries, eventually outputs its guess b' for the bit b involved in the Test query and terminates.

In this game, the advantage of the adversary $\mathcal{A}$ is measured by the ability distinguishing the session group key from a random value, i.e. its ability guessing b. We define Succ to be the event that $\mathcal{A}$ correctly guesses the bit b used by the Test oracle in the answering this query. The advantage of an adversary $\mathcal{A}$ in attacking protocol P is defined as $\mathsf{Adv}_{\mathcal{A},P}(k) = \left|2 \cdot Pr[\mathsf{Succ}] - 1\right|$.

We say that a protocol P is a *secure (ID-based authenticated) group key agreement* scheme if the following two properties are satisfied:

- Correctness: in the presence of a (active) passive adversary partner oracles accept the same key.
- Indistinguishability: for every PPT (active) passive adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A},P}(k)$ is negligible.

Forward Secrecy. In this paper, we are concerned with protocols providing forward secrecy meaning that an adversary gets negligible knowledge information about *previously* established session keys when making a Corrupt query. We define $\mathsf{Adv}_P^{GKA\text{-}fs}(t, q_{ex})$ to be the maximal advantage of any passive adversary attacking P, running in time t and making q_{ex} Execute queries. Similarly, we define $\mathsf{Adv}_P^{AGKA\text{-}fs}(t, q_{ex}, q_s)$ to be the maximal advantage of any active adversary attacking P, running in time t and making q_{ex} Execute queries and q_s Send queries.

Authentication. In this paper, we focus on AGKA with implicit authentication; a key agreement protocol is said to provide implicit key authentication if users are assured that no other users except partners can possibly learn the value of a

particular secret key. Note that the property of implicit key authentication does not necessarily mean that partners have actually obtained the key.

3 The Bilinear Maps and Assumptions

In this section, we review some assumptions related to our protocols. Through the paper, we assume that $\mathbb{G}_1$ is a cyclic additive group of prime order q and $\mathbb{G}_2$ is a cyclic multiplicative group of same order q, and the discrete logarithm problem (DLP) in both $\mathbb{G}_1$ and $\mathbb{G}_2$ are intractable.

CDH Parameter Generator: A CDH parameter generator $\mathcal{IG}_{CDH}$ is a PPT algorithm that takes a security parameter 1^k, runs in polynomial time, and outputs an additive group $\mathbb{G}$ of prime order q.

Computational Diffie-Hellman (CDH) problem in $\mathbb{G}$: Informally speaking, the computational DH problem is to compute abP when given a generator P of $\mathbb{G}$ and aP, bP for some $a, b \in \mathbb{Z}_q^*$. More formally, the advantage of $\mathcal{A}$ with respect to $\mathcal{IG}_{BDH}$ is defined to be

$$\Pr\left[\mathcal{A}(\mathbb{G}, P, aP, bP) = abP \;\middle|\; \mathbb{G} \leftarrow \mathcal{IG}_{CDH}(1^k); P \leftarrow \mathbb{G}; a, b \leftarrow \mathbb{Z}_q^*\right].$$

$\mathcal{IG}_{CDH}$ is said to satisfy the CDH assumption if any PPT $\mathcal{A}$ has negligible advantage in solving CDH problem.

Admissible Bilinear Map. We call $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ an *admissible bilinear* map if it saifsfies the following properties:

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

BDH Parameter Generator: A BDH parameter generator $\mathcal{IG}_{BDH}$ is a probabilistic polynomial time (PPT) algorithm that takes a security parameter 1^k, runs in polynomial time, and outputs the description of two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of the same order q and an admissible bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$.

Decisional Bilinear Diffie-Hellman (DBDH) problem in $[\mathbb{G}_1, \mathbb{G}_2, e]$: Informally speaking, the decisional BDH problem is to distinguish between tuples of the form $(P, aP, bP, cP, e(P, P)^{abc})$ and $(P, aP, bP, cP, e(P, P)^d)$ for random $P \in \mathbb{G}_1$, and $a, b, c, d \in \mathbb{Z}_q^*$. More formally, the advantage of $\mathcal{A}$ with respect to $\mathcal{IG}_{BDH}$ is defined to be

$$\left| \Pr\left[\mathcal{A}\left(\mathbb{G}_1, \mathbb{G}_2, e, P, aP, bP, cP, e(P, P)^{abc}\right) = 1 \;\middle|\; \begin{array}{c}(\mathbb{G}_1, \mathbb{G}_2, e) \leftarrow \mathcal{IG}_{BDH}(1^k); \\ P \leftarrow \mathbb{G}_1; a, b, c \leftarrow \mathbb{Z}_q^*\end{array}\right] \right.$$
$$\left. - \Pr\left[\mathcal{A}\left(\mathbb{G}_1, \mathbb{G}_2, e, P, aP, bP, cP, e(P, P)^d\right) = 1 \;\middle|\; \begin{array}{c}(\mathbb{G}_1, \mathbb{G}_2, e) \leftarrow \mathcal{IG}_{BDH}(1^k); \\ P \leftarrow \mathbb{G}_1; a, b, c, d \leftarrow \mathbb{Z}_q^*\end{array}\right] \right|.$$

$\mathcal{IG}_{BDH}$ is said to satisfy the DBDH assumption if any PPT $\mathcal{A}$ has negligible advantage in solving DBDH problem.

As noted in [7], BDH parameter generators satisfying the DBDH assumption is believed to be constructed from Weil and Tate pairings associated with supersingular eliptic curves or Abelian varieties.

4 Our GKA and ID-based AGKA Protocol

4.1 GKA Protocol Using a Bilinear Map

We now describe a 2-round GKA protocol using bilinear maps. We denote this protocol by B-GKA. In fact, this protocol is a bilinear variant of the protocol by Burmester and Desmedt. In this protocol, no long-term public/private keys are required. In the following description groups $\mathbb{G}_1$, $\mathbb{G}_2$ and a bilinear map e are generated by a BDH generator in Section 3 and P is a random generator of $\mathbb{G}_1$. When n users $U_1,..,U_n$ want to establish a session key, they proceed as follows :

[Round 1] Each user U_i picks a random integer $a_i \in \mathbb{Z}_q^*$ and computes $P_i = a_i P$. Then each U_i broadcasts P_i to all others and keeps a_i secret.

[Round 2] Upon receipt of P_{i-1}, P_{i+1} and P_{i+2}, each users U_i computes

$$D_i = e\big(a_i(P_{i+2} - P_{i-1}),\ P_{i+1}\big)$$

and broadcasts D_i to all others.

Key Computation. Each U_i computes the session key as follows :

$$K_i = e(a_i P_{i-1}, P_{i+1})^n D_i^{n-1} D_{i+1}^{n-2} \cdots D_{i-2}.$$

It is obvious that all honest users compute the same key as follows:

$$K = e(P, P)^{a_1 a_2 a_3 + \cdots + a_{n-1} a_n a_1 + a_n a_1 a_2}.$$

We note that the trivial conversion of the BD protocol to a bilinear setting by simply substituting generators does not provide security even against a passive adversary. This is possible because of the gap property of $\mathbb{G}_1$ where DDH problem is easy but CDH problem hard.

Theorem 1. *The protocol* B-GKA *is a secure GKA protocol providing forward secrecy under the DBDH assumption. Concretely,*

$$\mathrm{Adv}^{\mathrm{GKA-fs}}_{\text{B-GKA}}(t, q_{ex}) \leq 4 \cdot \mathrm{Adv}^{\mathrm{DBDH}}_{\mathbb{G}_1, \mathbb{G}_2, e}(t).$$

We can prove Theorem 1 in two steps by using standard hybrid argument and showing information theoretical independence of a secret key. The security analysis is similar to that of Katz, et al. [17]. For space limitation we omit the proof. However a tighter security reduction can be obtained using *random self-reducibility* properties of the DBDH problem. The method of the reduction in [17,21] is similarly applied to our reduction.

4.2 ID-based Authenticated Group Key Agreement Protocol

In this section we present an ID-based AGKA protocol based on the previous protocol B-GKA. We denote this protocol by ID-GKA. The protocol involves the trusted key generation center (KGC). In the following description $H : \{0,1\}^* \longrightarrow \mathbb{Z}_q$ and $H_1 : \{0,1\}^* \longrightarrow \mathbb{G}_1$ are cryptographic hash functions. H and H_1 are considered as random oracles in the security proof.

Setup. KGC runs BDH parameter generator, and chooses a random $s \in \mathbb{Z}_q^*$ and a generator P of $\mathbb{G}_1$ and computes $P_{pub} = sP$. Then KGC keeps s secret as the *master secret key* and publishes system parameters params $= \{e, \mathbb{G}_1, \mathbb{G}_2, q, P, P_{pub}, H, H_1\}$.

Extract. When a user with identity ID wishes to obtain a key pair, KGC computes $Q_{ID} = H_1(ID)$ and the private key $S_{ID} = sQ_{ID}$, and returns S_{ID} to the user.

Let $\{U_1, ..., U_n\}$ be a set of users who want to establish a session key and ID_i be the identity of each U_i. The indices are subject to modulo n. U_i's long-term public and private key pair is $\langle ID_i,\ S_i = sQ_i\rangle$.

[Round 1] Each user U_i picks a random integer $a_i \in \mathbb{Z}_q^*$ and computes $P_i = a_iP$, $h_i = H(P_i)$ and $T_i = a_iP_{pub} + h_iS_i$. Each U_i broadcasts $\langle P_i, T_i\rangle$ to all others and keeps a_i secret.

[Round 2] Upon the receipt of $\langle P_{i-1}, T_{i-1}\rangle$, $\langle P_{i+1}, T_{i+1}\rangle$ and $\langle P_{i+2}, T_{i+2}\rangle$, each user U_i checks if the following equation holds:

$$e\Big(\textstyle\sum_{k\in\{-1,1,2\}} T_{i+k}, P\Big) = e\Big(\sum_{k\in\{-1,1,2\}}(P_{i+k} + h_{i+k}Q_{i+k}), P_{pub}\Big)$$

If the above equation is satisfied, then U_i computes
$D_i = e\big(a_i(P_{i+2} - P_{i-1}), P_{i+1}\big)$
and broadcasts D_i to all others. Otherwise U_i stops.

Key Computation. Each U_i computes the session key,

$$K_i = e(a_iP_{i-1}, P_{i+1})^n D_i^{n-1} D_{i+1}^{n-2} \cdots D_{i-2}.$$

The correctness of key computation is same to that of the protocol B-GKA.
In the above protocol, we used an authentication scheme Γ defined as follows;

Generation. Given a secret key S_{ID}=$sH_1(ID)$, compute $T = aP_{pub} + hS_{ID}$ where $a \in_R \mathbb{Z}_q^*$ and $h = H(aP)$; $\langle aP, T\rangle \leftarrow \Gamma_{gen}(S_{ID})$.

Verification. Given a public Q_{ID} and $\langle aP, T\rangle$, verify that $e(T, P) = e(aP + hQ_{ID}, P_{pub})$, where $h = H(aP)$; `True` or `False` $\leftarrow \Gamma_{ver}(Q_{ID}, \langle aP, T\rangle)$.

The correctness of Γ is easily proved as follows; for given public Q_{ID} and $\langle aP, T\rangle$, $e(T, P) = e(asP + hsQ_{ID}, P) = e(aP + hQ_{ID}, P_{pub})$ where $h = H(aP)$.

In fact, in Round 2, each user uses a *screening* test [3] to verify the validity of authentication for computational efficiency. This test provides a weaker notion determining if each user has at some point generated the transcript for

authentication rather than checking the given data is a valid transcript for authentication. This validation notion is adequate for our goal since each user wants to do implicit authentication for a session rather than to have an authentication data. However we can directly adapt a batch technique providing a strong notion, such like random subset test and small exponent test, etc., as in [3,9].

We note that the authentication scheme Γ can be easily transformed into an ID-based signature scheme.

For the following security analysis we define $Forger_{\Gamma}$ as a PPT forger of the authentication scheme Γ under the adaptively chosen ID attack and $Forger_{\Gamma}^{ID}$ a PPT forger of Γ under given ID attack.

Theorem 2. *Suppose the hash functions H, H_1 are random oracles. Then the protocol* ID-GKA *is a secure AGKA protocol providing forward secrecy under the DBDH assumption and the CDH assumption. Concretely,*

$$\mathrm{Adv}_{\text{ID-GKA}}^{\mathrm{AGKA-fs}}(t, q_{ex}, q_s) \leq 2nq_{ex} \cdot \mathrm{Adv}_{\mathbb{G}_1,\mathbb{G}_2,\mathrm{e}}^{\mathrm{DBDH}}(t) + \mathrm{Adv}_{\Gamma}^{\mathrm{Forge}}(t).$$

where $\mathrm{Adv}_{\Gamma}^{\mathrm{Forge}}(t)$ *is the maximum advantage of any $Forger_{\Gamma}$ running in time t.*

Proof. Let $\mathcal{A}$ be an active adversary that gets advantage in attacking ID-GKA. The adversary $\mathcal{A}$ can get an advantage by forging authentication transcripts, namely impersonating a user or 'breaking' the protocol without altering transcripts.

First we assume that $\mathcal{A}$ breaks ID-GKA by using adaptive impersonation ability. Using $\mathcal{A}$, we can construct a $Forger_{\Gamma}$ $\mathcal{C}$ that generates a valid message pair $\langle ID, aP, T\rangle$ with respect to Γ as follows: a $Forger_{\Gamma}$ $\mathcal{C}$ honestly generates all other public and private keys for the system. $\mathcal{C}$ simulates the oracle queries of $\mathcal{A}$ in the natural way; this results in a perfect simulation unless $\mathcal{A}$ queries Corrupt(ID). If this occurs, $\mathcal{C}$ simply aborts. Otherwise, if $\mathcal{A}$ generates a new and valid message pair $\langle ID, aP, T\rangle$, this event is denoted by Forge, then $\mathcal{C}$ generates the message pair $\langle ID, aP, T\rangle$. The success probability of $\mathcal{C}$ satisfies $\Pr_{\mathcal{A}}[\mathsf{Forge}] \leq \mathrm{Adv}_{\mathcal{C},\Gamma}^{\mathrm{Forge}}(t) \leq \mathrm{Adv}_{\Gamma}^{\mathrm{Forge}}(t)$.

Next we assume that $\mathcal{A}$ breaks ID-GKA without altering transcripts. Before describing the details we define the Modified DBDH(MDBDH) problem related to our security reduction. The MDBDH problem in $[\mathbb{G}_1, \mathbb{G}_2, e]$ is to distinguish between tuples of the form $(P, aP, bP, cP, sP, saP, sbP, scP, e(P,P)^{abc})$ and $(P, aP, bP, cP, sP, saP, sbP, scP, e(P,P)^{d})$ for random $P \in \mathbb{G}_1$, and $a, b, c, d, s \in \mathbb{Z}_q^*$. It is easily showed that the DBDH problem and the MDBDH problem in $[\mathbb{G}_1, \mathbb{G}_2, e]$ are computationally equivalent. Namely, $\mathrm{Adv}_{\mathbb{G}_1,\mathbb{G}_2,\mathrm{e}}^{\mathrm{DBDH}}(t) = \mathrm{Adv}_{\mathbb{G}_1,\mathbb{G}_2,\mathrm{e}}^{\mathrm{MDBDH}}(t)$.

We first consider the case that an adversary $\mathcal{A}$ makes only a single Execute query $\mathsf{Execute}(ID_1, ..., ID_n)$ and then extend this to the case that $\mathcal{A}$ makes multiple Execute queries. Let n be the number of users chosen by the adversary $\mathcal{A}$. The distribution of the transcript $\mathcal{T}$ and the resulting group key K is given by:

$$\mathsf{params} = \left[\begin{array}{c} (\mathbb{G}_1, \mathbb{G}_2, e) \leftarrow \mathcal{IG}_{BDH}(1^k);\ P \leftarrow \mathbb{G}_1;\ s \leftarrow \mathbb{Z}_q^*;\ P_{pub} = sP \\ Q_1, ..., Q_n \leftarrow \mathbb{G}_1;\ S_1 = sQ_1, ..., S_n = sQ_n : (\mathbb{G}_1, \mathbb{G}_2, e, P, P_{pub}) \end{array} \right]$$

$$Real = \left[\begin{array}{c} a_1, ..., a_n, h_1, ..., h_n \leftarrow \mathbb{Z}_q^*;\ P_1 = a_1P, .., P_n = a_nP; \\ T_1 = a_1P_{pub} + h_1S_1, ..., T_n = a_nP_{pub} + h_nS_n; \\ D_1 = \frac{e(a_1P_2, P_3)}{e(a_1P_n, P_2)}, D_2 = \frac{e(a_2P_3, P_4)}{e(a_2P_1, P_3)}, ..., D_n = \frac{e(a_nP_1, P_2)}{e(a_nP_{n-1}, P_1)}; \\ \mathcal{T} = \langle P_1, ..., P_n, T_1, ..., T_n, D_1, ..., D_n\rangle; \\ K = e(a_1P_n, P_2)^n D_1^{n-1} \cdots D_{n-1} : (\mathcal{T}, K) \end{array} \right]$$

Consider the distributions $Fake_i$ defined as follows:

$$Fake_1 = \left[\begin{array}{c} r_{n,1,2}, a_1, ..., a_n, h_1, ..., h_n \leftarrow \mathbb{Z}_q^*;\ P_1 = a_1 P, .., P_n = a_n P; \\ T_1 = a_1 P_{pub} + h_1 S_1, ..., T_n = a_n P_{pub} + h_n S_n; \\ D_1 = \frac{e(a_1 P_2, P_3)}{e(r_{n,1,2} P, P)}, D_2 = \frac{e(a_2 P_3, P_4)}{e(a_2 P_1, P_3)}, ..., D_n = \frac{e(r_{n,1,2} P, P)}{e(a_n P_{n-1}, P_1)}; \\ \mathcal{T} = \langle P_1, ..., P_n, T_1, ..., T_n, D_1, ..., D_n \rangle; \\ K = e(r_{n,1,2} P, P)^n D_1^{n-1} \cdots D_{n-1} : (\mathcal{T}, K) \end{array}\right] \cdots$$

Continuing in this way, we obtain the distribution:

$$Fake_n = \left[\begin{array}{c} r_{n,1,2}, ..., r_{n-1,n,1}, a_1, ..., a_n, h_1, ..., h_n \leftarrow \mathbb{Z}_q^*;\ P_1 = a_1 P, .., P_n = a_n P; \\ T_1 = a_1 P_{pub} + h_1 S_1, ..., T_n = a_n P_{pub} + h_n S_n; \\ D_1 = \frac{e(r_{1,2,3} P, P)}{e(r_{n,1,2} P, P)}, D_2 = \frac{e(r_{2,3,4} P, P)}{e(r_{1,2,3} P, P)}, ..., D_n = \frac{e(r_{n,1,2} P, P)}{e(r_{n-1,n,1} P, P)}; \\ \mathcal{T} = \langle P_1, ..., P_n, T_1, ..., T_n, D_1, ..., D_n \rangle; \\ K = e(r_{n,1,2} P, P)^n D_1^{n-1} \cdots D_{n-1} : (\mathcal{T}, K) \end{array}\right]$$

$\mathcal{A}$ can compute all $a_i P_{pub} = T_i - hS_i$ from the transcripts since $\mathcal{A}$ can obtain all secret keys S_i and hash values h_i $(i = 1, ..., n)$ by using multiple Corrupt and H queries, respectively. Therefore the distribution of previous transcripts is changed by the distribution related to the modified DBDH problem. Let $\varepsilon(t) = \mathrm{Adv}^{\mathrm{MDBDH}}_{\mathbb{G}_1, \mathbb{G}_2, e}(t)$. A standard argument shows that for any algorithm $\mathcal{A}$ running in time t we have:

$$\left|Pr[\mathcal{T} \leftarrow Real; K \leftarrow Real: \mathcal{A}(\mathcal{T}, K) = 1] - Pr[\mathcal{T} \leftarrow Fake_1; K \leftarrow Fake_1: \mathcal{A}(\mathcal{T}, K) = 1]\right| \leq \varepsilon(t)$$
$$\left|Pr[\mathcal{T} \leftarrow Fake_1; K \leftarrow Fake_1: \mathcal{A}(\mathcal{T}, K) = 1] - Pr[\mathcal{T} \leftarrow Fake_2; K \leftarrow Fake_2: \mathcal{A}(\mathcal{T}, K) = 1]\right| \leq \varepsilon(t)$$
$$\vdots$$
$$\left|Pr[\mathcal{T} \leftarrow Fake_{n-1}; K \leftarrow Fake_{n-1}: \mathcal{A}(\mathcal{T}, K) = 1] - Pr[\mathcal{T} \leftarrow Fake_n; K \leftarrow Fake_n: \mathcal{A}(\mathcal{T}, K) = 1]\right| \leq \varepsilon(t).$$

Let $e(P, P) = g$ in $\mathbb{G}_2$. In experiment Fake_n, the values $r_{1,2,3}, ..., r_{n,1,2}$ are constrained by $\mathcal{T}$ according to the following n equations $\log_g D_1 = r_{1,2,3} - r_{n,1,2}$, $\log_g D_2 = r_{2,3,4} - r_{1,2,3}$,...,$\log_g D_n = r_{n,1,2} - r_{n-1,n,1}$ of which only $n - 1$ of these are linearly independent. Furthermore, K may be expressed as $K = e(P, P)^{a_n a_1 a_2 + \cdots + a_{n-1} a_n a_1}$; equivalently, we have

$$\log_g K = r_{1,2,3} + r_{2,3,4} + \cdots + r_{n,1,2}.$$

Since this final equation is linearly independent from the set of equations above, the value of K is independent of $\mathcal{T}$. This implies that, for any adversary $\mathcal{A}$:

$$\left|Pr[\mathcal{T} \leftarrow Fake_n; K \leftarrow Fake_n: \mathcal{A}(\mathcal{T}, K) = 1] = Pr[\mathcal{T} \leftarrow Fake_n; K \leftarrow Random: \mathcal{A}(\mathcal{T}, K) = 1]\right|.$$

Similarly, a standard argument shows that for any algorithm $\mathcal{A}$ running in time t we have:

$$\left|Pr[\mathcal{T} \leftarrow Fake_n; K \leftarrow Fake_n: \mathcal{A}(\mathcal{T}, K) = 1] - Pr[\mathcal{T} \leftarrow Fake_{n-1}; K \leftarrow Random: \mathcal{A}(\mathcal{T}, K) = 1]\right| \leq \varepsilon(t)$$
$$\vdots$$
$$\left|Pr[\mathcal{T} \leftarrow Fake_1; K \leftarrow Random: \mathcal{A}(\mathcal{T}, K) = 1] - Pr[\mathcal{T} \leftarrow Real; K \leftarrow Random: \mathcal{A}(\mathcal{T}, K) = 1]\right| \leq \varepsilon(t)$$

Eventually, we obtain the equation as follow:

$$\left|Pr[\mathcal{T} \leftarrow Real; K \leftarrow Real: \mathcal{A}(\mathcal{T}, K) = 1] - Pr[\mathcal{T} \leftarrow Real; K \leftarrow Random: \mathcal{A}(\mathcal{T}, K) = 1]\right| \leq 2n\varepsilon(t)$$

Since $\epsilon(t) = \mathrm{Adv}^{\mathrm{MDBDH}}_{\mathbb{G}_1, \mathbb{G}_2, e}(t) = \mathrm{Adv}^{\mathrm{DBDH}}_{\mathbb{G}_1, \mathbb{G}_2, e}(t)$, we have the result that the advantage of $\mathcal{A}$ conditioned on the event ~ Forge is bounded by $2n \cdot \mathrm{Adv}^{\mathrm{DBDH}}_{\mathbb{G}_1, \mathbb{G}_2, e}(t)$. Hence we have

$$\mathrm{Adv}^{\mathrm{AGKA-fs}}_{\mathit{ID\text{-}GKA}}(t, 1, q_s) \leq 2n \cdot \mathrm{Adv}^{\mathrm{DBDH}}_{\mathbb{G}_1,\mathbb{G}_2,\mathrm{e}}(t) + \mathrm{Adv}^{\mathrm{Forge}}_{\Gamma}(t).$$

Finally we have the desired result by adapting a standard hybrid argument that

$$\mathrm{Adv}^{\mathrm{AGKA-fs}}_{\mathit{ID\text{-}GKA}}(t, q_{ex}, q_s) \leq 2nq_{ex} \cdot \mathrm{Adv}^{\mathrm{DBDH}}_{\mathbb{G}_1,\mathbb{G}_2,\mathrm{e}}(t) + \mathrm{Adv}^{\mathrm{Forge}}_{\Gamma}(t).$$

We next show that the authentication scheme Γ is secure against existential forgery on adaptively chosen *ID* attack.

Lemma 1. *Let the hash function H_1 be random oracle. Suppose there exists a $Forger_\Gamma$ $\mathcal{A}$ for an adaptively chosen ID with running time t_0 and advantage ε_0. Suppose $\mathcal{A}$ makes at most q_{H_1} queries to the hash function H_1. Then a $Forger^{ID}_\Gamma$ $\mathcal{B}$ for a given ID with running time $t_1 \leq t_0$ has advantage $\varepsilon_1 \leq \varepsilon_0(1 - \frac{1}{q})/q_{H_1}$.*

Lemma 2. *Let the hash function H, H_1 be random oracles. Suppose there exists a $Forger^{ID}_\Gamma$ $\mathcal{A}$ for a given ID with running time t_1 and advantage $\varepsilon_1 \geq 10(q_s + 1)(q_s + q_H)/q$. Suppose $\mathcal{A}$ makes at most q_H, q_{H_1}, q_s and q_{ex} queries to the H, H_1, Send and Extract respectively. Then there exists an attacker $\mathcal{B}$ that can solve the CDH problem within expected time $t_2 \leq 120686 q_H t_1/\varepsilon_1$.*

The proofs of the above two lemmas are given in Appendix. Combining the Lemma 1 and 2, we obtain that $\mathrm{Adv}^{\mathrm{Forge}}_{\Gamma}(t)$ is negligible in the following theorem. Therefore we can show that our ID-GKA is a secure AGKA providing forward secrecy.

Theorem 3. *Let the hash functions H, H_1 be random oracles. Suppose there exists a $Forger_\Gamma$ $\mathcal{A}$ for an adaptively chosen ID with running time t_0 and advantage $\varepsilon_0 \geq 10 q_{H_1}(q_s + 1)(q_s + q_H)/(q - 1)$. Suppose $\mathcal{A}$ makes at most q_H, q_{H_1}, q_s and q_{ex} queries to the H, H_1, Send and Extract respectively. Then there exists an attacker $\mathcal{B}$ that can solve the CDH problem within expected time $t_2 \leq 120686 q_H t_0/\varepsilon_0$.*

5 Comparison and Conclusion

We now compare our protocol ID-GKA with other previously known ID-based GKA protocols, the binary tree based 2T-IDAGKA [18] and the ternary tree based 3T-IDAGKA [2] in Table 1. We use notations as follows:

- *Round:* The total number of rounds.
- *Message:* The total number of messages sent by users.
- *Computation:* The total number of scalar multiplications.
- *Pairing:* The total number of pairing-computations.

Because the number of users is relatively small in practice, we can assume that, in our ID-GKA, the key computation step requires just one scalar multiplication.

Protocol	Round	Message	Computation	Pairing
2T-IDAGKA [18]	$\mathcal{O}(\lg n)$	$\mathcal{O}(n \lg n)$	$\mathcal{O}(n \lg n)$	$\mathcal{O}(n \lg n)$
3T-IDAGKA [2]	$\mathcal{O}(\lg n)$	$\mathcal{O}(n)$	$\mathcal{O}(n)$	$\mathcal{O}(n \lg n)$
Our ID-GKA protocol	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n)$	$\mathcal{O}(n)$

Table 1. Comparison of ID-based AGKA protocols

As we shown in Table 1, our protocol is the most efficient one as compared to other protocols. In particular, our protocol require $\mathcal{O}(1)$ round and only $\mathcal{O}(n)$ pairing-computations.

In this paper, we have presented a 2-round and scalable ID-based AGKA protocol based on a bilinear variant of the BD protocol [13]. Moreover, we have adapted batch verification technique verifying the validity of transcripts simultaneously, which greatly improves the computational efficiency. We have proved the security of both protocols under the intractability of CDH and DBDH.

References

1. S. Al-Riyami and K.G. Paterson, *Tripartite Authenticated Key Agreement Protocols from Pairings,* Cryptology ePrint Archive, Report 2002/035, 2002. http://eprint.iacr.org/.
2. R. Barua, R. Dutta and P. Sarker, *Extending Joux's Protocol to Multi Party Key Agreement.,* Proc. of Indocrypt '03.
3. M. Bellare, J. A. Garay and T. Rabin, *Fast Batch Verification for modular Exponentiation and Digital Signatures,* Proc. of Eurocrypt '98, LNCS 1403, pp.236-250, Springer-Verlag, 1998.
4. M. Bellare, D. Pointcheaval, and P. Rogaway, *Authenticated key exchange secure against dictionary attacks.,* Proc. of Eurocrypt '00, LNCS 1807, pp.139-155, Springer-Verlag, 2000.
5. M. Bellare, P. Rogaway, *Entity authentication and key distribution,* Proc. of Crypto '93, pp.232-249.
6. M. Bellare, P. Rogaway, *Provably-Secure Session Key Distribution : The Three Party Case,* Proc. of STOC '95, pp. 57-66.
7. D. Boneh and M. Franklin, *Identity-based encryption from the Weil pairing,* Proc. of Crypto '01, LNCS 2139, pp.213-229, Springer-Verlag, 2001.
8. C. Boyd and J.M.G. Nieto, *Round-Optimal Contributory Conference Key Agreement,* Proc. of PKC '03, LNCS 2567, pp.161-174. Springer-Verlag, 2003.
9. C. Boyd and C. Pavlovski, *Attacking and Repairing Batch Verification Schemes,* Proc. of Asiacrypt '00, LNCS 1976, pp.58-71, Springer-Verlag, 2000.
10. E. Bresson, O. Chevassut and D. Pointcheval, *Provably Authenticated Group Diffie-Hellman Key Exchange-The Dynamic Case,* Proc. of Asiacrypt '02, LNCS 2248, pp.290-309, Springer-Verlag, 2001.
11. E. Bresson, O. Chevassut and D. Pointcheval, *Dynamic Group Diffie-Hellman Key Exchange under Standard Assumption (Full version),* Proc. of Eurocrypt '02, LNCS 2332, pp.321-336. Springer-Verlag, 2002.
12. E. Bresson, O. Chevassut, D. Pointcheval. J.-J. Quisquater, *Provably Authenticated Group Diffie-Hellman Key Exchange,* In Proc. of 8th ACM CCCS, pp.255-264, 2001.

13. M. Burmester and Y. Desmedt, *A Secure and Efficient Conference Key Distribution System,* Proc. of Eurocrypt '94, pp.267-275, LNCS 950, Springer-Verlag, 1994.
14. W. Diffie and M. Hellman, *New Directions in Cryptography,* IEEE Transactions on Information Theory 22(6), pp.644-654, 1976.
15. Joux, *A one round protocol for tripartite Diffie-Hellman,* In W. Bosma, editor, Proceedings of Algorithmic Number Theory Symposium. ANTS IV, LNCS 1838, pp.385-394, Springer-Verlag, 2000.
16. J. Katz and M. Yung, *Scalable Protocols for Authenticated Group Key Exchange,* Proc. of Crypto 2003, LNCS 2729, pp.110-125, Springer-Verlag, 2003.
17. J. Katz and M. Yung, *Scalable Protocols for Authenticated Group Key Exchange,* full version.
18. D.Nalla and K. C. Reddy, *Identity Based Authenticated Group Key Agreement Protocol,* Proc. of Indocrypt '02, LNCS 2551, pp.215-233, Springer-Verlag, 2002.
19. D. Nalla, K. C. Reddy, *ID-based tripartite Authenticated Key Agreement Protocols from pairings,* Cryptology ePrint Archive, Report 2003/004. http://eprint.iacr.org/2003/004/.
20. D. Pointcheval and J. Stern, *Security arguments for digital signatures and blind signatures,* J. of Cryptology, Vol. 13, pp.361-396, 2000.
21. V. Shoup, *On formal models for secure key exchange. ,* In ACM Security '99.
22. N.P.Smart, *An Identity based authenticated Key Agreement protocol based on the Weil pairing,* Cryptology ePrint Archive, Report 2001/111,2001. http://eprint.iacr.org/.
23. M. Steiner, G. Tsudik and M. Waidner, *Key Agreement in Dynamic Peer Groups,* IEEE Trans, on Parallel and Distributed Systems 11(8), pp.769-780, 2000.
24. F. Zhang, S. Liu and K. Kim, *ID-based One Round Authenticated Tripartite Key Agreement Protocols with Pairings,* Cryptology ePrint Archive 2002. http://eprint.iacr.org/.

A Proof of Lemma 1

$\mathcal{B}$ is given ID^*. Without any loss of generality, we assume that for any *ID*, $\mathcal{A}$ makes H_1, Send and Extract queries at most once, and Send and Extract queries for public key are preceded by H_1 hash query.

To respond to these queries $\mathcal{B}$ maintains a list L_{H_1} of $\langle ID_i, Q_i \rangle$. The list is initially empty. First, $\mathcal{B}$ chooses $\alpha \in \{1, ..., q_{H_1}\}$ randomly. $\mathcal{B}$ interacts with $\mathcal{A}$ as follows:

- When $\mathcal{A}$ makes the α-th H_1 query on *ID*, $\mathcal{B}$ issues a H_1 query for ID^* and returns the result Q^* to $\mathcal{A}$. Then $\mathcal{B}$ adds $\langle ID, Q^* \rangle$ to L_{H_1}. Otherwise, $\mathcal{B}$ issues a H_1 query for *ID* and returns the result to $\mathcal{A}$. Then $\mathcal{B}$ inserts $\langle ID, Q \rangle$ into L_{H_1}.
- When $\mathcal{A}$ issues an Extract query on Q_i, if $Q_i = Q^*$, then $\mathcal{B}$ outputs FAIL and aborts. Otherwise, $\mathcal{B}$ issues an Extract query for Q_i and returns the result S_i to $\mathcal{A}$.
- When $\mathcal{A}$ issues a H query on $a_i P$, $\mathcal{B}$ issues a H query for aP and returns the result $H(a_i P)$ to $\mathcal{A}$.

- When $\mathcal{A}$ issues a Send query on ID_i, $\mathcal{B}$ issues a Send query for ID_i and returns the result $\langle ID_i, a_iP, T_i\rangle$ to $\mathcal{A}$.

Eventually, $\mathcal{A}$ outputs $\langle ID', a'P, T'\rangle$. Then $\mathcal{B}$ finds the tuple of the form $\langle ID', Q'\rangle$ in L_{H_1}. If $Q' = Q^*$ then $\mathcal{B}$ outputs $\langle ID^*, a'P, T'\rangle$. Otherwise, $\mathcal{B}$ outputs FAIL and aborts.

To complete the proof of Lemma 1 it remains to calculate the probability that algorithm $\mathcal{B}$ aborts during the simulation. Notice that, if $Q' \neq Q^*$ and a pair $\langle ID', Q'\rangle$ is not found in L_{H_1}, then the output $\langle ID', a'P, T'\rangle$ is independent of the knowledge $\mathcal{A}$ accumulated from its various queries. This means that $\mathcal{A}$ succeeds in this case with probability $1/q$. Therefore, the probability that $\mathcal{B}$ does not abort during the simulation is $\frac{1}{q_{H_1}}(1 - \frac{1}{q})$.

B Proof of Lemma 2

First, a BDH parameter generator is run and $\langle e, \mathbb{G}_1, \mathbb{G}_2\rangle$ is outputted. Then $\mathcal{B}$ receives a CDH instance (P, xP, yP) for randomly chosen $x, y \in \mathbb{Z}_q^*$ and $P \in \mathbb{G}_1$. Its goal is to compute xyP.

$\mathcal{B}$ runs a $Forger_\Gamma^{ID}$ $\mathcal{A}$ as a subroutine and simulates its attack environment. $\mathcal{B}$ sets the public system parameters params=$\langle e, \mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, ID^*, H, H_1\rangle$ by letting $P_{pub} = xP$ where x is the master secret key, which is unknown to $\mathcal{B}$ and selecting an identity ID^* for given ID attack of $\mathcal{A}$. $\mathcal{B}$ gives params to $\mathcal{A}$. Note that, for given *ID,* the corresponding private key associated to params is $S_{ID} = xQ_{ID} = xH_1(ID)$.

Without loss of generality, we assume that for any *ID,* $\mathcal{A}$ queries H_1, Send and Extract at most once, and Send and Extract queries for public keys are preceded by an H_1 hash query. To avoid collision and consistently respond to these queries $\mathcal{B}$ maintains two lists L_{H_1} and L_T of $\langle ID_i, r_i, Q_i\rangle$ and $\langle ID_j, a_jP\rangle$, respectively. The lists are initially empty. Algorithm $\mathcal{B}$ interacts with $\mathcal{A}$ as follows:

- When $\mathcal{A}$ issues $H_1(ID^*)$ query, $\mathcal{B}$ returns $Q^* = yP$. For all other H_1 queries, $\mathcal{B}$ picks a random $r_i \in \mathbb{Z}_q^*$ and adds $\langle ID_i, r_i, Q_i\rangle$ to L_{H_1}, and returns $Q_i = r_iP$ to $\mathcal{A}$.
- When $\mathcal{A}$ issues Extract query on Q_i, if $Q_i = Q^*$, then $\mathcal{B}$ outputs FAIL and aborts. Otherwise, $\mathcal{B}$ finds the tuple of the form $\langle ID_i, r_i, Q_i\rangle$ in L_{H_1}, and returns private keys $r_iP_{pub} = r_ixP = xr_iP = xQ_i$ to $\mathcal{A}$.
- When $\mathcal{A}$ issues an H query for a_iP, then $\mathcal{B}$ picks a random $h_i \in \mathbb{Z}_q^*$ and returns h_i to $\mathcal{A}$.
- When $\mathcal{A}$ issues a Send query on ID_i, $\mathcal{B}$ picks a random $a_i \in \mathbb{Z}_q^*$ and computes a_iP, and adds the tuple $\langle ID_i, a_iP\rangle$ to L_T. $\mathcal{B}$ finds the tuple of the form $\langle ID_i, r_i, Q_i\rangle$ in L_{H_1}. Then $\mathcal{B}$ computes $T_i = a_ixP + h_ir_ixP = a_iP_{pub} + h_iS_i$ and returns $\langle ID_i, a_iP, T_i\rangle$ to $\mathcal{A}$.

Eventually, $\mathcal{A}$ outputs a valid tuple$\langle ID^*, aP, h, T\rangle$ such that $\langle ID^*, aP\rangle \notin L_T$, which is expected to be valid for the fixed ID^*, without accessing any oracles expect H. By replays of $\mathcal{B}$ with the same random tape but different choices of

H, as done in the *forking lemma* (theorem 3) [20], $\mathcal{A}$ outputs two valid tuples $\langle ID^*, aP, h, T\rangle$ and $\langle ID^*, aP, h', T'\rangle$ such that $h \neq h'$. If both outputs are expected ones, then $\mathcal{B}$ computes $(T-T')/(h-h') = xyP$ and outputs it. Otherwise, $\mathcal{B}$ outputs FAIL and aborts.

The total running time t_2 of $\mathcal{B}$ is equal to the running time of the *forking lemma* (theorem 3)[20] which is bounded by $120686 q_H t_1/\varepsilon_1$, as desired.

New Security Results on Encrypted Key Exchange

Emmanuel Bresson[1], Olivier Chevassut[2], and David Pointcheval[3]

[1] Dépt Cryptologie, CELAR, 35174 Bruz Cedex, France
Emmanuel.Bresson@polytechnique.org
[2] Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA,
OChevassut@lbl.gov – http://www.itg.lbl.gov/~chevassu
[3] Dépt d'informatique, École normale supérieure, 75230 Paris Cedex 05, France
David.Pointcheval@ens.fr – http://www.di.ens.fr/users/pointche

Abstract. Schemes for *encrypted key exchange* are designed to provide two entities communicating over a public network, and sharing a (short) password only, with a session key to be used to achieve data integrity and/or message confidentiality. An example of a very efficient and "elegant" scheme for *encrypted key exchange* considered for standardization by the IEEE P1363 Standard working group is AuthA. This scheme was conjectured secure when the symmetric-encryption primitive is instantiated via either a cipher that closely behaves like an "ideal cipher", or a mask generation function that is the product of the message with a hash of the password. While the security of this scheme in the former case has been recently proven, the latter case was still an open problem. For the first time we prove in this paper that this scheme is secure under the assumptions that the hash function closely behaves like a random oracle and that the computational Diffie-Hellman problem is difficult. Furthermore, since Denial-of-Service (DoS) attacks have become a common threat we enhance AuthA with a mechanism to protect against them.

1 Introduction

The need for authentication is obvious when two entities communicate on the Internet. However, proving knowledge of a secret over a public link without leaking any information about this secret is a complex process. One extreme example is when a short string is used by a human as a means to get access to a remote service. This password is used by the human to authenticate itself to the remote service in order to establish a session key to be used to implement an authenticated communication channel within which messages set over the wire are cryptographically protected. Humans directly benefit from this approach since they only need to remember a low-quality string chosen from a relatively small dictionary (i.e. 4 decimal digits).

The seminal work in this area is the *Encrypted Key Exchange* (EKE) protocol proposed by Bellovin and Merritt in [5, 6]. EKE is a classical Diffie-Hellman key exchange wherein the two flows are encrypted using the password as a common symmetric key. This encryption primitive can be instantiated via either a

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 145–158, 2004.

password-keyed symmetric cipher or a mask generation function computed as the product of the message with a hash of the password. This efficient structure later evolved into a protocol named AuthA considered for standardization by the IEEE P1363 Standard working group on public-key cryptography [3]. AuthA was conjectured secure against dictionary attacks by its designers, but actually proving it was left as an open problem.

Cryptographers have began to analyze the AuthA protocol in an ideal model of computation wherein a hash function is modeled via a random function and a block cipher is modeled via random permutations [2, 5, 8]. These analyses have provided useful arguments in favor of AuthA, but do not guarantee that AuthA is secure in the real world. These analyses only show that AuthA is secure against generic attacks that do not exploit a particular implementation of the block cipher, but in practice current block ciphers are far from being random permutations. A security proof in the random-oracle model only, while still using ideal objects, would provide a stronger and more convincing argument in favor of AuthA.

One should indeed note that the ideal-cipher model seems to be a stronger model than the random-oracle one. Even if one knows constructions to build random permutations from random functions [13], they cannot be used to build ideal ciphers from random oracles. The difference here comes from the fact that the inner functions (random oracles) are available to the adversary. It could compute plaintext-ciphertext relations starting from the middle of the Feistel network, while in the *programmable* ideal-cipher model, one needs to control all these relations.

Moreover, a AuthA scheme resistant to Denial-of-Service (DoS) attacks would be more suited to the computing environment we face every day since nowadays through the Internet hackers make servers incapable of accepting new connections. These so-called *Distributed DoS attacks* exhaust the memory and computational power of the servers.

Contributions. This paper examines the security of the AuthA password-authenticated key exchange protocol in the random-oracle model under the computational Diffie-Hellman assumption; no ideal-cipher assumption is needed. We work out our proofs by first defining the execution of AuthA in the communication model of Bellare *et al.* [2] and then adapting the proof techniques recently published by Bresson *et al.* [8]. We exhibit very compact and "elegant" proofs to show that the *One-Mask* (OMDHKE– one flow is encrypted only) and the *Two-Mask* (MDHKE– both flows are encrypted) formal variants of AuthA and EKE are secure in the random-oracle model when the encryption primitive is a mask generation function. Because of lack of space, the latter variant is postponed to the full version of this paper [9].

We define the execution of AuthA in the Bellare *et al.*'s model wherein the protocol entities are modeled through oracles, and the various types of attacks are modeled by queries to these oracles. This model enables a treatment of dictionary attacks by allowing the adversary to obtain honest executions of the

AuthA protocol. The security of AuthA against dictionary attacks depends on how many interactions the adversary carries out against the protocol entities rather than on the adversary's computational power.

We furthermore enhance the schemes with a mechanism that offers protection against Denial-of-Service (DoS) attacks. This mechanism postpones the computation of any exponentiations on the server side, as well as the storage of any states, after that the initiator of the connection has been identified as being a legitimate client. Roughly speaking, the server sends to the client a "puzzle" [12] to solve which will require from the client to perform multiple cryptographic computations while the server can easily and efficiently check that the solution is correct.

Related Work. The IEEE P1363.2 Standard working group on password-based authenticated key-exchange methods [11] has been focusing on key exchange protocols wherein clients use short passwords in place of certificates to identify themselves to servers. This standardization effort has its roots in the works of Bellare *et al.* [2] and Boyko *et al.* [7], wherein formal models and security goals for password-based key agreement were first formulated. Bellare *et al.* analyzed the EKE (where EKE stands for *Encrypted Key Exchange*) protocol [5], a classical Diffie-Hellman key exchange wherein the two flows are encrypted using the password as a common symmetric key. Several proofs have already been proposed, in various models, but all very intricate. The present paper provides a very short and "elegant" proof of AuthA or OMDHKE (but also of EKE or MDHKE in the full version), that is less prone to errors.

Several works have already focused on designing mechanisms to protect against DoS attacks. Aiello *et al.* [1] treat the amount of Perfect Forward-Secrecy (PFS) as an engineering parameter that can be traded off against resistance to DoS attacks. DoS-resistance is achieved by saving the "state" of the current session in the protocol itself (i.e., in the flows) rather than on the server side. More precisely, the "state" of the protocol is hashed and put into a *cookie,* while the server needs only to memorize the hash value. Only once this is done, the server saves the full state and the connection is established. This technique prevents the attacker from exhausting the server's memory but do not prevent it from exhausting the server's computational power. One approach to counter the latter threat is to make the client compute some form of proof of computational effort, using a "puzzle" [12], also more recently used by Dwork *et al.* [10] to discourage spam. The present paper builds on that latter concept.

2 The OMDHKE Protocol: One-Mask Diffie-Hellman Key Exchange

The arithmetic is in a finite cyclic group $\mathbb{G} = \langle g \rangle$ of order a ℓ-bit prime number q, where the operation is denoted multiplicatively. We also denote by $\mathbb{G}^\star$ the subset $\mathbb{G}\backslash\{1\}$ of the generators of $\mathbb{G}$. Hash functions from $\{0,1\}^\star$ to $\{0,1\}^{\ell_i}$ are denoted $\mathcal{H}_i$,for $i = 0, 1$. While $\mathcal{G}$ denotes a full-domain hash function from $\{0,1\}^\star$ into $\mathbb{G}$,

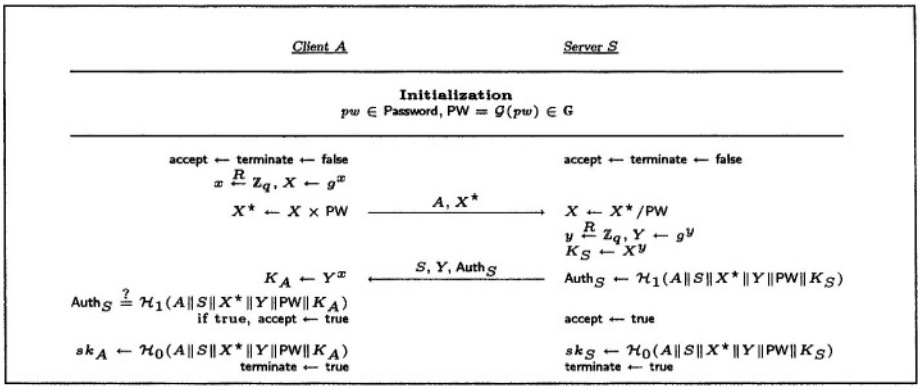


Fig. 1. An execution of the protocol OMDHKE, run between a client and a server.

As illustrated on Figure 1 (with an honest execution of the OMDHKE protocol), the protocol runs between two parties A and S, and the session-key space **SK** associated to this protocol is $\{0,1\}^{\ell_0}$ equipped with a uniform distribution.

The parties initially share a low-quality string pw, the password, drawn from the dictionary Password according to the distribution $\mathcal{D}_{pw}$. In the following, we use the notation $\mathcal{D}_{pw}(q)$ for the probability to be in the most probable set of q passwords:

$$\mathcal{D}_{pw}(q) = \max_{P \subseteq \mathsf{Password}} \left\{ \Pr_{pw \in \mathcal{D}_{pw}} [pw \in P \mid \#P \leq q] \right\}.$$

Note that if we denote by $\mathcal{U}_N$ the uniform distribution among N passwords, $\mathcal{U}_N(q) = q/N$.

The protocol then runs as follows. The client chooses at random a private random exponent x and computes the corresponding Diffie-Hellman public value g^x, but does not send this last value in the clear. The client encrypts the Diffie-Hellman public value using a mask generation function as the product of a Diffie-Hellman value with a full-domain hash of the password. Upon receiving this encrypted value, the server unmasks it and computes the Diffie-Hellman secret value g^{xy} which is used by the server to compute its authenticator Auth_S and the session key. The server sends its Diffie-Hellman public value g^y in the clear, Auth_S, and terminates the execution of the protocol. Upon receiving these values, the client computes the secret Diffie-Hellman value and checks that the authenticator Auth_S is a valid one. If the authenticator is valid, the client computes the session key, and terminates the execution of the protocol.

3 The Formal Model

The security model is the same as the one defined by Bellare *et al.* [2]. We briefly review it.

The Security Model. We denote by A and S two parties that can participate in the key exchange protocol P. Each of them may have several *instances* called oracles involved in distinct, possibly concurrent, executions of P. We denote A (resp. S) instances by A^i (resp. S^j), or by U when we consider any user instance. The two parties share a low-entropy secret pw which is drawn from a small dictionary Password, according to the distribution $\mathcal{D}_{pw}$.

The key exchange algorithm P is an interactive protocol between A^i and S^j that provides the instances of A and S with a session key sk. During the execution of this protocol, the adversary has the entire control of the network, and tries to break the privacy of the key, or the authentication of the players. To this aim, several queries are available to it. Let us briefly recall the capability that each query captures:

- Execute(A^i, S^j): This query models passive attacks, where the adversary gets access to honest executions of P between the instances A^i and S^j by eavesdropping.
- Reveal(U): This query models the misuse of the session key by instance U *(known-key attacks).* The query is only available to $\mathcal{A}$ if the attacked instance actually "holds" a session key and it releases the latter to $\mathcal{A}$.
- Send(U, m): This query enables to consider active attacks by having $\mathcal{A}$ sending a message to instance U. The adversary $\mathcal{A}$ gets back the response U generates in processing the message m according to the protocol P. A query Send$(A^i, \texttt{Start})$ initializes the key exchange algorithm, and thus the adversary receives the initial flow the player A should send out to the player S.

In the active scenario, the Execute-query may at first seem useless since using the Send-query the adversary has the ability to carry out honest executions of P among parties. Yet, even in this scenario, the Execute-query is essential for properly dealing with dictionary attacks. The number q_s of Send-queries directly asked by the adversary does not take into account the number of Execute-queries. Therefore, q_s represents the number of flows the adversary has built by itself, and therefore the number of passwords it would have tried.

Security Notions. As already noticed, the aim of the adversary is to break the privacy of the session key (a.k.a., semantic security) or the authentication of the players (having a player accepting while no instance facing him). The security notions take place in the context of executing P in the presence of the adversary $\mathcal{A}$. One first draws a password pw from Password according to the distribution $\mathcal{D}_{pw}$, provides coin tosses to $\mathcal{A}$, all oracles, and then runs the adversary by letting it ask any number of queries as described above, in any order.

AKE Security. The privacy (semantic security) of the session key is modeled by the game $\mathbf{Game}^{\mathsf{ake}}(\mathcal{A}, P)$, in which one more query is available to the adversary: Test(U).The Test-query can be asked at most once by the adversary $\mathcal{A}$ and is only available to $\mathcal{A}$ if the attacked instance U is **Fresh** (which roughly means that the session key is not "obviously" known to the adversary.) This query

is answered as follows: one flips a (private) coin b and forwards sk (the value $\mathsf{Reveal}(U)$ would output) if $b = 1$, or a random value if $b = 0$. When playing this game, the goal of the adversary is to guess the bit b involved in the Test-query, by outputting this guess b'. We denote the **AKE advantage** as the probability that $\mathcal{A}$ correctly guesses the value of b. More precisely we define $\mathsf{Adv}_P^{\mathsf{ake}}(\mathcal{A}) = 2\Pr[b = b'] - 1$. The protocol P is said to be (t, ε)-**AKE-secure** if $\mathcal{A}$'s advantage is smaller than ε for any adversary $\mathcal{A}$ running with time t.

Authentication. Another goal is to consider *unilateral authentication* of either A (A-Auth) or S (S-Auth) wherein the adversary impersonates a party. We denote by $\mathsf{Succ}_P^{\mathsf{A-auth}}(\mathcal{A})$ (resp. $\mathsf{Succ}_P^{\mathsf{S-auth}}(\mathcal{A})$) the probability that $\mathcal{A}$ successfully impersonates an A instance (resp. an S instance) in an execution of P, which means that S (resp. A) agrees on a key, while the latter is shared with no instance of A (resp. S). A protocol P is said to be (t, ε)-**Auth-secure** if $\mathcal{A}$'s success for breaking either A-Auth or S-Auth is smaller than ε for any adversary $\mathcal{A}$ running with time t.

3.1 Computational Diffie-Hellman Assumption

A (t, ε)-$\mathsf{CDH}_{g,\mathbb{G}}$ attacker, in a finite cyclic group $\mathbb{G}$ of prime order q with g as a generator, is a probabilistic machine Δ running in time t such that its success probability $\mathsf{Succ}_{g,\mathbb{G}}^{\mathsf{cdh}}(\Delta)$, given random elements g^x and g^y to output g^{xy}, is greater than ε. As usual, we denote by $\mathsf{Succ}_{g,\mathbb{G}}^{\mathsf{cdh}}(t)$ the maximal success probability over every adversaries running within time t. The CDH-Assumption states that $\mathsf{Succ}_{g,\mathbb{G}}^{\mathsf{cdh}}(t) \leq \varepsilon$ for any t/ε not too large.

4 Security Proof for the OMDHKE Protocol

In this section we show that the OMDHKE protocol distributes session keys that are semantically-secure and provides *unilateral authentication* of the server S. The specification of this protocol is found on Figure 1.

Theorem 1 (AKE/UA Security). *Let us consider the protocol* OMDHKE, *over a group of prime order* q, *where* Password *is a dictionary equipped with the distribution* $\mathcal{D}_{pw}$. *For any adversary* $\mathcal{A}$ *within a time bound* t, *with less than* q_s *active interactions with the parties (*Send*-queries) and* q_p *passive eavesdroppings (*Execute*-queries), and asking* q_g *and* q_h *hash queries to* $\mathcal{G}$ *and any* $\mathcal{H}_i$ *respectively,*

$$\mathsf{Adv}^{\mathsf{ake}}_{\mathsf{omdhke}}(\mathcal{A}) \leq \frac{2q_s}{2^{\ell_1}} + 12 \times \mathcal{D}_{pw}(q_s) + 12q_h^2 \times \mathsf{Succ}^{\mathsf{cdh}}_{g,\mathbb{G}}(t + 2\tau_e) + \frac{2Q^2}{q},$$

$$\mathsf{Succ}^{\mathsf{S-auth}}_{\mathsf{omdhke}}(\mathcal{A}) \leq \frac{q_s}{2^{\ell_1}} + 3 \times \mathcal{D}_{pw}(q_s) + 3q_h^2 \times \mathsf{Succ}^{\mathsf{cdh}}_{g,\mathbb{G}}(t + 3\tau_e) + \frac{Q^2}{2q},$$

where $Q = q_p + q_s + q_g$ *and* τ_e *denotes the computational time for an exponentiation in* $\mathbb{G}$.

This theorem shows that the protocol is secure against dictionary attacks since the advantage of the adversary essentially grows with the ratio of interactions (number of Send-queries) to the number of passwords.

Proof. In this proof, we incrementally define a sequence of games starting at the real game $\mathbf{G}_0$ and ending up at $\mathbf{G}_5$. We use the Shoup's lemma [14] to bound the probability of each event in these games.

Game $\mathbf{G}_0$: This is the real protocol, in the random-oracle model. We are interested in the two following events:

- S_0 (for semantic security), which occurs if the adversary correctly guesses the bit b involved in the Test-query;
- A_0 (for S-authentication), which occurs if an instance A^i accepts with no partner instance S^j (with the same transcript $((A, X^*), (S, Y, \mathsf{Auth}))$.)

$$\mathsf{Adv}^{\mathsf{ake}}_{\mathsf{omdhke}}(\mathcal{A}) = 2\Pr[\mathsf{S}_0] - 1 \qquad \mathsf{Succ}^{\mathsf{S-auth}}_{\mathsf{omdhke}}(\mathcal{A}) = \Pr[\mathsf{A}_0]. \tag{1}$$

Actually, in any game $\mathbf{G}_n$ below, we study the event A_n, and the restricted event $\mathsf{SA}_n = \mathsf{S}_n \wedge \neg\mathsf{A}_n$.

Game $\mathbf{G}_1$: In this game, we simulate the hash oracles ($\mathcal{G}$, $\mathcal{H}_0$ and $\mathcal{H}_1$, but also additional hash functions, for $i = 0, 1$: $\mathcal{H}'_i : \{0,1\}^\star \to \{0,1\}^{\ell_i}$ that will appear in the Game $\mathbf{G}_3$) as usual by maintaining hash lists $\Lambda_{\mathcal{G}}$, $\Lambda_{\mathcal{H}}$ and $\Lambda_{\mathcal{H}'}$ (see Figure 2). We also simulate all the instances, as the real players would do, for the Send-queries and for the Execute, Reveal and Test-queries (see Figure 3). From this simulation, we easily see that the game is perfectly indistinguishable from the real attack.

$\mathcal{G}$ and $\mathcal{H}_i$ oracles

For a hash-query $\mathcal{H}_i(q)$ (resp. $\mathcal{H}'_i(q)$), such that a record (i, q, r) appears in $\Lambda_{\mathcal{H}}$ (resp. $\Lambda_{\mathcal{H}'}$), the answer is r. Otherwise one chooses a random element $r \in \{0,1\}^\ell$, answers with it, and adds the record (i, q, r) to $\Lambda_{\mathcal{H}}$ (resp. $\Lambda_{\mathcal{H}'}$).

For a hash-query $\mathcal{G}(q)$ such that a record $(q, r, \star)$ appears in $\Lambda_{\mathcal{G}}$, the answer is r. Otherwise the answer r is defined according to the following rule:

▶**Rule** $\mathcal{G}^{(1)}$

Choose a random element $r \in \mathbb{G}$. The record $(q, r, \perp)$ is added to $\Lambda_{\mathcal{G}}$.

Note: the third component of the elements of this list will be explained later.

Fig. 2. Simulation of the hash functions

Game $\mathbf{G}_2$: For an easier analysis in the following, we cancel games in which some (unlikely) collisions appear:

- collisions on the partial transcripts $((A, X^*), (S, Y))$. Note that transcripts involve at least one honest party, and thus one of X^* or Y is truly uniformly distributed;
- collisions on the output of $\mathcal{G}$.

Send-queries to A

We answer to the Send-queries to an A-instance as follows:

- A $\mathsf{Send}(A^i, \mathtt{Start})$-query is processed according to the following rule:
  - ▶**Rule A1**$^{(1)}$
    Choose a random exponent $\theta \in \mathbb{Z}_q$, compute $X = g^\theta$ and $X^\star = X \times \mathsf{PW}$.

  Then the query is answered with $(A, X^\star)$, and the instance goes to an expecting state.
- If the instance A^i is in an expecting state, a query $\mathsf{Send}(A^i, (S, Y, \mathsf{Auth}))$ is processed by computing the authenticator and the session key. We apply the following rules:
  - ▶**Rule A2**$^{(1)}$
    Compute $K_A = Y^\theta$.
  - ▶**Rule A3**$^{(1)}$
    Compute the authenticator and the session key:
    $\mathsf{Auth}' = \mathcal{H}_1(A\|S\|X^\star\|Y\|\mathsf{PW}\|K_A)$;
    $sk_A = \mathcal{H}_0(A\|S\|X^\star\|Y\|\mathsf{PW}\|K_A)$.

  If $\mathsf{Auth} = \mathsf{Auth}'$, the instance accepts. In any case, the instance terminates.

Send-queries to S

We answer to the Send-queries to a S-instance as follows:

- A $\mathsf{Send}(S^j, (A, X^\star))$-query is processed according to the following rules:
  - ▶**Rule S1**$^{(1)}$
    Choose a random exponent $\varphi \in \mathbb{Z}_q$, compute $Y = g^\varphi$.

  Then, the instance compute the authenticator and session key. We apply the following rules:
  - ▶**Rule S2**$^{(1)}$
    Compute $X = X^\star/\mathsf{PW}$ and $K_S = X^\varphi$.
  - ▶**Rule S3**$^{(1)}$
    Compute the authenticator and the session key:
    $\mathsf{Auth} = \mathcal{H}_1(A\|S\|X^\star\|Y\|\mathsf{PW}\|K_S)$;
    $sk_S = \mathcal{H}_0(A\|S\|X^\star\|Y\|\mathsf{PW}\|K_S)$.

  Then the query is answered with (S, Y, Auth), and the instance accepts and terminates.

Other queries

An $\mathsf{Execute}(A^i, S^j)$-query is processed using successively the above simulations of the Send-queries: $(A, X^\star) \leftarrow \mathsf{Send}(A^i, \mathtt{Start})$ and $(S, Y, \mathsf{Auth}) \leftarrow \mathsf{Send}(S^j, (A, X^\star))$, and then outputting the transcript $((A, X^\star), (S, Y, \mathsf{Auth}))$.

A $\mathsf{Reveal}(U)$-query returns the session key (sk_A or sk_S) computed by the instance U (if the latter has accepted).

A $\mathsf{Test}(U)$-query first gets sk from $\mathsf{Reveal}(U)$, and flips a coin b. If $b = 1$, we return the value of the session key sk, otherwise we return a random value drawn from $\{0,1\}^\ell$.

Fig. 3. Simulation of the OMDHKE protocol

Both probabilities are bounded by the birthday paradox:

$$\Pr[\mathsf{Coll}_2] \leq \frac{(q_p + q_s)^2}{2q} + \frac{q_g^2}{2q}. \tag{2}$$

Game $\mathbf{G}_3$: We compute the session key sk and the authenticator Auth using the private oracles $\mathcal{H}'_0$ and $\mathcal{H}'_1$ respectively:

►**Rule A3/S3**$^{(3)}$

| Compute the authenticator $\mathsf{Auth} = \mathcal{H}'_1(A\|S\|X^\star\|Y)$.
| Compute the session key $sk_{A/S} = \mathcal{H}'_0(A\|S\|X^\star\|Y)$.

Since we do no longer need to compute the values K_A and K_S, we can simplify the second rules:

►**Rule A2/S2**$^{(3)}$

| Do nothing.

Finally, one can note that the password is not used anymore either, then we can also simplify the generation of $X^\star$, using the group property of $\mathbb{G}$:

►**Rule A1**$^{(3)}$

| Choose a random element $x \in \mathbb{Z}_q$ and compute $X^\star = g^x$.

The games $\mathbf{G}_3$ and $\mathbf{G}_2$ are indistinguishable unless some specific hash queries are asked, denoted by event $\mathsf{AskH}_3 = \mathsf{AskH0w1}_3 \vee \mathsf{AskH1}_3$:

- $\mathsf{AskH1}_3$: $\mathcal{A}$ queries $\mathcal{H}_1(A\|S\|X^\star\|Y\|\mathsf{PW}\|K_A)$ or $\mathcal{H}_1(A\|S\|X^\star\|Y\|\mathsf{PW}\|K_S)$ for some execution transcript $((A, X^\star), (S, Y, \mathsf{Auth}))$;
- $\mathsf{AskH0w1}_3$: $\mathcal{A}$ queries $\mathcal{H}_0(A\|S\|X^\star\|Y\|\mathsf{PW}\|K_A)$ or $\mathcal{H}_0(A\|S\|X^\star\|Y\|\mathsf{PW}\|K_S)$ for some execution transcript $((A, X^\star), (S, Y, \mathsf{Auth}))$, where some party has accepted, but event $\mathsf{AskH1}_3$ did not happen.

The authenticator is computed with a random oracle that is private to the simulator, then one can remark that it cannot be guessed by the adversary, better than at random for each attempt, unless the same partial transcript $((A, X^*), (S, Y))$ appeared in another session with a real instance S^j. But such a case has already been excluded (in Game $\mathbf{G}_2$). A similar remark can be led about the session key:

$$\Pr[\mathsf{A}_3] \leq \frac{q_s}{2^{\ell_1}} \qquad \Pr[\mathsf{SA}_3] = \frac{1}{2}. \tag{3}$$

When collisions of partial transcripts have been excluded, the event AskH1 can be split in 3 disjoint sub-cases:

- $\mathsf{AskH1\text{-}Passive}_3$: the transcript $((A, X^\star), (S, Y, \mathsf{Auth}))$ comes from an execution between instances of A and S (Execute-queries or forward of Send-queries, replay of part of them). This means that both $X^\star$ and Y have been simulated;

- $\mathsf{AskH1\text{-}WithA}_3$: the execution involved an instance of A, but Y has not been sent by any instance of S. This means that $X^\star$ has been simulated, but Y has been produced by the adversary;
- $\mathsf{AskH1\text{-}WithS}_3$: the execution involved an instance of S, but $X^\star$ has not been sent by any instance of A. This means that Y has been simulated, but $X^\star$ has been produced by the adversary.

Game $\mathbf{G}_4$: In order to evaluate the above events, we introduce a random Diffie-Hellman instance (P, Q), (with both $P \in \mathbb{G}^\star$ and $Q \in \mathbb{G}^\star$, which are thus generators of $\mathbb{G}$. Otherwise, the Diffie-Hellman problem is easy.) We first modify the simulation of the oracle $\mathcal{G}$, involving the element Q. The simulation introduces values in the third component of the elements of $\Lambda_{\mathcal{G}}$, but does not use it.

►**Rule $\mathcal{G}^{(4)}$**

Choose a random element $k \in \mathbb{Z}_q^\star$ and compute $r = Q^{-k}$.
The record (q, r, k) is added to $\Lambda_{\mathcal{G}}$.

We introduce the other part P of the Diffie-Hellman instance in the simulation of the party S.

►**Rule $\mathbf{S1}^{(4)}$**

Choose a random element $y \in \mathbb{Z}_q^\star$ and compute $Y = P^y$.

It would let the probabilities unchanged, but note that we excluded the cases $\mathsf{PW} = 1$ and $Y = 1$:

$$|\Pr[\mathsf{AskH}_4] - \Pr[\mathsf{AskH}_3]| \leq \frac{q_s + q_p}{q} + \frac{q_g}{q}. \tag{4}$$

Game $\mathbf{G}_5$: It is now possible to evaluate the probability of the event AskH (or more precisely, the sub-cases). Indeed, one can remark that the password is never used during the simulation, it can be chosen at the very end only. Then, an information-theoretic analysis can be performed, which simply uses cardinalities of some sets.

To this aim, we first cancel a few more games, wherein for some pairs $(X^\star, Y) \in \mathbb{G}^2$, involved in a communication between **an instance** S^j and either the adversary or an instance A^i, there are two distinct elements PW such that the tuple $(X^\star, Y, \mathsf{PW}, \mathsf{CDH}_{g,\mathbb{G}}(X^\star/\mathsf{PW}, Y))$ is in $\Lambda_{\mathcal{H}}$ (which event is denoted CollH_5):

$$|\Pr[\mathsf{AskH}_5] - \Pr[\mathsf{AskH}_4]| \leq \Pr[\mathsf{CollH}_5]. \tag{5}$$

Hopefully, event CollH_5 can be upper-bounded, granted the following Lemma:

Lemma 2. *If for some pair $(X^\star, Y) \in \mathbb{G}^2$, involved in a communication with an instance S^j, there are two elements PW_0 and PW_1 such that $(X^\star, Y, \mathsf{PW}_i, Z_i)$ are in $\Lambda_{\mathcal{H}}$ with $Z_i = \mathsf{CDH}_{g,\mathbb{G}}(X^\star/\mathsf{PW}_i, Y))$, one can solve the computational Diffie-Hellman problem:*

$$\Pr[\mathsf{CollH}_5] \leq q_h^2 \times \mathsf{Succ}^{\mathsf{cdh}}_{g,\mathbb{G}}(t + \tau_e). \tag{6}$$

Proof. Assume there exist such elements $(X^\star, Y = P^y) \in \mathbb{G}^2$, $\mathsf{PW}_0 = Q^{-k_0}$, and $\mathsf{PW}_1 = Q^{-k_1}$. Note that

$$Z_i = \mathsf{CDH}_{g,\mathbb{G}}(X^\star/\mathsf{PW}_i, Y) = \mathsf{CDH}_{g,\mathbb{G}}(X^\star \times Q^{k_i}, Y)$$
$$= \mathsf{CDH}_{g,\mathbb{G}}(X^\star, Y) \times \mathsf{CDH}_{g,\mathbb{G}}(Q, Y)^{k_i} = \mathsf{CDH}_{g,\mathbb{G}}(X^\star, Y) \times \mathsf{CDH}_{g,\mathbb{G}}(P, Q)^{yk_i}.$$

As a consequence, $Z_1/Z_0 = \mathsf{CDH}_{g,\mathbb{G}}(P, Q)^{y(k_1-k_0)}$, and thus $\mathsf{CDH}_{g,\mathbb{G}}(P, Q) = (Z_1/Z_0)^u$, where u is the inverse of $y(k_1 - k_0)$ in $\mathbb{Z}_q$. The latter exists since $\mathsf{PW}_1 \neq \mathsf{PW}_2$, and $y \neq 0$. By guessing the two queries asked to the $\mathcal{H}_i$, one concludes the proof. □

In order to conclude, let us study separately the three sub-cases of AskH1 and then AskH0w1 (keeping in mind the absence of several kinds of collisions: for partial transcripts, for $\mathcal{G}$, and for PW in $\mathcal{H}$-queries):

- AskH1-Passive: About the passive transcripts (in which both $X^\star$ and Y have been simulated), one can state the following lemma:

  Lemma 3. *If for some pair $(X^\star, Y) \in \mathbb{G}^2$, involved in a passive transcript, there is an element PW such that $(X^\star, Y, \mathsf{PW}, Z)$ is in $\Lambda_{\mathcal{H}}$, with $Z = \mathsf{CDH}_{g,\mathbb{G}}(X^\star/\mathsf{PW}, Y)$), one can solve the computational Diffie-Hellman problem:*

  $$\Pr[\mathsf{AskH1\text{-}Passive}_5] \leq q_h \times \mathsf{Succ}^{\mathsf{cdh}}_{g,\mathbb{G}}(t + 2\tau_e).$$

  Proof. Assume there exist such elements $(X^\star = g^x, Y = P^y) \in \mathbb{G}^2$ and $\mathsf{PW} = Q^{-k}$. As above,

  $$Z = \mathsf{CDH}_{g,\mathbb{G}}(X^\star, Y) \times \mathsf{CDH}_{g,\mathbb{G}}(Q, Y)^k = P^{xy} \times \mathsf{CDH}_{g,\mathbb{G}}(P, Q)^{yk}.$$

  As a consequence, $\mathsf{CDH}_{g,\mathbb{G}}(P, Q) = (Z/P^{xy})^u$, where u is the inverse of yk in $\mathbb{Z}_q$. The latter exists since we have excluded the cases where $y = 0$ or $k = 0$. By guessing the query asked to the $\mathcal{H}_i$, one concludes the proof. □

- AskH1-WithA: this event may correspond to an attack where the adversary tries to impersonate S to A (break unilateral authentication). But each authenticator sent by the adversary has been computed with at most one PW value. Without any $\mathcal{G}$-collision, it corresponds to at most one *pw:*

  $$\Pr[\mathsf{AskH1\text{-}WithA}_5] \leq \mathcal{D}_{pw}(q_s).$$

- AskH1-WithS: The above Lemma 2, when applied to games where the event Coll_5 did not happen (and without $\mathcal{G}$-collision), states that for each pair $(X^\star, Y)$ involved in a transcript with an instance S^j, there is at most one element *pw* such that for $\mathsf{PW} = \mathcal{G}(pw)$ the corresponding tuple is in $\Lambda_{\mathcal{H}}$: the probability over a random password is thus less than $\mathcal{D}_{pw}(q_s)$. As a consequence,

  $$\Pr[\mathsf{AskH1\text{-}WithS}_5] \leq \mathcal{D}_{pw}(q_s).$$

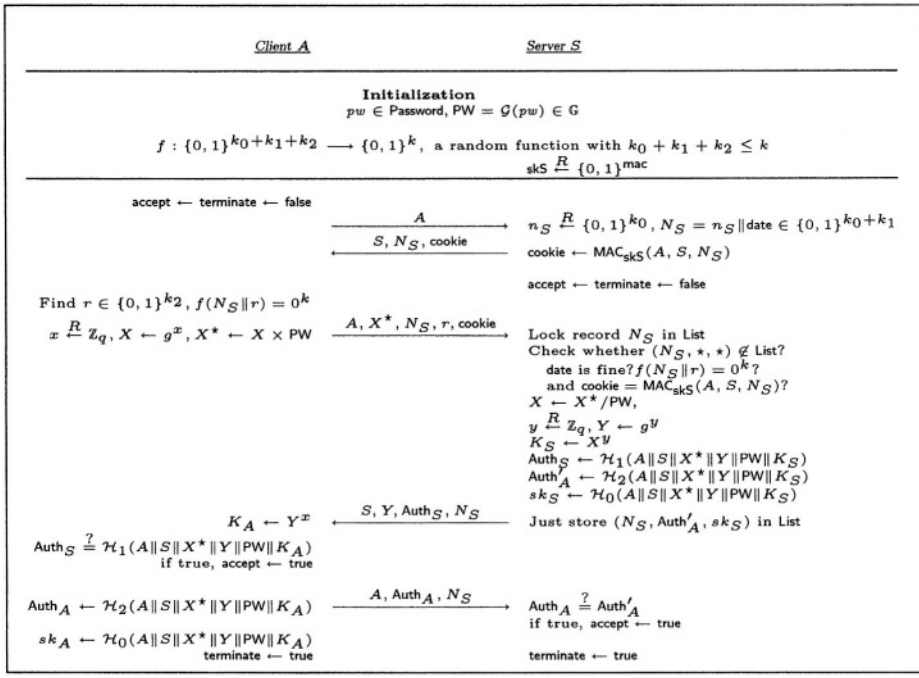


Fig. 4. An execution of the protocol OMDHKE, run between a client and a server, enhanced with mutual authentication and a denial-of-service protection.

About AskH0w1 (when the three above events did not happen), it means that only executions with an instance of S (and either A or the adversary) may lead to acceptation. Exactly the same analysis as for AskH1-Passive and AskH1-WithS leads to $\Pr[\mathsf{AskH0w1}_5] \leq \mathcal{D}_{pw}(q_s) + q_h \times \mathsf{Succ}^{\mathsf{cdh}}_{g,\mathbb{G}}(t + 2\tau_e)$. As a conclusion,

$$\Pr[\mathsf{AskH}_5] \leq 3\mathcal{D}_{pw}(q_s) + 2q_h \times \mathsf{Succ}^{\mathsf{cdh}}_{g,\mathbb{G}}(t + 2\tau_e). \tag{7}$$

Combining all the above equations, one gets the announced result. □

5 The DoS-resistant OMDHKE Protocol

In a computing environment where Distributed DoS attacks are a continual threat, a server needs to protect itself from non-legitimate clients that will exhaust its memory and computational power. Intensive cryptographic computations (i.e. exponentiation), as well as states, are only performed after a client proves to the server that it was able to solve a given "puzzle". The "puzzle" is chosen so that the client can only solve it by exhaustive search while the server can quickly checks whether a given proposition solves it. This "puzzle" is chosen as follows.

The server first picks at random a MAC-symmetric key that it will use to authenticate *cookie;* the MAC-key is used across multiple connections. The server then forms the authenticated *cookie* which is the MAC of a random nonce and the date, and sends it to the client. The precision of the date is determined according to the level of DoS required. The use of a cookie makes the protocol stateless on the server side. Upon receiving the cookie, the client tries to find an input which hashes to the NULL value. Since this hash function is seen as a random oracle, the only way for the client to solve this "puzzle" is to run through all possible prefixed strings and query the random oracle [4]. Later in practice this function is instantiated using specific functions derived from standard hash functions such as SHA1. Once the client has found such a proof of computational effort, it sends it back with the authenticated cookie and its Diffie-Hellman public value to the server. Upon receiving these values the server checks whether the client is launching a DoS attack by initiating several connections in parallel and replaying this proof of computational effort on another connection. The server reaches this aim by locking the cookie and not admitting the same cookie twice (hence the date in this challenge is used to tune the size of the database). If all the checks verify, the server starts saving states and computing the necessary exponentiations to establish a session key. From this point on the protocol works as the original AuthA protocol, adding mutual authentication [2].

6 Conclusion

The above proof does not deal with forward-secrecy. Forward-secrecy entails that the corruption of the password does not compromise the semantic security of previously established session keys.One could easily prove that this scheme achieves forward secrecy, as in [8], while loosing a quadratic factor in the reduction.

In conclusion, this paper provides strong security arguments that support the standardization of the AuthA protocol by the IEEE P1363.2 Standard working group on password-based public key cryptography. We have presented a compact and "elegant" proof of security for the AuthA protocol [3] when the symmetric-encryption primitive is instantiated using a mask generation function, which extends our previous work when the symmetric-encryption primitive is assumed to behave like an ideal cipher [8]. The security of the protocol was indeed stated as an open problem by its designers. In our study, the symmetric encryption basic block takes the form of a multiplication in the Diffie-Hellman group. Our result is a significant departure from previous known results since the security of AuthA can now be based on weaker and more reasonable assumptions involving both the random-oracle model and the computational Diffie-Hellman problem. Moreover, we investigate and propose a practical, reasonable solution to make the protocol secure against DoS attacks. One can also find further studies on the variant in which both flows are encrypted between the client and the server in the full version of this paper [9].

Acknowledgments

The second author was supported by the Director, Office of Science, Office of Advanced Scientific Computing Research, Mathematical Information and Computing Sciences Division, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098. This document is report LBNL-53099. Disclaimer available at http://www-library.lbl.gov/disclaimer.

References

1. W. Aiello, S. M. Bellovin, M. Blaze, R. Canetti, J. Ioannidis, A. D. Keromytis, and O. Reingold. Efficient, DoS-resistant, Secure Key Exchange for Internet Protocols. In *Proc. of the 9th CCS,* pages 48–58. ACM Press, New York, 2002.
2. M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated Key Exchange Secure Against Dictionary Attacks. In *Eurocrypt '00,* LNCS 1807, pages 139–155. Springer-Verlag, Berlin, 2000.
3. M. Bellare and P. Rogaway. The AuthA Protocol for Password-Based Authenticated Key Exchange. Contributions to IEEE P1363. March 2000. Available from `http://grouper.ieee.org/groups/1363/`.
4. M. Bellare and P. Rogaway. Random Oracles Are Practical: a Paradigm for Designing Efficient Protocols. In *Proc. of the 1st CCS,* pages 62–73. ACM Press, New York, 1993.
5. S. M. Bellovin and M. Merritt. Encrypted Key Exchange: Password-Based Protocols Secure against Dictionary Attacks. In *Proc. of the Symposium on Security and Privacy,* pages 72–84. IEEE, 1992.
6. S. M. Bellovin and M. Merritt. Augmented Encrypted Key Exchange: A Password-Based Protocol Secure against Dictionary Attacks and Password File Compromise. In *Proc. of the 1st CCS,* pages 244–250. ACM Press, New York, 1993.
7. V. Boyko, P. MacKenzie, and S. Patel. Provably Secure Password Authenticated Key Exchange Using Diffie-Hellman. In *Eurocrypt '00,* LNCS 1807, pages 156–171. Springer-Verlag, Berlin, 2000. Final version available at: `http://cm.bell-labs.com/who/philmac/research/`.
8. E. Bresson, O. Chevassut, and D. Pointcheval. Security Proofs for Efficient Password-Based Key Exchange. In *Proc. of the 10th CCS.* ACM Press, New York, 2003. The full version is available on the Cryptology ePrint Archive 2002/192.
9. E. Bresson, O. Chevassut, and D. Pointcheval. New Security Results on Encrypted Key Exchange. In *PKC '04,* LNCS. Springer-Verlag, Berlin, 2004. Full version available at: `http://www.di.ens.fr/users/pointche/`.
10. C. Dwork, A. Goldberg, and M. Naor. On Memory-Bound Functions for Fighting Spam. In *Crypto '03,* LNCS 2729, pages 426–444. Springer-Verlag, Berlin, 2003.
11. IEEE Standard 1363.2 Study Group. Password-Based Public-Key Cryptography. Available from `http://grouper.ieee.org/groups/1363/passwdPK`.
12. A. Juels and J. Brainard. Client Puzzles: A Cryptographic Defense Against Connection Depletion Attacks. In *Proc. of NDSS '99,* pages 151–165, 1999.
13. M. Luby and Ch. Rackoff. How to Construct Pseudorandom Permutations from Pseudorandom Functions. *SIAM Journal of Computing,* 17(2):373–386, 1988.
14. V. Shoup. OAEP Reconsidered. *Journal of Cryptology,* 15(4):223–249, September 2002.

New Results on the Hardness of Diffie-Hellman Bits

María Isabel González Vasco[1], Mats Näslund[2], and Igor E. Shparlinski[3]

[1] Dept. of Mathematics, University of Oviedo
Oviedo 33007, Spain
mvasco@orion.ciencias.uniovi.es
[2] Ericsson Research
SE-16480 Stockholm, Sweden
mats.naslund@ericsson.com
[3] Dept. of Computing, Macquarie University
Sydney, NSW 2109, Australia
igor@ics.mq.edu.au

Abstract. We generalize and extend results obtained by Boneh and Venkatesan in 1996 and by González Vasco and Shparlinski in 2000 on the hardness of computing bits of the Diffie-Hellman key, given the public values. Specifically, while these results could only exclude (essentially) error-free predictions, we here exclude any non-negligible advantage, though for larger fractions of the bits. We can also demonstrate a trade-off between the tolerated error rate and the number of unpredictable bits.
Moreover, by changing computational model, we show that even a very small proportion of the most significant bits of the Diffie–Hellman secret key cannot be retrieved from the public information by means of a Las Vegas type algorithm, unless the corresponding scheme is weak itself.

1 Introduction

So called "provable" security models, in which the robustness of a cryptographic tool can be justified by means of a formal proof, are gaining more and more attention. In such models, the security of a scheme or protocol is measured in terms of the chances (non-negligible advantage over a random guess) a malicious adversary has of retrieving information he is not supposed to have access to. There are already several proposals for schemes that are robust in this sense, some of them merely theoretical but others already deployed in practice. Much research has been devoted to this topic, and there is indeed a large battery of results for various schemes in different computational models (e.g. [2,3,9,10]).

Since the early days of cryptography, one security property that has been extensively studied is the security with respect to "approximate cracking". Robustness in this sense is stated by proving that single bits in an encrypted message are no easier to obtain than the whole message itself (or other information

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 159–172, 2004.

close to the secret in some metric). General frameworks for such studies are for example the *hard-core bit problem,* first formalized in [4], and the *hidden number problem,* introduced by Boneh and Venkatesan [6,7].

In addition, any security property can be studied in different computational models, ranging from the classical Turing machine model, passive/active adversaries, restricted algebraic models, up to the more recent quantum and side-channel attack models, the latter two being more "physical" in nature. Indeed, models that might seem unrealistic today could become a reality in 20 years, a time-span which may be required to cryptographically guard secrets in many applications. We therefore believe it is important to keep an open mind to various models and investigate which implications they have.

In this paper we extend the area of application of algorithms for the *hidden number problem,* deriving new bit security properties of the Diffie–Hellman key exchange scheme. Detailed surveys of bit security results for various cryptographic schemes are given in [14]; several more recent results can be found in [5,6,7,15,16,17,20,22,26,27,32,34,35].

We show that making some adjustments to the scheme proposed in [6] and refined in [16], one can obtain bit security results for the Diffie-Hellman secret key of the same strength as in [6,16], but in a much more restricted computational model of unreliable oracles, which represent adversaries that only retrieve correct guesses for the target bits with a certain probability. We perform the study with two types of unreliable oracles, roughly corresponding to the classical "Monte Carlo" type algorithms, as well as the in cryptography less conventional "Las Vegas" type of algorithm. In fact, we also obtain an improvement of the result of [16] for "error-free" oracles, as we use the recent bound of exponential sums over small subgroups from [8] instead of the bound from [21] that lead to the result in [16]. Also, our Lemma 3 is based on a recent improvement [1] in lattice reduction algorithms, whereas in [16] older results were applied.

2 Notation

As usual we assume that for a prime p the field $\mathbb{F}_p$ of p elements is represented by the set $\{0, 1, \ldots, p-1\}$. Accordingly, sometimes, where obvious, we treat elements of $\mathbb{F}_p$ as integer numbers in the above range. Also, for an integer s we denote by $\lfloor s \rfloor_p$ the remainder of s on division by p.

For a real $\eta > 0$ and $t \in \mathbb{F}_p$ we denote by $\mathrm{MSB}_{\eta,p}(t)$ any integer which satisfies the inequalities

$$\frac{p}{2^\eta}(\mathrm{MSB}_{\eta,p}(t) - 1) \le t < \frac{p}{2^\eta}(\mathrm{MSB}_{\eta,p}(t)). \tag{1}$$

Thus, roughly speaking, $\mathrm{MSB}_{\eta,p}(t)$ is the integer defined by the η most significant bits of t. However, this definition is more flexible and better suited to our purposes. In particular note that η in the inequality (1) need not be an integer.

Throughout the paper $\log x$ denotes the binary logarithm of $x \geq 1$. The implied constants in the symbol *"O"* may occasionally, where obvious, depend on a real parameter $\varepsilon > 0$ and are absolute otherwise.

We denote by $\mathbb{E}[\xi]$ the expected value of a random variable ξ. Accordingly, $\mathbb{E}_\xi[g(\xi)]$ denotes the expected value of a random variable $g(\xi)$, which, for a given function g, only depends on the distribution of ξ. We make use of the following variant of the Markov inequality: for positive c and a random variable ξ upper bounded by M,

$$\Pr[\xi \geq \mathbb{E}_\xi[\xi]/c] \geq M^{-1}(1 - 1/c)\mathbb{E}_\xi[\xi]. \tag{2}$$

3 Preparations

Reconstructing g^{ab} from "noisy" approximations of g^{ab} can be formulated as a *hidden number problem,* [6,7]. We review important ingredients for this problem. In particular, we collect several useful results about the hidden number problem, lattices and exponential sums and establish some links between these techniques.

3.1 Hidden Number Problem and Uniform Distribution mod p

One of many possible variations of the hidden number problem is:

> *Given a finite sequence $\mathcal{T}$ of elements of $\mathbb{F}_p^*$, recover $\alpha \in \mathbb{F}_p^*$ for which for polynomially many known random $t \in \mathcal{T}$ we are given $\mathrm{MSB}_{\eta,p}(\alpha t)$ for some $\eta > 0$.*

The case of $\mathcal{T} = \mathbb{F}_p^*$ is exactly the one considered in [6,7]. However, it has been noticed for the first time in [16], and exploited in a series of works, see [11,17,26,28,29], that in fact for cryptographic applications one has to consider more general sequences $\mathcal{T}$. An important issue is the uniformity of distribution of these sequences.

For a sequence of N points $0 \leq \vartheta_1, \ldots, \vartheta_N < 1$ define its *discrepancy* D by

$$D = \sup_{0 \leq \gamma < 1} \left| \frac{T(\gamma)}{N} - \gamma \right|,$$

where $T(\gamma)$ is the number of points of this sequence in the interval $[0, \gamma]$.

We say that a finite sequence $\mathcal{T}$ of integers is *Δ-homogeneously distributed modulo a prime p* if for any integer a with $\gcd(a, p) = 1$, the discrepancy $\mathcal{D}_a(\mathcal{T})$ of the sequence of fractional parts $\{at/p\}$, $t \in \mathcal{T}$, satisfies $\mathcal{D}_a(\mathcal{T}) \leq \Delta$. It has been shown in Lemma 4 of [28] that the algorithm of [6] can be modified to work for sequences that are Δ-homogeneously distributed modulo a prime p, provided Δ is small enough.

3.2 Lattices

As in the pioneering papers [6,7], our results rely on rounding techniques in lattices. We briefly review a few results and definitions. For general references on lattice theory and its important cryptographic applications, we refer to [18] and also to the recent surveys [30,31].

A basic lattice problem is the *closest vector problem* (CVP): given a basis of a lattice L in $\mathbb{R}^s$ and a target $\mathbf{u} \in \mathbb{R}^s$, find a lattice vector $\mathbf{v} \in L$ which minimizes the Euclidean norm $\|\mathbf{u}-\mathbf{v}\|$ among all lattice vectors. A modification where $\mathbf{u} = 0$ is a zero vector (thus $\mathbf{u} \in L$) is the *shortest vector problem* (SVP): find a nonzero $\mathbf{v} \in L$ of smallest Euclidean norm $\|\mathbf{v}\|$ among all lattice vectors.

Here, as in [28], we use the best CVP approximation polynomial-time result known, which follows from the recent shortest vector algorithm of [1] combined with the reduction of [23] from approximating the CVP to approximating the SVP, which leads to the following statement:

Lemma 1. *For any constant $\gamma > 0$, there exists a randomized polynomial time algorithm which, given a lattice L and a vector $\mathbf{r} \in \mathbb{Q}^s$, finds a lattice vector $\mathbf{v}$ satisfying with probability exponentially close to 1 the inequality*

$$\|\mathbf{v}-\mathbf{r}\| \le 2^{\gamma s \log\log s/\log s} \min\{\|\mathbf{z}-\mathbf{r}\|, \quad \mathbf{z} \in L\}.$$

For integers $t_1, \ldots, t_d$ selected in the interval $[0, p-1]$, we denote by $L(t_1, \ldots, t_d)$ the full rank $d+1$-dimensional lattice generated by the rows of the following $(d+1) \times (d+1)$-matrix

$$\begin{pmatrix} p & 0 & \ldots & 0 & 0 \\ 0 & p & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & p & 0 \\ t_1 & t_2 & \ldots & t_d & 1/p \end{pmatrix}. \tag{3}$$

Our principal tool is an extension of Lemma 4 of [28], which in turn extends the algorithm of [6]. The results below are analogues of Lemmas 6.2 and 6.3 of [35]. For applications to Diffie-Hellman we deal with sequences corresponding to small finite subgroups of $\mathbb{F}_p^*$ which satisfy the above requirement of Δ-homogeneous distribution, so Lemma 4 of [28] can be applied directly to them.

Lemma 2. *Assume that a real μ and an integer d satisfy*

$$d(\mu - \log 5) \ge 2 \log p$$

and let α be a fixed integer in the interval $[0, p-1]$. Assume that $t_1, \ldots, t_d$ are chosen uniformly and independently at random from a finite $2^{-\mu}$-homogeneously

distributed integer sequence $\mathcal{T}$ *modulo* p. *Then with probability* $P \geq 1 - 1/p$ *for any vector* $\mathbf{s} = (s_1, \ldots, s_d, 0)$ *with*

$$\left(\sum_{i=1}^{d} \left(\lfloor \alpha t_i \rceil_p - s_i \right)^2 \right)^{1/2} \leq p2^{-\mu},$$

all vectors $\mathbf{v} = (v_1, \ldots, v_d, v_{d+1}) \in L(t_1, \ldots, t_d)$ *satisfying*

$$\left(\sum_{i=1}^{d} (v_i - s_i)^2 \right)^{1/2} \leq p2^{-\mu},$$

are such that

$$v_i \equiv \beta t_i \pmod{p}, \quad i = 1, \ldots, d, \qquad v_{d+1} = \beta/p$$

with some $\beta \equiv \alpha \pmod{p}$.

Proof. We define the *modular norm* of an integer γ modulo p as

$$\|\gamma\|_p = \min_{b \in \mathbb{Z}} |\gamma - bp|.$$

For any γ such that $\gamma \not\equiv 0 \pmod{p}$ the probability $P(\gamma)$ of

$$\|\gamma t\|_p > p2^{-\mu+1}$$

for an integer t chosen uniformly at random from the elements of a Δ-homogeneously distributed sequence modulo p is

$$P(\gamma) \geq 1 - 2^{-\mu+2} - \Delta.$$

Thus for the $2^{-\mu}$-homogeneously distributed sequence modulo p, $\mathcal{T}$, we have

$$P(\gamma) \geq 1 - \frac{5}{2^{\mu}}.$$

Therefore, for any $\beta \not\equiv \alpha \pmod{p}$,

$$\Pr\left[\exists i \in [1, d] \mid \|\beta t_i - \alpha t_i\|_p \geq p2^{-\mu+1} \right] = 1 - (1 - P(\beta - \alpha))^d \geq 1 - \left(\frac{5}{2^{\mu}} \right)^d,$$

where the probability is taken over integers $t_1, \ldots, t_d$ chosen uniformly and independently at random from the elements of $\mathcal{T}$.

Since for $\beta \not\equiv \alpha \pmod{p}$ there are only $p - 1$ possible values for the residue of β modulo p, we obtain

$$\Pr\left[\forall \beta \not\equiv \alpha \pmod{p}, \exists i \in [1, d] \mid \|\beta t_i - \alpha t_i\|_p > p2^{-\mu+1} \right] \geq 1 - (p-1) \left(\frac{5}{2^{\mu}} \right)^d > 1 - 1/p,$$

because of the conditions of the theorem.

The rest of the proof is identical to the proof of Theorem 5 of [6], we outline it for the sake of completeness.

Let us fix some integers $t_1, \ldots, t_d$ with

$$\min_{\beta \not\equiv \alpha \pmod p} \max_{i \in [1,d]} \|\beta t_i - \alpha t_i\|_p > p 2^{-\mu+1}. \tag{4}$$

Let $\mathbf{v}$ be a lattice point satisfying

$$\left(\sum_{i=1}^{d} (v_i - s_i)^2 \right)^{1/2} \le p 2^{-\mu}.$$

Clearly, since $\mathbf{v} \in L(t_1, \ldots, t_d)$, there are integers $\beta, z_1, \ldots, z_d$ such that

$$\mathbf{v} = (\beta t_1 - z_1 p, \ldots, \beta t_d - z_d p, \beta/p).$$

If $\beta \equiv \alpha \pmod p$, then we are done, so suppose that $\beta \not\equiv \alpha \pmod p$. In this case,

$$\begin{aligned} \left(\sum_{i=1}^{d} (v_i - s_i)^2 \right)^{1/2} &\ge \min_{i \in [1,d]} \|\beta t_i - s_i\|_p \\ &\ge \min_{i \in [1,d]} \left(\|\beta t_i - \alpha t_i\|_p - \|s_i - \alpha t_i\|_p \right) \\ &> p 2^{-\mu+1} - p 2^{-\mu} = p 2^{-\mu} \end{aligned}$$

that contradicts our assumption. As we have seen, the condition (4) holds with probability exceeding $1 - 1/p$ and the result follows. □

Lemma 3. *Let $1 > \tau > 0$ be an arbitrary absolute constant and p be a prime. Assume that a real η and an integer d satisfy*

$$\eta \ge \left\lceil \left(\tau \frac{\log p \log\log\log p}{\log\log p} \right)^{1/2} \right\rceil \qquad \text{and} \qquad d = \lceil 5 \log p/\eta \rceil .$$

Let $\mathcal{T}$ be a sequence of $2^{-\eta}$-homogeneously distributed integers modulo p. There exists a probabilistic polynomial-time algorithm $\mathcal{A}$ such that for any fixed integer $\alpha \in \mathbb{F}_p^*$, given $2d$ integers

$$t_i \qquad \text{and} \qquad s_i = \mathrm{MSB}_{\eta,p}(\alpha t_i), \qquad i = 1, \ldots, d,$$

its output satisfies for sufficiently large p

$$\Pr[\mathcal{A}(m, t_1, \ldots, t_d; s_1, \ldots, s_d) = \alpha] \ge 1 - p^{-1},$$

with probability taken over all $t_1, \ldots, t_d$ chosen uniformly and independently at random from the elements of $\mathcal{T}$ and all coin tosses of the algorithm $\mathcal{A}$.

Proof. We follow the same arguments as in the proof Theorem 1 of [6] which we briefly outline here for the sake of completeness. We refer to the first d vectors in the defining matrix of $L(t_1, \ldots, t_d)$ as p-vectors.

Multiplying the last row vector $(t_1, \ldots, t_d, 1/p)$ of the matrix (3) by α and subtracting certain multiples of p-vectors, we obtain a lattice point

$$\mathbf{u}_\alpha = (u_1, \ldots, u_d, \alpha/p) \in L(t_1, \ldots, t_d)$$

such that $|u_i - s_i| < p2^{-\eta}$, $i = 1, \ldots, d+1$. Therefore,

$$\min\left\{\sum_{i=1}^{d+1}(z_i - s_i)^2, \quad \mathbf{z} = (z_1, \ldots, z_d, z_{d+1})\right\} \le \sum_{i=1}^{d+1}(u_i - s_i)^2 \le (d+1)p^2 2^{-2\eta}.$$

Let $\mu = \eta/2$. One can verify that under the conditions of the theorem we have,

$$0.1\tau\frac{(d+1)\log\log(d+1)}{\log(d+1)} \le \mu - 1 \qquad \text{and} \qquad d(\mu - \log 5) \ge 2\log p.$$

Now we use the algorithm of Lemma 1 with $\mathbf{s} = (s_1, \ldots, s_d, 0)$ to find in probabilistic polynomial time a lattice vector

$$\mathbf{v} = (v_1, \ldots, v_d, v_{d+1}) \in L(t_1, \ldots, t_d)$$

such that

$$\left(\sum_{i=1}^{d}(v_i - s_i)^2\right)^{1/2} \le 2^{0.1\gamma(d+1)\log\log(d+1)/\log(d+1)}p(d+1)^{1/2}2^{-\eta} \le p2^{-\mu-1},$$

provided that p is sufficiently large. We also have

$$\left(\sum_{i=1}^{d}(u_i - s_i)^2\right)^{1/2} \le pd^{1/2}2^{-\eta} \le p2^{-\mu-1}.$$

Therefore,

$$\left(\sum_{i=1}^{d}(u_i - v_i)^2\right)^{1/2} \le p2^{-\mu}.$$

Applying Lemma 2, we see that $\mathbf{v} = \mathbf{u}_\alpha$ with probability at least $1 - 1/p$, and therefore, α can be recovered in polynomial time. □

3.3 Distribution of Exponential Functions Modulo p

To apply the results above, we will need to establish approximate uniform distribution of sequences of form $t_i = g^{u_i}$, $i = 1, 2, \ldots$. A procedure to establish such

results in general is to bound certain exponential sums, related to the sequences under consideration.

The following statement is a somewhat simplified version of Theorem 4 of [8] and greatly improve several previously known bounds fron [21,25], which have been used in [16].

Lemma 4. For *any* $\varepsilon > 0$ *there exists* $\delta > 0$ *such that for any element* $g \in \mathbb{F}_p$ *of multiplicative order* $T \geq p^{\varepsilon}$ *we have*

$$\max_{\gcd(c,p)=1} \left| \sum_{x=0}^{T-1} \exp\left(2\pi i c g^x / p\right) \right| \leq T^{1-\delta}.$$

Using Lemma 4 and arguing as in [16], we derive the following statement.

Lemma 5. *For any* $\varepsilon > 0$ *there exists* $\delta > 0$ *such that for any element* $g \in \mathbb{F}_p$ *of multiplicative order* $T \geq p^{\varepsilon}$ *the sequence* g^x, $x = 1, \ldots, T$, *is* $p^{-\delta}$*-homogeneously distributed modulo* p.

4 Bit Security of the Diffie-Hellman Scheme

Let us fix an element $g \in \mathbb{F}_p^*$ of multiplicative order q, where q is prime. We recall that classically, breaking the Diffie–Hellman scheme means the ability to recover the value of the *secret key* g^{xy} from publicly known values of g^x and g^y (with unknown x and y, of course).

The attacker, however, may pursue a more modest goal of recovering only partial information about the secret g^{xy}. For instance, the Legendre symbol of g^{xy} is trivially deducible from that of g^x, g^y. If only part of g^{xy} is used to derive a key for a secret key cryptosystem, this may be harmful enough. The purpose of the *bit security* results is to show that deriving such partial information is as hard as finding the whole key, which is believed to be infeasible.

It has been shown in [6,16] that recovering (without significant errors) about $\log^{1/2} p$ most significant bits of g^{xy} for every x and y is not possible unless the whole scheme is insecure.

However, it is already dangerous enough if the attacker possesses a *probabilistic* algorithm which recovers some bits of g^{xy} only for some, not too small, fraction of key exchanges. Here we obtain first results in this direction. We consider two types of attacking algorithms:

- more traditional Monte Carlo type algorithms where our results are weaker in terms of number of bits, but stronger in error-tolerance than those of [6,16];
- more powerful Las Vegas type algorithms where, given such an algorithm, our results are stronger than the case of deterministic algorithms obtained in [6,16]. Cryptographic security of other schemes in this model has been studied in [27].

In fact, it is more convenient to treat a possible attacking algorithm as an *oracle* which, given g^x and g^y returns, sometimes, some information about g^{xy}. Accordingly, our purpose is to show that having such an oracle one can recover the secret key completely.

In the sequel, to demonstrate our arguments in the simplest situation we restrict ourselves to the case of most practical interest, that is, g generating a sub-group of prime order q.

4.1 Monte Carlo Type Attacks

Given positive η and γ, we define the oracle $\mathrm{DH}^{\mathrm{MC}}_{\eta,\gamma}$ as a "black box" which, given $g^x, g^y \in \mathbb{F}_p^*$, outputs the value of $\mathrm{MSB}_{\eta,p}(g^{xy})$, with probability γ, taken over random pairs $(x, y) \in \mathbb{Z}_q^2$ (and possible internal coin-flips), and outputs an arbitrary value otherwise.

That is, $\mathrm{DH}^{\mathrm{MC}}_{\eta,\gamma}$ is a Monte Carlo type oracle which sometimes outputs some useful information and otherwise returns a wrong answer following *any* distribution. This is qualitatively thus the same type of oracles considered in [6,16].

Theorem 1. *For any $\varepsilon > 0$ such that the following statement holds. Let $\delta > 0$ be an arbitrary positive number and let*

$$\eta = \lceil \delta \log p \rceil .$$

For any element $g \in \mathbb{F}_p^*$ of multiplicative order $q \geq p^{\varepsilon}$, where q is prime, there exists a probabilistic algorithm which, in time polynomial in $\log p$ and $(0.25\gamma)^{-(5\delta^{-1}+1)} \log \gamma^{-1}$, for any pair $(a, b) \in \mathbb{Z}_q^2$, given the values of $g^a, g^b \in \mathbb{F}_p$, makes the expected number of $O\left(\delta^{-1}(0.25\gamma)^{-(5\delta^{-1}+1)} \log \gamma^{-1} \log\log p\right)$ calls to the oracle $\mathrm{DH}^{\mathrm{MC}}_{\eta,\gamma}$ and computes g^{ab} correctly with probability $1 + O\left(\log^{-1} p\right)$.

Proof. Put $d = \lceil 5 \log p/\eta \rceil \leq 5\delta^{-1} + 1$. Given g^x, g^y the oracle $\mathrm{DH}^{\mathrm{MC}}_{\eta,\gamma}$ returns $\mathrm{MSB}_{\eta,p}(g^{xy})$ with probability γ. We define an algorithm, $\mathcal{O}(g^x, g^y)$, which uses $\mathrm{DH}^{\mathrm{MC}}_{\eta,\gamma}$ as a black box and retrieves g^{xy} with non-neglible probability, ρ. We then apply a result by Shoup, [33], to this $\mathcal{O}$, and get an algorithm which retrieves g^{xy} almost surely. In the following we define and analyze $\mathcal{O}$.

By randomizing the second component input to $\mathrm{DH}^{\mathrm{MC}}_{\eta,\gamma}$, g^y, we hope to hit a set of "good" values of y, for which we have a sufficient advantage, taken over x only. We then query by randomizing the g^x-component, keeping y fixed.

Let γ_y be the average success probability of $\mathrm{DH}^{\mathrm{MC}}_{\eta,\gamma}(g^x, g^y)$, taken over random x for a given y. Thus, $\mathbb{E}_y[\gamma_y] = \gamma$. Let us define $k = \lceil \log(2/\gamma) \rceil$ and say that y is *j-good* if $\gamma_y \in [2^{-j}, 2^{-j+1})$, $j = 1, 2, \ldots, k$, and let $S_j = \{y \mid y \text{ is } j\text{-good}\}$ (thus we do not care about y for which $\gamma_y < \gamma/2$). By the Markov inequality, (2),

$$\Pr_y[\gamma_y \geq \gamma/2] \geq \frac{\gamma}{2}. \tag{5}$$

We claim that there must exist j as above for which $\Pr_y[y \in S_j] \geq 2^{j-2}\gamma/k$. If this was not the case, by (5), we would get the following contradiction:

$$\frac{\gamma}{2} \leq \sum_{j=1}^{k} 2^{-j+1} \Pr_y[y \in S_j] < \sum_{j=1}^{k} 2^{-j+1} \frac{2^{j-2}\gamma}{k} = \frac{\gamma}{2}.$$

Now, given g^a, g^b, the algorithm $\mathcal{O}$ starts by choosing a random $v \in \mathbb{Z}_q$. Next, choose d independent random elements $u_1, \ldots, u_d \in \mathbb{Z}_q$ and query the oracle $\mathrm{DH}^{\mathrm{MC}}_{\eta,\gamma}$ with g^{a+u_i} and (the fixed) g^{b+v}. After that we apply the algorithm of Lemma 3 to the obtained answers, and the value returned is finally output by $\mathcal{O}$. To analyze this, note that *if* $y = v + b$ is j-good for some j, the oracle $\mathrm{DH}^{\mathrm{MC}}_{\eta,\gamma}$ with probability 2^{-jd} returns the correct values of

$$s_i = \mathrm{MSB}_{\eta,p}\left(g^{(a+u_i)(b+v)}\right) = \mathrm{MSB}_{\eta,p}(\alpha t_i)$$

for every $i = 1, \ldots, d$, where $\alpha = g^{a(b+v)}$ and $t_i = g^{u_i(b+v)}$. Since q is prime, $t_1, \ldots, t_d$ are distinct and applying Lemma 5 we see that the algorithm of Lemma 3 then finds α (and thus also $g^{ab} = \alpha g^{-av}$) with probability $1 - 1/p$.

The above procedure performs as stated with probability at least $2^{-jd} \Pr_v[v + b \in S_j]$. As we have seen, there must be a j for which $\Pr_v[v + b \in S_j] \geq 2^{j-2}\gamma/k$, so $\mathcal{O}$ succeeds with probability at least

$$\rho = (1-p^{-1})\frac{\gamma 2^{-j(d-1)}}{4k} \geq (1-p^{-1})\frac{\gamma 2^{-k(d-1)}}{4k} \geq \frac{\gamma 2^{-(\log(2/\gamma)+1)(d-1)}}{8\log(2/\gamma)} = \frac{\gamma^d 2^{-2d-1}}{\log(2/\gamma)},$$

which is $\Omega\left((0.25\gamma)^{5\delta^{-1}+1}/\log\gamma^{-1}\right)$. The above algorithm satisfies the definition of faulty Diffie-Hellman oracle given in [33]. Therefore, applying Corollary 1 of [33] (with $\varepsilon = \rho$ and $\alpha = 1/\log p$ in the notations of [33]) we finish the proof. □

We remark that the proof of Theorem 1 only relies on the *existence* of a "good" j, but we stress that it is also possible to efficiently find this j and a corresponding v such that $v + b$ is indeed j-good. To this end, choose a random v, and query the oracle on inputs of the form g^r, g^{b+v} for random, independent r. Since r, v, and g^b are known, so is the corresponding Diffie-Hellman secret $g^{(b+v)r}$. This means that we for each r can check if the oracle is correct on this input. Repeating this for polynomially many independent r, we get a sufficiently good approximation of γ_{v+b}, on which we can base the decision on whether $v + b$ is "good" or not, see also the proof of Theorem 2.

Obviously, the algorithm of Theorem 1 remains polynomial time under the condition $\delta^{-1}\log\gamma^{-1} = O(\log\log p)$. For example, for any fixed $\delta > 0$ (that is, when η corresponds to $\Omega(\log p)$ bits) the tolerated rate of correct oracle answers can be as low as $\gamma = \Omega(\log^{-A} p)$ with some constant $A > 0$. On the other hand, if the oracle is correct with a constant rate γ, then it is enough if it outputs

$\eta = O(\log p/\log\log p)$ bits. This range can be compared to the original works in [6,16], which apply with only $O(\log^{1/2} p)$ bits from the oracle, but on the other hand requires the rate of correct answers to be $\gamma = 1 + o(1)$.

4.2 Las Vegas Type Attacks

We now turn to the more powerful type of oracles. Given positive η and A, we define the oracle $\mathrm{DH}^{\mathrm{LV}}_{\eta,A}$ as a "black box" which, given $g^x, g^y \in \mathbb{F}_p^*$, outputs the value of $\mathrm{MSB}_{\eta,p}(g^{xy})$, with probability at least $\log^{-A} p$, (taken over random pairs $(x,y) \in \mathbb{Z}_q^2$ and possible internal coin-flips), and outputs an error audit message, $\perp$, otherwise.

That is, $\mathrm{DH}^{\mathrm{LV}}_{\eta,A}$ is a Las Vegas type oracle which outputs some useful (correct) information non-negligibly often and never returns a wrong answer (but rather gives no answer at all). Again, the case of $A = 0$ quantitatively corresponds to the "error-free" oracle which has been considered in [6,16].

Theorem 2. *For any $\varepsilon > 0$ the following statement holds. Let*

$$\eta = \left\lceil \left(\frac{\tau \log p \log\log\log p}{\log\log p} \right)^{1/2} \right\rceil,$$

where $\tau > 0$ is an arbitrary absolute constant. For any element $g \in \mathbb{F}_p^*$ of multiplicative order $q \geq p^{\varepsilon}$, where q is prime, there exists a probabilistic polynomial time algorithm which for any pair $(a,b) \in \mathbb{Z}_q^2$, given the values of $g^a, g^b \in \mathbb{F}_p$, makes the expected number of $O\left((\log p)^{\max\{1,A\}} \log\log p\right)$ calls to the oracle $\mathrm{DH}^{\mathrm{LV}}_{\eta,A}$ and computes g^{ab} correctly with probability $1 + O\left(\log^{-A} p\right)$.

Proof. The proof is similar to that of Theorem 1, though for simplicity, we use a slightly rougher estimate. Let $\gamma = \log^{-A} p$ and let γ_y be as in the notation of the proof of Theorem 1.

To find $v \in \mathbb{Z}_q$ with at least $\gamma_{v+b} > \gamma/4$ choose a random $v \in \mathbb{Z}_q$. We check whether $b = \pm v \pmod q$, in which case we are done. Otherwise we choose $N = \lceil 20\gamma^{-1} \log \gamma^{-1} \rceil$ independent, random elements $u_1, \ldots, u_N \in \mathbb{Z}_q$ and query the oracle $\mathrm{DH}^{\mathrm{LV}}_{\eta,A}$ with g^{a+u_i} and g^{b+v}. If the oracle returns $K \geq \gamma N/2$ queries then $\gamma_{v+b} > \gamma/4$ with probability $1 + O(\gamma)$. Indeed, by the Chernoff bound, see for example Section 9.3 of [24], we get that if $\gamma_{v+b} < \gamma/4$ then even after

$$M = \left\lceil \frac{8}{\gamma_{v+b}} \log(2/\gamma) \right\rceil \geq N$$

the oracle returns at most $2M\gamma_{v+b} \leq K$ queries with probability at least $1 - \gamma$.

By the Markov inequality (5), we see that after the expected number of $2\gamma^{-1}$ random choices of v we find v with $\gamma_{v+b} \geq \gamma/4$ with probability $1 + O(\gamma)$.

For this v we re-use the first $d = \lfloor 5 \log p/\eta \rfloor$ replies of the oracle $\mathrm{DH}^{\mathrm{LV}}_{\eta,A}$ which have been used for testing whether $b+v$ is "good" (if $K \geq d$) or get $d-K$ additional replies (if $K < d$, which will not happen for interesting γ). Thus we get d values

$$s_i = \mathrm{MSB}_{\eta,p}\left(g^{(a+u_i)(b+v)}\right) = \mathrm{MSB}_{\eta,p}(\alpha t_i)$$

where $\alpha = g^{a(b+v)}$ and $t_i = g^{u_i(b+v)}$, $i = 1, \ldots, d$. Applying Lemma 5 we see that the algorithm of Lemma 3 finds α with probability at least $1 - 1/p$. Finally we compute $g^{ab} = \alpha g^{-av}$. □

A recent paper by Hast [19] studies an oracle model which falls somewhere in between Monte Carlo and Las Vegas oracles. Specifically, [19] considers oracles which, for some $\varepsilon, \delta \in [0,1]$, output $\perp$ with probability $1-\delta$, and where non-$\perp$ answers are correct with advantage ε. Our Las Vegas oracles thus correspond to ones with non-negligible δ and the extreme case of $\varepsilon = 1$. For the specific case of Goldreich-Levin [13] based pseudo-random bit generator, Hast [19], shows that for a given (non-negligible) success-rate, the existence of oracles with small δ would indeed be more serious than existence of traditional oracles, (for which $\delta = 1$). Perhaps not surprisingly, Theorems 1 and 2 demonstrate this for the case of the Diffie-Hellman scheme, by comparing the complexity of the respective reductions.

5 Summary

We have extended existing hardness results on the Diffie-Hellman scheme to tolerate higher error-rates in the predictions, by a trade-off on the number of bits predicted. We also studied an alternative (and much stronger) computational prediction-model whose realization, albeit less likely than more classical models, would have more severe impact on the security of Diffie-Hellman. The idea to consider Las Vegas type attacks in these setting appears to be new and definitely deservers further studying.

We remark that the analysis using Las Vegas type predictors can be applied to the study of RSA as well, for instance, when analyzing the use of RSA to send reasonably short bit strings with random padding. (In particular, results concerning Δ-homogeneous distribution generalize to composite moduli). Qualitatively, such results could of course also have been derived from the bit security results in [20]. Nevertheless, the significantly tighter reductions possible from a Las Vegas oracle show (as one would expect) that the existence of such an oracle would indeed also be *quantitatively* more severe for the security of RSA.

Of course, the most intriguing open problem remains: show that even single, individual bits of the Diffie-Hellman scheme are hard to approximate.

References

1. M. Ajtai, R. Kumar and D. Sivakumar, 'A sieve algorithm for the shortest lattice vector problem', *Proc. 33rd ACM Symp. on Theory of Comput.,* ACM, 2001, 601–610.
2. M. Bellare, A. Desai, D. Pointcheval and P. Rogaway, 'Relations among notions of security for public-key encryption schemes', *Advances in Cryptology - Proceedings of CRYPTO'98. Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **1462,** 1998, 26–46.
3. M. Bellare and P. Rogaway, 'Random oracles are practical: a paradigm for designing efficient protocols', *Proc. 1st ACM Computer and Communication Security' 93,* ACM Press, 1993, 62–73.
4. M. Blum and S. Micali, 'How to generate cryptographically strong sequences of pseudo-random bits', *SIAM J. Comp.,* **13,** 1984, 850–864.
5. D. Boneh and I. E. Shparlinski, 'On the unpredictability of bits of the elliptic curve Diffie–Hellman scheme', *Advances in Cryptology - CRYPTO 2001. Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **2139,** 2001, 201–212.
6. D. Boneh and R. Venkatesan, 'Hardness of computing the most significant bits of secret keys in Diffie–Hellman and related schemes', *Advances in Cryptology - CRYPTO '96. Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **1109,** 1996, 129–142.
7. D. Boneh and R. Venkatesan, 'Rounding in lattices and its cryptographic applications', *Proc. 8th Annual ACM-SIAM Symp. on Discr. Algorithms,* ACM, 1997, 675–681.
8. J. Bourgain and S. V. Konyagin, 'Estimates for the number of sums and products and for exponential sums over subgroups in fields of prime order', *Comptes Rendus Mathematique,* **337** (2003), 75–80.
9. R. Canetti, O. Goldreich and S. Halevi. 'The random oracle model, revisited', *Proc. 30th ACM Symp. on Theory of Comp.,* ACM, 1998, 209–218.
10. R. Cramer and V. Shoup 'Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption', *Advances in Cryptology - EUROCRYPT 2002. Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **2332,** 2002, 45–64.
11. E. El Mahassni, P. Q. Nguyen and I. E. Shparlinski, 'The insecurity of Nyberg–Rueppel and other DSA-like signature schemes with partially known nonces', *Cryptography and Lattices : International Conference, CaLC 2001. Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **2146,** 2001, 97–109.
12. R. Fischlin and C. P. Schnorr, 'Stronger security proofs for RSA and Rabin bits', *J. Cryptology,* **13,** 2000, 221–244.
13. O. Goldreich and L. A. Levin, 'A hard core predicate for any one way function', *Proc. 21st ACM Symp. on Theory of Comput.,* ACM, 1989, 25–32.
14. M. I. González Vasco and M. Näslund, 'A survey of hard core functions', *Proc. Workshop on Cryptography and Computational Number Theory,* Singapore 1999, Birkhäuser, 2001, 227–256.
15. M. I. González Vasco, M. Näslund and I. E. Shparlinski, 'The hidden number problem in extension fields and its applications', *Proc. of LATIN 2002: Theoretical Informatics : 5th Latin American Symposium. Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **2286,** 2002, 105–117.
16. M. I. González Vasco and I. E. Shparlinski, 'On the security of Diffie–Hellman bits', *Proc. Workshop on Cryptography and Computational Number Theory,* Singapore 1999, Birkhäuser, 2001, 257–268.

17. M. I. González Vasco and I. E. Shparlinski, 'Security of the most significant bits of the Shamir message passing scheme', *Math. Comp.,* **71**, 2002, 333–342.
18. M. Grötschel, L. Lovász and A. Schrijver, *Geometric algorithms and combinatorial optimization,* Springer-Verlag, Berlin, 1993.
19. G. Hast, 'Nearly one-sided tests and the Goldreich-Levin predicate', *Advances in Cryptology - EUROCRYPT 2003. Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **2656,** 2003, 195–210.
20. J. Håstad and M. Näslund, 'The security of individual RSA and discrete log bits', *J. of the ACM,* (to appear).
21. D. R. Heath-Brown and S. V. Konyagin, 'New bounds for Gauss sums derived from kth powers, and for Heilbronn's exponential sum', *Ouart. J. Math.,* **51,** 2000, 221–235.
22. N. A. Howgrave-Graham, P. Q. Nguyen and I. E. Shparlinski, 'Hidden number problem with hidden multipliers, timed-release crypto and noisy exponentiation', *Math. Comp.,* **72**, 2003, 1473–1485.
23. R. Kannan, 'Algorithmic geometry of numbers', *Annual Review of Comp. Sci.,* **2**, 1987, 231–267.
24. M. J. Kearns and U. V. Vazirani, *An introduction to computational learning theory,* MIT Press, Cambridge MA, 1994.
25. S. V. Konyagin and I. Shparlinski, *Character sums with exponential functions and their applications,* Cambridge Univ. Press, Cambridge, 1999.
26. W.-C. W. Li, M. Näslund and I. E. Shparlinski, 'The hidden number problem with the trace and bit security of XTR and LUC', *Advances in Cryptology - CRYPTO 2002. Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **2442,** 2002, 433–448.
27. M. Näslund, I. E. Shparlinski and W. Whyte, 'On the bit security of NTRU', *Proc of Public Key Cryptography - PKC 2003. Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **2567,** 2003, 62–70.
28. P. Q. Nguyen and I. E. Shparlinski, 'The insecurity of the digital signature algorithm with partially known nonces', *J. Cryptology,* **15**, 2002, 151–176.
29. P. Q. Nguyen and I. E. Shparlinski, 'The insecurity of the elliptic curve digital signature algorithm with partially known nonces', *Designs, Codes and Cryptography,* **30** (2003), 201–217.
30. P. Q. Nguyen and J. Stern, 'Lattice reduction in cryptology: An update', *Proc. of ANTS 2000. Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **1838,** 2000, 85–112.
31. P. Q. Nguyen and J. Stern, 'The two faces of lattices in cryptology', *Proc. of CalC 2001. Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **2146** (2001), 146–180.
32. C. P. Schnorr, 'Security of almost all discrete log bits', *Electronic Colloq. on Comp. Compl.,* Univ. of Trier, **TR98-033,** 1998, 1–13.
33. V. Shoup, 'Lower bounds for discrete logarithms and related problems', *Preprint,* available from `http://www.shoup.net`.
34. I. E. Shparlinski, 'Security of most significant bits of g^{x^2}', *Inform. Proc. Letters,* **83**, 2002, 109–113.
35. I. E. Shparlinski, *Cryptographic applications of analytic number theory,* Birkhauser, 2003.

Short Exponent Diffie-Hellman Problems

Takeshi Koshiba[1,2] and Kaoru Kurosawa[3]

[1] Secure Computing Lab., Fujitsu Laboratories Ltd.
[2] ERATO Quantum Computation and Information Project,
Japan Science and Technology Agency,
Matsuo Bldg.2F, 406 Iseyacho, Kawaramachi Marutamachi,
Kamigyo-ku, Kyoto 602-0873, Japan
koshiba@acm.org
[3] Department of Computer and Information Sciences, Ibaraki University,
4-12-1 Nakanarusawa, Hitachi, Ibaraki 316-8511, Japan
kurosawa@cis.ibaraki.ac.jp

Abstract. In this paper, we study *short* exponent Diffie-Hellman problems, where significantly many lower bits are zeros in the exponent. We first prove that the decisional version of this problem is as hard as two well known hard problems, the standard decisional Diffie-Hellman problem (DDH) and the short exponent discrete logarithm problem. It implies that we can improve the efficiency of ElGamal scheme and Cramer-Shoup scheme under the two *widely accepted* assumptions. We next derive a similar result for the computational version of this problem.

1 Introduction

The discrete logarithm (DL) problem and the Diffie-Hellman (DH) problems are basis of many applications in modern cryptography.

1.1 Previous Works on DL Problem

Blum and Micali [1] presented the first cryptographically secure pseudo-random bit generators (PRBG) under the DL assumption over Z_p^*, where p is a prime. Long and Wigderson [6], and Peralta [9] showed that up to $O(\log \log p)$ pseudo-random bits can be extracted by a single modular exponentiation of the Blum-Micali generator.

The discrete logarithm with short exponent (DLSE) assumption is also useful. It claims that the DL problem is still hard even if the exponent is small. Van Oorschot and Wiener studied under what condition the DLSE assumption remains difficult (Their concern was to speed-up the key agreement method of Diffie-Hellman) [12]. They showed that the known attacks are precluded if safe primes p are used for Z_p^* (that is, $p - 1 = 2q$ for a prime q) or prime-order groups are used. Especially, the latter is highly recommended. Under the DLSE assumption, Patel and Sundaram [8] showed that it is possible to extract up to $n - \omega(\log n)$ bits from one iteration of the Blum-Micali generator by using safe

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 173–186, 2004.

primes p, where n is the bit length of p. Gennaro [4] further improved this result in such a way that each full modular exponentiation can be replaced with a short modular exponentiation.

1.2 Our Contribution on DH Problems

Let G_q be a finite Abelian group of prime order q. Let g be a generator, that is, $G_q = \langle g \rangle$. Then the computational Diffie-Hellman (CDH) problem is to compute g^{ab} from (g, g^a, g^b). The decisional Diffie-Hellman (DDH) problem is to distinguish between (g, g^a, g^b, g^{ab}) and (g, g^a, g^b, g^c), where a, b and c are uniformly and randomly chosen from Z_q.

	Short exp ≈ Full exp	Short DL
$\lvert Z_p^* \rvert$ = even	Gennaro [4]	Application to PRBG [8,4]

Table 1. Previous works over Z_p^*

	Short ≈ Full	Short exp. DDH	Short exp. CDH
$\lvert G_q \rvert$ = prime	This paper	DDH+Short DL Application to encryption	CDH+Short DL Application to OT

Table 2. Our work over G_q

In this paper, we study *short* exponent variants of the DDH problem and the CDH problem over G_q, where significantly many lower bits are zeros in the exponent. More precisely, the short exponent DDH problem has two sub-problems, a (Short, Full)-DDH problem in which a is small, and a (Short, Short)-DDH problem in which both a and b are small. The short exponent CDH problem has two sub-problems, similarly.

We first prove that each of the short exponent DDH problems is as hard as two well known hard problems, the standard DDH problem and the DLSE problem. That is, we show our equivalence:

$$\text{(Short, Full)-DDH} \Longleftrightarrow \text{(Short, Short)-DDH} \Longleftrightarrow DDH + DLSE.$$

To prove these equivalence, we show that short exponents $\{g^s \mid s \text{ is small}\}$ and full exponents $\{g^x \mid x \in Z_q\}$ are indistinguishable under the DLSE assumption over prime-order groups G_q. A similar result was proved for Z_p^* by Gennaro [4] based on [8], where p is a safe prime. Our proof shows that the indistinguishability can be proved much simpler over G_q than over Z_p^*. (Remember that prime-order groups are highly recommended for the DLSE assumption by van Oorschot and Wiener [12]. It is also consistent with the DDH problem which is defined over prime-order groups.)

Our result implies that we can improve the efficiency of ElGamal encryption scheme and Cramer-Shoup encryption scheme directly under the two widely accepted assumptions, the DDH assumption and the DLSE assumption. Indeed, we present such variants of ElGamal scheme and Cramer-Shoup scheme. They are much faster than the original encryption algorithms because short exponents are

used instead of full exponents. (Remember that under the DDH assumption, ElGamal encryption scheme [3] is secure in the sense of indistinguishability against chosen plaintext attack (IND-CPA) and Cramer-Shoup scheme [2] is secure in the sense of indistinguishability against chosen ciphertext attack (IND-CCA).)

We next show a similar result for the CDH problem. That is, we prove the equivalence such that

$$\text{(Short, Full)-CDH} \Longleftrightarrow \text{(Short, Short)-CDH} \Longleftrightarrow CDH + DLSE.$$

This result implies that we can improve the efficiency of the oblivious transfer protocols of [7,5] under the CDH assumption plus the DLSE assumption.

We believe that there will be many other applications of our results.

2 Preliminaries

2.1 Notation

$|x|$ denotes the bit length of x. $x \in_R X$ means that x is randomly chosen from a set X. We sometimes assume the uniform distribution over X. Throughout the paper, an "efficient algorithm" means a probabilistic polynomial time algorithm.

Let n denote the bit length of q, where q is the prime order of G_q. Let $c = \omega(\log n)$. It means that 2^c grows faster than any polynomial in n. Let $lsb_k(z)$ be the function that returns the least significant k bits of z and $msb_k(z)$ the function that returns the most significant k bits of z. If we write $b = msb_k(z)$, we sometimes mean that the binary representation of b is $msb_k(z)$.

2.2 Discrete Logarithm with Short Exponent (DLSE) Assumption

Let $f(g, z) = (g, g^z)$, where g is a generator of G_q. The discrete logarithm (DL) problem is to compute the inverse of f. The DL assumption says that the DL problem is hard.

We next define the discrete logarithm with short exponent (DLSE) problem as follows. Let $f^{\text{se}}(g, u||0^{n-c}) = (g, g^{u||0^{n-c}})$, where $|u| = c$ and $||$ denotes concatenation. That is, the exponent of $g^{u||0^{n-c}}$ is short. Then the DLSE problem is to compute the inverse of f^{se}. The DLSE assumption says that the DLSE problem is hard. Formally,

Assumption 1. *(DLSE assumption)* There exists no efficient algorithm which solves the DLSE problem with non-negligible probability.

3 Short EXP ≈ Full EXP

In this section, we prove that full exponents and short exponents are indistinguishable under the DLSE assumption. More formally, define A_0 and A_{n-c} as

$$A_0 = \{(g, g^x) \mid x \in R_0\} \quad \text{and} \quad A_{n-c} = \{(g, g^x) \mid x \in R_{n-c}\},$$

where

$$R_0 = \{u \mid 0 \leq u < q\} \quad \text{and} \quad R_{n-c} = \{2^{n-c}u \mid 0 \leq 2^{n-c}u < q\}.$$

Theorem 1. *A_0 and A_{n-c} are indistinguishable under the DLSE assumption.*

A proof is given in Appendix A. We show a sketch of the proof here. For $1 \leq i \leq n-c$, let

$$A_i = \{(g, g^x) \mid x \in R_i\}, \text{ where } R_i = \{2^i u \mid 0 \leq 2^i u < q\}.$$

Suppose that there exists a distinguisher D which can distinguish A_{n-c} from A_0. Then by using a hybrid argument, there exists j such that A_j and A_{j+1} are distinguishable.

We will show that (i) the j can be found in polynomial time and (ii) the DLSE problem can be solved by using the (D, j). We briefly sketch below how to solve the DLSE problem by using the (D, j). (Remember that the DLSE problem is to find x from (g, g^x) in A_{n-c}.)

1. The difference between A_j and A_{j+1} appears in the $(j+1)$-th least significant bit b_{j+1} of exponents x. That is,
$$b_{j+1} = \begin{cases} 1 \text{ if } (g, g^x) \in A_j \setminus A_{j+1} \\ 0 \text{ if } (g, g^x) \in A_{j+1} \end{cases}$$
Hence we can show that (D, j) can be used as a prediction algorithm of b_{j+1}.
2. We can compute $g^{x/2}$ from g^x because the order of G_q is a prime q. This enables us to use (D, j) to predict all higher bits of x as well as b_{j+1} (except several most significant bits γ).
3. Suppose that $(g, y) \in A_{n-c}$ is given, where $y = g^{v||0^{n-c}}$. In order to find the $b_1 = lsb_1(v)$, we carefully randomize y so that the exponent is uniformly distributed over R_j. For this randomization, we need to search some most significant bits γ of v exhaustively, but in polynomial time.
4. After all, by taking the majority vote, we can find $b_1 = lsb_1(v)$ with overwhelming probability. Next let
$$y_1 = (y(g^{2^{n-c}})^{-b_1})^{1/2} = g^{0||v'||0^{n-c}},$$
where $v = v'||b_1$. Applying the same process, we can find $lsb_1(v')$ similarly. By repeating this algorithm, we can finally find v with overwhelming probability.

4 (*Short*, Full)-DDH = Standard DDH + DLSE

The standard DDH assumption claims that

$$B_0 = \{(g, g^x, g^y, g^{xy}) \mid x \in Z_q, y \in Z_q\} \quad \text{and}$$
$$C_0 = \{(g, g^x, g^y, g^z) \mid x \in Z_q, y \in Z_q, z \in Z_q\}$$

are indistinguishable.

We now define the (Short, Full)-DDH assumption as follows. Let

$$B_{n-c} = \{(g, g^x, g^y, g^{xy}) \mid x \in R_{n-c}, y \in Z_q\} \quad \text{and}$$
$$C_{n-c} = \{(g, g^x, g^y, g^z) \mid x \in R_{n-c}, y \in Z_q, z \in Z_q\},$$

where $c = \omega(\log n)$ with $n = |q|$. The (Short, Full)-DDH assumption claims that B_{n-c} and C_{n-c} are still indistinguishable. Note that x is short and y is of full length.

We then prove the (Short, Full)-DDH assumption is equivalent to the standard DDH assumption and the DLSE assumption. We first show that the standard DDH assumption and the DLSE assumption implies the (Short, Full)-DDH assumption.

Theorem 2. *Suppose that the DDH assumption and the DLSE assumption are true. Then the (Short, Full)-DDH assumption is true.*

Proof. From Theorem 1, A_0 and A_{n-c} are indistinguishable under the DLSE assumption, where

$$A_0 = \{(g, g^x) \mid x \in R_0\} \quad \text{and} \quad A_{n-c} = \{(g, g^x) \mid x \in R_{n-c}\}.$$

First it is clear that C_0 and C_{n-c} are indistinguishable because y and z are random independently of x.

Next we prove that B_0 and B_{n-c} are indistinguishable. Suppose that there exists a distinguisher D which distinguishes B_{n-c} from B_0. Then we show that there exists a distinguisher D' which distinguishes A_{n-c} from A_0. On input (g, g^x), D' chooses $y \in Z_q$ at random and computes g^y and $(g^x)^y$. D' then gives $(g, g^x, g^y, (g^x)^y)$ to D. Note that

$$(g, g^x, g^y, (g^x)^y) \in_R \begin{cases} B_0 & \text{if } (g, g^x) \in_R A_0, \\ B_{n-c} & \text{if } (g, g^x) \in_R A_{n-c}. \end{cases}$$

D' finally outputs the output bit of D. Then it is clear that D' can distinguish A_{n-c} from A_0. However, this is against Theorem 1. Hence B_0 and B_{n-c} are indistinguishable.

Consequently we obtain that $B_{n-c} \approx B_0 \approx C_0 \approx C_{n-c}$, where $\approx$ means indistinguishable. ($B_0 \approx C_0$ comes from the standard DDH assumption.) Therefore, B_{n-c} and C_{n-c} are indistinguishable. □

We next show that the (Short, Full)-DDH assumption implies the standard DDH assumption and the DLSE assumption.

Theorem 3. *Suppose that the (Short, Full)-DDH assumption is true. Then the DDH assumption and the DLSE assumption are true.*

Proof. First suppose that there exists an efficient algorithm M which can solve the DLSE problem with some non-negligible probability ϵ. Then we show that there exists a distinguisher D between B_{n-c} and C_{n-c}.

On input (g, g^x, g^y, α), D gives g^x to M. If M does not output x correctly, then D outputs a random bit b. Suppose that M outputs x correctly. Then D outputs b such that

$$b = \begin{cases} 1 \text{ if } \alpha = (g^y)^x \\ 0 \text{ if } \alpha \neq (g^y)^x. \end{cases}$$

Then it is easy to see that D distinguishes between B_{n-c} and C_{n-c}.

Next suppose that there exists a distinguisher D_0 which breaks the DDH assumption. Then we show that there exists a distinguisher D_1 which breaks the (Short, Full)-DDH assumption.

Let (g, g^x, g^y, g^a) be an input to D_1, where $a = xy \bmod q$ or random. D_1 chooses $r \neq 0$ at random and gives $(g, (g^x)^r, g^y, (g^a)^r)$ to D_0. It is easy to see that

$$(g, (g^x)^r, g^y, (g^a)^r) \in_R \begin{cases} B_0 \text{ if } (g, g^x, g^y, g^a) \in_R B_{n-c}, \\ C_0 \text{ if } (g, g^x, g^y, g^a) \in_R C_{n-c}. \end{cases}$$

Finally D_1 outputs the output bit of D_0. Then it is clear that D_1 distinguishes between B_{n-c} and C_{n-c}. □

From Theorem 2 and Theorem 3, we obtain the following corollary.

Corollary 1. *The (Short, Full)-DDH assumption is equivalent to both the DDH assumption and the DLSE assumption.*

5 Extension to (Short, *Short*)-DDH

We define the (Short, *Short*)-DDH assumption as follows. Let

$$B'_{n-c} = \{(g, g^x, g^y, g^{xy}) \mid x \in R_{n-c}, y \in R_{n-c}\} \quad \text{and}$$
$$C'_{n-c} = \{(g, g^x, g^y, g^z) \mid x \in R_{n-c}, y \in R_{n-c}, z \in Z_q\}.$$

Then the (Short, *Short*)-DDH assumption claims that B'_{n-c} and C'_{n-c} are indistinguishable. Note that both x and y are short in B'_{n-c} and C'_{n-c}.

We first show that the (Short, Full)-DDH assumption implies the (Short, Short)-DDH assumption.

Theorem 4. *Suppose that the (Short, Full)-DDH assumption is true. Then the (Short, Short)-DDH assumption is true.*

Proof. First suppose that the (Short, Full)-DDH assumption is true. From Theorem 3, both the DLSE assumption and the DDH assumption are true. From Theorem 1, A_0 and A_{n-c} are indistinguishable. Then it is clear that C_{n-c} and C'_{n-c} are indistinguishable because x and z are random independently of y.

Next we prove that B_{n-c} and B'_{n-c} are indistinguishable. Suppose that there exists a distinguisher D which distinguishes B_{n-c} and B'_{n-c}. Then we show that there exists a distinguisher D' which distinguishes A_{n-c} from A_0. On input

(g, g^y), D' chooses $x \in R_{n-c}$ at random and computes g^x and $(g^y)^x$. D' then gives $(g, g^x, g^y, (g^y)^x)$ to D. Note that

$$(g, g^x, g^y, (g^x)^y) \in_R \begin{cases} B_{n-c} \text{ if } (g, g^x) \in_R A_0, \\ B'_{n-c} \text{ if } (g, g^x) \in_R A_{n-c}. \end{cases}$$

D' finally outputs the output bit of D. Then it is clear that D' can distinguish A_{n-c} from A_0. However, this contradicts that A_0 and A_{n-c} are indistinguishable. Hence B_{n-c} and B'_{n-c} are indistinguishable.

Consequently we obtain that

$$B'_{n-c} \approx B_{n-c} \approx C_{n-c} \approx C'_{n-c},$$

where $\approx$ means indistinguishable. Therefore, B'_{n-c} and C'_{n-c} are indistinguishable. □

We next show that the (Short, Short)-DDH assumption implies the (Short, Full)-DDH assumption.

Theorem 5. *Suppose that the (Short, Short)-DDH assumption is true. Then the (Short, Full)-DDH assumption is true.*

Proof. First suppose that the (Short, Full)-DDH assumption is false. Then, from Theorem 2, either the DDH assumption or the DLSE assumption is false.

Further suppose that the DLSE assumption is false. That is, there exists an efficient algorithm M which can solve the DLSE problem with some non-negligible probability ϵ. Then we show that there exists a distinguisher D between B'_{n-c} and C'_{n-c}.

On input (g, g^x, g^y, α), D gives g^x to M. If M does not output x correctly, then D outputs a random bit b. Suppose that M outputs x correctly. Then D outputs b such that

$$b = \begin{cases} 1 \text{ if } \alpha = (g^y)^x \\ 0 \text{ if } \alpha \neq (g^y)^x. \end{cases}$$

Then it is easy to see that D distinguishes between B'_{n-c} and C'_{n-c}.

Next suppose that the DDH assumption is false. That is, there exists a distinguisher D_0 which breaks the DDH assumption. Then we show that there exists a distinguisher D_1 which breaks the (Short, Short)-DDH assumption.

Let (g, g^x, g^y, g^a) be an input to D_1, where $a = xy \bmod q$ or random. D_1 chooses $r_1, r_2 \neq 0$ at random and gives $(g, (g^x)^{r_1}, (g^y)^{r_2}, (g^a)^{r_1 r_2})$ to D_0. It is easy to see that

$$(g, (g^x)^{r_1}, (g^y)^{r_2}, (g^a)^{r_1 r_2}) \in_R \begin{cases} B_0 \text{ if } (g, g^x, g^y, g^a) \in_R B'_{n-c}, \\ C_0 \text{ if } (g, g^x, g^y, g^a) \in_R C'_{n-c}. \end{cases}$$

Finally D_1 outputs the output bit of D_0. Then it is clear that D_1 distinguishes between B'_{n-c} and C'_{n-c}. □

Corollary 2. *The (Short, Short)-DDH assumption is equivalent to the (Short, Full)-DDH assumption.*

From Corollary 1, we obtain the following corollary.

Corollary 3. *The (Short, Short)-DDH assumption is equivalent to both the DDH assumption and the DLSE assumption.*

6 Short Computational DH

Remember that the computational Diffie-Hellman (CDH) problem is to compute g^{xy} from g, g^x, g^y, where $x, y \in Z_q$. The CDH assumption says that the CDH problem is hard.

In this section, we introduce two variants of the CDH assumption, (Short, Full)-CDH assumption and (Short, Short)-CDH assumption. We then prove that each of them is equivalent to the standard CDH assumption and the DLSE assumption.

Short variants of the CDH assumption are defined as follows.

Assumption 2. *((Short, Full)-CDH assumption)* There exists no efficient algorithm for computing g^{xy} with non-negligible probability from g, g^x, g^y, where $x \in R_{n-c}$ and $y \in Z_q$.

Assumption 3. *((Short, Short)-CDH assumption)* There exists no efficient algorithm for computing g^{xy} with non-negligible probability from g, g^x, g^y, where $x \in R_{n-c}$ and $y \in R_{n-c}$.

We first show that the standard CDH assumption and the DLSE assumption imply the (Short, Full)-CDH assumption.

Theorem 6. *Suppose that the CDH assumption and the DLSE assumption are true. Then the (Short, Full)-CDH assumption is true.*

Proof. Suppose that there exists an efficient algorithm A which computes g^{xy} from g, g^x, g^y such that $x \in R_{n-c}$ and $y \in Z_q$.
If the CDH problem is easy, then our claim holds. Suppose that the CDH problem is hard. We then show an efficient algorithm B which distinguishes between A_0 and A_{n-c}. On input (g, g^x), B chooses $y \in Z_q$ randomly and computes g^y. B gives (g, g^x, g^y) to A. Suppose that A outputs z. B checks if $z = (g^x)^y$.

Now from our assumption, if $(g, g^x) \in A_{n-c}$, then $z = g^{xy}$ with non-negligible probability. If $(g, g^x) \in A_0$, then $z = g^{xy}$ with negligible probability. This means that B can distinguish between A_0 and A_{n-c}. From Theorem 1, this means that the DLSE assumption is false. □

We can prove the converse of Theorem 6 similarly to Theorem 3. Therefore, we obtain the following corollary.

Corollary 4. *(Short, Full)-CDH = Standard CDH + DLSE.*

We next show that the (Short, Short)-CDH assumption is equivalent to the standard CDH assumption and the DLSE assumption.

Theorem 7. *(Short, Short)-CDH = Standard CDH + DLSE.*

The proof is based on the same argument for the (Short, Full)-CDH assumption and the random self-reducibility of the discrete logarithm problem. The details will be given in the final paper.

7 Applications

In this section, we present fast variants of ElGamal encryption scheme and Cramer-Soup encryption scheme. Each variant uses a short random exponent r such that r is essentially c bits long, where $c = \omega(\log |q|)$ and q is the order of the underlying group.

Note that computing g^r requires at most $2c$ modulo multiplications in our variants while it requires at most $2n$ modulo multiplications in the original algorithms. Hence our variants are much faster than the original encryption algorithms.

We can prove their security easily from our results. They are semantically secure under the DDH assumption and the DLSE assumption (i.e., our variant of ElGamal scheme is IND-CPA and our variant of Cramer-Shoup scheme is IND-CCA, respectively). They are one-way under the CDH assumption and the DLSE assumption.

7.1 Security of Public Key Cryptosystem

A public key encryption scheme is called one-way if it is hard to compute the message m from a public key pk and a ciphertext C.

The security in the sense of indistinguishability is defined as follows. Consider the following model of adversaries. In the find stage, the adversary chooses two messages m_0, m_1 on input pk. She then sends these to an encryption oracle. The encryption oracle chooses a random bit b, and encrypts m_b. In the guess stage, the ciphertext C_b is given to the adversary. The adversary outputs a bit b'. We say that the public key cryptosystem is secure in the sense of indistinguishability against chosen plaintext attack (IND-CPA) if $|\Pr(b' = b) - 1/2|$ is negligibly small (as a function of the security parameter).

The security against chosen-ciphertext attack (IND-CCA) is defined similarly except for that the adversary gets the decryption oracle and is allowed to query any ciphertext C, where it must be $C \neq C_b$ in the guess stage.

7.2 (Short, Full) ElGamal Encryption Scheme

ElGamal encryption scheme is (1) one-way under the CDH assumption and (2) IND-CPA under the DDH assumption. Now our variant of ElGamal encryption scheme is described as follows.

(Key generation) Choose a generator G_q and $x \in Z_q$ randomly. Let $\hat{g} = g^{2^{n-c}}, \hat{y} = \hat{g}^x$. The public key is $(\hat{g}, \hat{y})$ and the secret key is x.

(Encryption) Given a message $m \in G$, first choose r such that $2^{n-c}r \in R_{n-c}$ randomly. Next compute $c_1 = \hat{g}^r \ (= g^{2^{n-c}r}), c_2 = m\hat{y}^r$. The ciphertext is (c_1, c_2).

(Decryption) Given a ciphertext (c_1, c_2), compute

$$c_2/c_1^x = m\hat{y}^r/(g^{2^{n-c}r})^x = m(g^{2^{n-c}x})^r/(g^{2^{n-c}r})^x = m.$$

Note that the encryption is very efficient because small r is used. The security is proved as follows.

Theorem 8. *The above scheme is still one-way under the CDH assumption and the DLSE assumption.*

Theorem 9. *The above scheme is still IND-CPA under the DDH assumption and the DLSE assumption.*

7.3 (Short, Full) Cramer-Shoup Encryption Scheme

We next show our variant of Cramer-Shoup scheme.

(Key generation) Choose two generator g_1 and g_2 at random. Also Choose $x_1x_2, y_1, y_2, z \in Z_q$ randomly. Let $\hat{g}_1 = g_1^{n-c}$ and $\hat{g}_2 = g_2^{n-c}$. Also let

$$\hat{c} = \hat{g}_1^{x_1}\hat{g}_2^{x_2}, \ \hat{d} = \hat{g}_1^{y_1}\hat{g}_2^{y_2}, \ \hat{h} = \hat{g}_1^{z}.$$

The public key is $(\hat{g}_1, \hat{g}_2, \hat{c}, \hat{d}, \hat{h}, H)$ and the secret key is (x_1x_2, y_1, y_2, z), where H is a randomly chosen universal one-way hash function.

(Encryption) Given a message $m \in G$, first choose r such that $2^{n-c}r \in R_{n-c}$ randomly. Next compute

$$u_1 = \hat{g}_1^r, u_2 = \hat{g}_2^r, e = \hat{h}^r m, \alpha = H(u_1, u_2, e), v = (\hat{c}\hat{d}^{\alpha})^r.$$

The ciphertext is (u_1, u_2, e, v).

(Decryption) Given a ciphertext (u_1, u_2, e, v), first compute $\alpha = H(u_1, u_2, e)$ and test if $u_1^{x_1+y_1\alpha}u_2^{x_2+y_2\alpha} = v$. If this condition does not hold, the decryption algorithm outputs "reject". Otherwise, it outputs $m = e/u_1^z$.

The encryption algorithm is very efficient because small r is used. Cramer-Shoup scheme is IND-CCA under the DDH assumption [2]. The proposed scheme is secure under the following assumption.

Theorem 10. *The above scheme is still IND-CCA under the DDH assumption and the DLSE assumption.*

The proof is almost the same as the proof of [2]. We use Corollary 1. The details will be given in the final paper.

7.4 (Short, Short) Versions

We can construct (Short, Short) versions of ElGamal scheme and Cramer-Shoup scheme, and prove their security. The details will be given in the final paper.

References

1. M. Blum and S. Micali: "How to generate cryptographically strong sequences of pseudo-random bits", SIAM J. Computing 13(4), pp.850–864 (1984)
2. R. Cramer and V. Shoup: "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack", Advances in Cryptology – Crypto'98 Proceedings, Lecture Notes in Computer Science 1462, Springer, pp.13–25 (1998)
3. T. ElGamal: " A public key cryptosystem and a signature scheme based on discrete logarithms", IEEE Trans. Information Theory IT-31(4), pp.469–472 (1985)
4. R. Gennaro: "An improved pseudo-random generator based on discrete log", Advances in Cryptology – Crypto 2000 Proceedings, Lecture Notes in Computer Science 1880, Springer, pp.469–481 (2000)
5. K. Kurosawa and Q. V. Duong: "How to design efficient multiple-use 1-out-n oblivious transfer", IEICE Trans. Fundamentals E87A(1), (2004)
6. D. L. Long and A. Wigderson: "The discrete logarithm hides $O(\log n)$ bits", SIAM J. Computing 17(2), pp.363–372 (1988)
7. M. Naor and B. Pinkas: "Efficient oblivious transfer protocols," Proc. the 12th ACM-SIAM Symposium on Discrete Algorithms (SODA), pp.448–457 (2001)
8. S. Patel and G. S. Sundaram: "An efficient discrete log pseudo random generator", Advances in Cryptology – Crypto'98 Proceedings, Lecture Notes in Computer Science 1462, Springer, pp.304–317 (1998)
9. R. Peralta: "Simultaneous security of bits in the discrete log", Advances in Cryptology – Eurocrypt'85 Proceedings, Lecture Notes in Computer Science 219, Springer, pp.62–72 (1986)
10. J. M. Pollard: "Kangaroos, monopoly and discrete logarithms", J. Cryptology 13(4), pp.437–447 (2000)
11. C. Schnorr: "Security of almost all discrete log bits", Electronic Colloquium on Computational Complexity. TR-98-033. http:/www.eccc.uni-trier.de.eccc/
12. P. C. van Oorschot and M. J. Wiener: "On Diffie-Hellman key agreement with short exponents", Advances in Cryptology – Eurocrypt'96 Proceedings, Lecture Notes in Computer Science 1070, Springer, pp.332–343 (1996)
13. P. C. van Oorschot and M. J. Wiener: "Parallel Collision Search with Cryptographic Applications", J. Cryptology 12(1), pp.1–28 (1999)

Appendix

A Proof of Theorem 1

Before giving a proof of Theorem 1, we show some technical lemmas. Remember that $c = \omega(\log n)$.

Lemma 1. We consider an index i which can be computed in probabilistic polynomial time. Suppose that there exists an efficient algorithm D that on input $(g, g^{u||0^i}) \in_R A_i$, outputs the *lsb* of u with probability $1/2 + \epsilon$, where ϵ is non-negligible.Then for any fixed $g \in G$,there exists an efficient algorithm that on input $g^{u||0^i}$, outputs the *lsb* of u with probability $1/2 + \epsilon$, where $u||0^i \in_R R_i$.

Lemma 1 is easily obtained from the random self-reducibility such that computing z from (g, g^z) is equivalent to computing z from (g^r, g^{rz}). Next let g be a generator of G_q.

Lemma 2. We consider an index i such that $i < n - c$ which can be computed in probabilistic polynomial time. Suppose that there exists an efficient algorithm D that on input g and $g^{u||0^i}$, outputs the *lsb* of u with probability $1/2+\epsilon$, where $u||0^i \in_R R_i$ and ϵ is non-negligible.

Then there exists an efficient algorithm D' that on input $y = g^{v||0^{n-c}}$ and $msb_{\log t}(v)$, outputs the *lsb* of v with probability at least $1/2 + \epsilon - (2/t)$.

Proof. Let D be an efficient algorithm as stated above. We construct an efficient algorithm D' that, given g and $g^{v||0^{n-c}}$ and $msb_{\log t}(v)$, outputs the *lsb* of v with probability at least $1/2 + \epsilon - (2/t)$. Let $\gamma = msb_{\log t}(v)$. That is, $v = \gamma||v'$ for some v'. We will find the *lsb* of v' by using D (because $lsb_1(v) = lsb_1(v')$).

(1) First, D' zeros the $\log t$ most significant bits of v by computing

$$y_1 = y \cdot g^{-\gamma \cdot 2^{n-\log t}} = g^{0^{\log t}||v'||0^{n-c}}.$$

(2) Next D' computes

$$y_2 = y_1^e, \text{ where } e = 1/2^{n-c-i} \bmod q.$$

Note that the exponent of y_1 is shifted to the right $n - c - i$ bits. Therefore, y_2 is written as $y_2 = g^s$ in such a way that

$$s = 0^{n-c-i+\log t}||v'||0^i.$$

(3) D' chooses $r \in R_i$ randomly and computes

$$y' = y_2 \cdot g^r = g^{s+r}.$$

(Note that $r = 2^i r'$ for some r' since $r \in R_i$.)

(4) D' invokes D with input (g, y').

(5) Suppose that D outputs a bit α. (If D outputs neither 0 nor 1, D' chooses a bit α randomly.) Then D' outputs $\beta = \alpha \oplus lsb_1(r')$.

Let $u = s + r$. Then u is uniformly distributed over $\{s' : s \leq s' \leq s + r_{max} \text{ and } 2^i | s'\}$, where r_{max} is the maximum element of R_i. Since $2^i \mid u$, we let $u' = u/2^i$. Then

$$u' = v' + r'.$$

If $u < q$ and $\alpha = lsb_1(u')$, then

$$\alpha = lsb_1(u') = lsb_1(v) \oplus lsb_1(r').$$

Hence

$$lsb_1(v) = \alpha \oplus lsb_1(r') = \beta (= \text{the output of } D').$$

Therefore,

$$\begin{aligned} \Pr(D' \text{ suceeds}) &\geq \Pr(u < q \text{ and } \alpha = lsb_1(u')) \\ &= \Pr(u < q \text{ and } D(g, g^u) = lsb_1(u')) \end{aligned}$$

For a fixed random tape C of D, let

$$GOOD(C) = \{x \mid x \in R_i,\ D(g, g^x) = lsb_1(x/2^i)\}$$

(It is clear that $2^i \mid x$ for $x \in R_i$.) Then

$$\begin{aligned}\Pr(D' \text{ suceeds}) &\geq \Pr(u < q \text{ and } D(g, g^u) = lsb_1(u')) \\ &= E_C[\Pr(u < q \text{ and } u \in GOOD(C))]\end{aligned}$$

where E_C denotes the expected value over C.

It is easy to see that "$u < q$ and $u \in GOOD(C)$)" is equivalent to $u \in GOOD(C)$. Therefore,

$$\Pr(D' \text{ suceeds}) \geq E_C[\Pr(u \in GOOD(C))]$$

Further, since u is uniformly distributed over $\{s' : s \leq s' \leq s + r_{max} \text{ and } 2^i | s'\}$, we obtain

$$\begin{aligned}E_C[\Pr(u \in GOOD(C))] &\geq E_C[\Pr_{y \in R_i}(y \in GOOD(C)) - \Pr_{y \in R_i}(y < s)] \\ &\geq E_C[\Pr_{y \in R_i}(y \in GOOD(C))] - E_C[\Pr_{y \in R_i}(y < s)] \\ &\geq \Pr_{y \in R_i}(y \in GOOD(C)) - \Pr_{y \in R_i}(y < s) \\ &\geq 1/2 + \epsilon - 2/t.\end{aligned}$$

Consequently,

$$\Pr(D' \text{ suceeds}) \geq 1/2 + \epsilon - 2/t.$$

□

Lemma 3. *In Lemma 2, let $t = 4/\epsilon$. Then there exists an efficient algorithm that on input $g, y = g^{v||0^{n-c}}$ and $msb_{\log t}(v)$, outputs v with overwhelming probability.*

Proof. In Lemma 2, D' outputs $lsb_1(v)$ with probability at least $1/2+\epsilon/2$ because $t = 4/\epsilon$. Here $\epsilon/2$ is non-negligible from the assumption of Lemma 2. Then by running D' polynomially many times (i.e., $2/\epsilon^3$ times) independently and taking the majority vote, we can obtain $b_1 = lsb_1(v)$ with overwhelming (i.e., $e^{-1/\epsilon}$) probability.

Next let

$$y_1 = (y(g^{2^{n-c}})^{-b_1})^{1/2} = g^{0||v'||0^{n-c}},$$

where $v = v'||b_1$. Applying the same process, we can find $lsb_1(v')$ similarly. By repeating this algorithm, we can find v with overwhelming probability. □

Now, we are ready to prove Theorem 1.

Proof. Suppose that A_0 and A_{n-c} are distinguishable. Then we will show that we can solve the DLSE problem. Assume that there exists a distinguisher D between A_0 and A_{n-c}, namely,

$$|\Pr[D(A_0) = 1] - \Pr[D(A_{n-c}) = 1]| > \frac{1}{p(n)}$$

for infinitely many n for some polynomial $p(\cdot)$. (A_0 and A_{n-c} in the above equation denote the uniform distribution over the set A_0 and A_{n-c}, respectively.) Then, for some j such that $0 \leq j \leq n-c-1$,

$$|\Pr[D(A_j)=1] - \Pr[D(A_{j+1})=1]| > \frac{1}{np(n)}. \tag{1}$$

We first show that we can find such an index j in polynomial time.

Let $p_i = \Pr[D(A_i)=1]$ for $0 \leq i \leq n-c-1$. We estimate each p_i by the sampling method of m experiments. Let $\hat{p}_i$ denote the estimated value. By using the Chernoff bound, we can show that

$$\Pr[|\hat{p}_i - p_i| > 1/8np(n)] \leq 2e^{-2m/64(np(n))^2}.$$

In other words, we can estimate all p_i with accuracy $\pm 1/8np(n)$ with high probability by using $m = 2048n^3(p(n))^2$ random samples. This means that we have, for the j of eq.(1),

$$|\hat{p}_{j+1} - \hat{p}_j| > 1/np(n) - 2/8np(n) = 3/4np(n).$$

Therefore, there exists at least one j which satisfies the above equation.

Our algorithm first finds an index i such that

$$|\hat{p}_{i+1} - \hat{p}_i| > 3/4np(n),$$

by using $\hat{p}_i$. For this i, we see that

$$|p_{i+1} - p_i| > 1/2np(n) \tag{2}$$

by using the same argument as above.

We next show that D can be used as a prediction algorithm of Lemma 1. Wlog, we assume that $p_i - p_{i+1} > 1/2np(n)$ from eq.(2). Then we can show that

$$\frac{1}{2}\Pr(D(A_{i+1})=0) + \frac{1}{2}\Pr(D(A_i \setminus A_{i+1})=1) > \frac{1}{2} + \frac{1}{2np(n)}.$$

This means that

$$\Pr[D(g, g^{u||b||0^i}) = b) > \frac{1}{2} + \frac{1}{2np(n)}.$$

Thus D can be used as a prediction algorithm of Lemma 1 with $\epsilon = 1/2np(n)$.

We finally show that we can solve the DLSE problem by using Lemma 3. Suppose that we are given (g, y) such that $y = g^{v||0^{n-c}}$. In order to apply Lemma 3, we first let $t = 4/\epsilon = 8np(n)$. We next guess the value of $msb_{\log t}(v)$. For each guessed value γ, we apply Lemma 3 and obtain $\tilde{v}$. We then check if $y = g^{\tilde{v}||0^{n-c}}$. If so, we have found that $v = \tilde{v}$. Otherwise, we try another guessed value. The number of possible values of $msb_{\log t}(v)$ is

$$2^{\log t} = t = 8np(n).$$

Therefore, the exhaustive search on $msb_{\log t}(v)$ runs in polynomial time. Consequently, we can find v in polynomial time with overwhelming probability. □

Efficient Signcryption with Key Privacy from Gap Diffie-Hellman Groups

Benoît Libert* and Jean-Jacques Quisquater

UCL Crypto Group
Place du Levant, 3. B-1348 Louvain-La-Neuve. Belgium
{libert,jjq}@dice.ucl.ac.be – http://www.uclcrypto.org/

Abstract. This paper proposes a new public key authenticated encryption (signcryption) scheme based on the Diffie-Hellman problem in Gap Diffie-Hellman groups. This scheme is built on the scheme proposed by Boneh, Lynn and Shacham in 2001 to produce short signatures. The idea is to introduce some randomness into this signature to increase its level of security in the random oracle model and to re-use that randomness to perform encryption. This results in a signcryption protocol that is more efficient than any combination of that signature with an El Gamal like encryption scheme. The new scheme is also shown to satisfy really strong security notions and its strong unforgeability is tightly related to the Diffie-Hellman assumption in Gap Diffie-Hellman groups.

Keywords: signcryption, Gap Diffie-Hellman groups, provable security

1 Introduction

The concept of public key signcryption schemes was proposed by Zheng in 1997 ([29]). The purpose of this kind of primitive is to perform encryption and signature in a single logical step in order to obtain confidentiality, integrity, authentication and non-repudiation more efficiently than the sign-then-encrypt approach. The drawback of this latter solution is to expand the final ciphertext size (this could be impractical for low bandwidth networks) and increase the sender and receiver's computing time. Several efficient signcryption schemes have been proposed since 1997. The original scheme proposed in [29] was based on the discrete logarithm problem but no security proof was given. Zheng's original construction was only proven secure in 2002 ([3]) by Baek et al. who described a formal security model in a multi-user setting. In 2000, Steinfeld and Zheng ([27]) proposed another scheme for which the unforgeability of ciphertexts relies on the intractability of the factoring problem but they provided no proof of chosen ciphertext security.

The drawback of the previously cited solutions is that they do not offer easy non-repudiation of ciphertexts: a recipient cannot prove to a third party that some plaintext was actually signcrypted by the sender. Bao and Deng ([5]) proposed a method to add universal verifiability to Zheng's cryptosystem but their

* This author was supported by the DGTRE's First Europe Project.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 187–200, 2004.

scheme was shown ([26]) to leak some information about the plaintext as other schemes like [28]. The latter schemes can easily be modified to fix their problem but no strong guarantee of unforgeability can be obtained for them since the unforgeability of ciphertexts relies on the forking lemma ([24],[25]) which does not provide tight security reductions (see [16] for details). In the discrete logarithm setting, another scheme was shown in [26] to be chosen ciphertext secure under the Gap Diffie-Hellman assumption but it was built on a modified version of the DSA signature scheme which is not provably secure currently. As a consequence, no proof of unforgeability could be found for that scheme. An RSA-based scheme was described by Malone-Lee and Mao ([20]) who provided proofs for both unforgeability under chosen-message attacks and chosen ciphertext security. Unfortunately, they only considered a security in a single-user setting rather than the more realistic multi-user setting. Furthermore, the security of that scheme is only loosely related to the RSA assumption. However, none of these schemes is provably secure against insider attacks: in some of them, an attacker learning some user's private key can recover all messages previously signcrypted by that user.

In 2002, An et al. ([1]) presented an approach consisting in performing signature and encryption in parallel: a plaintext is first transformed into a pair (c, d) made of a commitment c and a de-commitment d in such a way that c reveals no information about m while the pair (c, d) allows recovering m. Once he completed the transformation, the signer can jointly encrypt c and sign d in parallel using appropriate encryption and signature schemes. The de-signcryption operation is then achieved by the recipient in a parallel fashion: the signature on d is verified while c is decrypted and the pair (c, d) is then used to recover the plaintext. This method decreases the computation time to signcrypt a message to the maximum of the times required by the underlying encryption and signature processes but the commitment step unfortunately involves some computation overhead. To improve this parallel approach, Pieprzyk and Pointcheval ([22]) proposed to use a (2,2)-Shamir secret sharing as an efficient commitment scheme: a plaintext is first splitted into two shares s_1, s_2 which do not individually reveal any information on m. s_1 is used as a commitment and encrypted while s_2 is signed as a de-commitment. The authors of [22] also gave a construction allowing them to integrate any one-way encryption system (such as the basic RSA) with a weakly secure signature (non-universally forgeable signatures in fact) into a chosen ciphertext secure and existentially unforgeable signcryption scheme.

Dodis et al. ([11]) recently proposed another technique to perform parallel signcryption. Their method consists in a Feistel probabilistic two-paddings (called PSEP for short) which can be viewed as a generalization of other existing probabilistic paddings (OAEP, OAEP+, PSS-R,etc.) and involve a particular kind of commitment schemes. The authors of [11] showed that their construction also allows optimal exact security, flexible key management, compatibility with PKCS standards and has other interesting properties. They also claim that their scheme outperforms all existing signcryption solutions. We do not agree with that point since their method, like all other parallel signcryption proposi-

tions, has a significant drawback: the recipient of a message is required to know from whom a ciphertext emanates before beginning to verify the signature in parallel with the decryption operation. A trivial solution to this problem would be to append a tag containing the sender's identity to the ciphertext but this would prevent the scheme from satisfying the notion of ciphertext anonymity formalized by Boyen in [10] (intuitively, this notion expresses the inability for someone observing a ciphertext to determine who the sender is nor to whom it is intended) that can be a desirable feature in many applications (see [10] for examples). Furthermore, by the same arguments as those in [6], one can easily notice that the probabilistic padding described in [11] does not allow the key privacy property to be achieved when instantiated with trapdoor permutations such as RSA, Rabin or Paillier: in these cases, given a ciphertext and a set of public keys, it is possible to determine under which key the message was encrypted. An anonymous trapdoor permutation or a repeated variant of the padding PSEP (as the solutions proposed in [6]) could be used to solve this problem but this would decrease the scheme's efficiency.

In this paper, we propose a new discrete logarithm based signcryption scheme which satisfies strong security notions: chosen ciphertext security against insider attacks (except the hybrid composition proposed in [17] and the identity based scheme described in [10], no discrete logarithm based authenticated encryption method was formally proven secure in such a model before), strong unforgeability against chosen-message attacks, ciphertext anonymity in the sense of [10] (this is an extension of the notion of key privacy proposed in [6] to the signcryption case). We also prove that it satisfies a new security notion that is related to the one of ciphertext anonymity and that we call 'key invisibility'. We show that the scheme's strong unforgeability is really tightly related to the hardness of the Diffie-Hellman problem unlike the scheme proposed in [10] whose proof of unforgeability relies on Pointcheval and Stern's forking lemma and thus only provides a loose reduction to a computational problem. In fact, except the hybrid construction of [17] (whose semantic security is based on the stronger hash oracle Diffie-Hellman assumption) our scheme appears to be the first discrete logarithm based signcryption protocol whose (strong) unforgeabililty is proven to be tightly related to the Diffie-Hellman problem. About the semantic security of the scheme, we give heuristic arguments showing that it is more tightly related to the Diffie-Hellman problem than expressed by the bounds at first sight. Unlike [1],[11] and [22], our protocol is sequential but it is efficient and does not require the recipient of a message to know who is the sender before starting the de-signcryption process. Our scheme borrows a construction due to Boyen ([10]) and makes extensive use of the properties of some bilinear maps over the so-called Gap Diffie-Hellman groups (in fact, the structure of these groups is also exploited in our security proofs). Before describing our scheme, we first recall the properties of these maps in section 2. The section 3 formally describes the security notions that our scheme, depicted in section 4, is shown to satisfy in the security analysis presented in section 5.

2 Preliminaries

2.1 Overview of Pairings

Let k be a security parameter and q be a $k-$bit prime number. Let us consider groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of the same prime order q. For our purposes, we need a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ satisfying the following properties:

1. Bilinearity: $\forall\ P, Q \in \mathbb{G}_1,\ \forall\ a, b \in \mathbb{Z}_q^*$, we have $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$.
2. Non-degeneracy: for any $P \in \mathbb{G}_1$, $\hat{e}(P, Q) = 1$ for all $Q \in \mathbb{G}_1$ iff $P = \mathcal{O}$.
3. Computability: an efficient algorithm allows computing $\hat{e}(P, Q)\ \forall\ P, Q \in \mathbb{G}_1$.

The modified Weil pairing ([8]) and the Tate pairing are admissible maps of this kind. The group $\mathbb{G}_1$ is a suitable cyclic elliptic curve subgroup while $\mathbb{G}_2$ is a cyclic subgroup of the multiplicative group associated to a finite field. We now recall some problems that provided underlying assumptions for many previously proposed pairing based cryptosystems. These problems are formalized according to the elliptic curve additive notation.

Definition 1. *Given groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order q, a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ and a generator P of $\mathbb{G}_1$,*

- *The **Computational Diffie-Hellman problem** (CDH) in $\mathbb{G}_1$ is, given $\langle P, aP, bP \rangle$ for unknown $a, b \in \mathbb{Z}_q$, to compute $abP \in \mathbb{G}_1$.*
- *The **Decisional Diffie-Hellman problem** (DDH) is, given $\langle P, aP, bP, cP \rangle$ for unknown $a, b, c \in \mathbb{Z}_q$, to decide whether $ab \equiv c \pmod q$ or not. Tuples of the form $\langle P, aP, bP, cP \rangle$ for which the latter condition holds are called "Diffie-Hellman tuples".*
- *The **Gap Diffie-Hellman problem** (GDH) is to solve a given instance $\langle P, aP, bP \rangle$ of the CDH problem with the help of a DDH oracle that is able to decide whether a tuple $\langle P, a'P, b'P, c'P \rangle$ is such that $c' \equiv a'b' \pmod q$.*

As shown in [18], a pairing can implement a DDH oracle. Indeed, in a group $\mathbb{G}_1$ for which pairings are efficiently computable, to determine whether a tuple $\langle P, aP, bP, cP \rangle$ is a valid Diffie-Hellman tuple or not, it suffices to check if $\hat{e}(P, cP) = \hat{e}(aP, bP)$. This kind of group, where the DDH problem is easy while the CDH one is still believed to be hard, is called Gap Diffie-Hellman groups in the literature ([18],[21]).

3 Security Notions for Signcryption Schemes

We first recall the two usual security notions: the security against chosen ciphertext attacks which is also called semantic security and the unforgeability against chosen-message attacks. We then consider other security notions that were proposed by Boyen ([10]) in 2003. In the notion of chosen ciphertext security, we consider a multi-user security model as already done in [1],[3],[11],l22] and [10] to allow the adversary to query the de-signcryption oracle on ciphertexts created

with other private keys than the attacked one. We also consider the security against insider attacks by allowing the attacker to choose to be challenged on a signcrypted text created by a corrupted user (i.e. a user whose private key is known to the attacker). Indeed, for confidentiality purposes, we require the owner of a private key to be unable to find any information on a ciphertext created with that particular key without knowing which randomness was used to produce that ciphertext. As already considered in [1],[10],[11] and [22], this also allows us showing that an attacker stealing a private key does not threaten the confidentiality of messages previously signcrypted using that private key.

Definition 2. *We say that a signcryption scheme is semantically secure against chosen ciphertext attacks (we call this security notion SC-IND-CCA) if no probabilistic polynomial time (PPT) adversary has a non-negligible advantage in the following game:*

1. *The challenger runs the key generation algorithm* ***Keygen*** *to generate a private/public key pair* (sk_U, pk_U). sk_U *is kept secret while* pk_U *is given to the adversary* $\mathcal{A}$.
2. $\mathcal{A}$ *performs a first series of queries in a first stage. These queries can be of the following kinds:*
   - *Signcryption queries:* $\mathcal{A}$ *produces a message* $m \in \mathcal{M}$ *and an arbitrary public key* pk_R *(that public key may differ from* pk_U*) and requires the result* $\mathtt{Signcrypt}(m, sk_U, pk_R)$ *of the signcryption oracle.*
   - *De-signcryption queries:* $\mathcal{A}$ *produces a ciphertext* σ *and requires the result of the operation* $\mathtt{De\text{-}signcryt}(\sigma, sk_U)$. *This result is made of a signed plaintext and a sender's public key if the obtained signed-plaintext is valid for the recovered sender's public key. Otherwise (that is if the obtained plaintext-signature pair is not valid for the obtained public key when performing the de-signcryption operation with the private key* sk_U*), the* $\perp$ *symbol is returned as a result.*

   These queries can be asked adaptively: each query may depend on the answers to previous ones.
3. $\mathcal{A}$ *produces two plaintexts* $m_0, m_1 \in \mathcal{M}$ *of equal size and an arbitrary private key* sk_S. *The challenger then flips a coin* $b \leftarrow_R \{0,1\}$ *to compute a signcryption* $\sigma = \mathtt{Signcrypt}(m_b, sk_S, pk_U)$ *of* m_b *with the sender's private key* sk_S *under the attacked receiver's public key* pk_U. σ *is sent to* $\mathcal{A}$ *as a challenge.*
4. *The adversary performs new queries as in the first stage. Now, it may not ask the de-signcryption of the challenge* σ *with the private key* sk_U *of the attacked receiver.*
5. *At the end of the game,* $\mathcal{A}$ *outputs a bit* b' *and wins if* $b' = b$.

$\mathcal{A}$*'s advantage is defined to be* $Adv^{\mathtt{ind\text{-}cca}}(\mathcal{A}) := 2Pr[b' = b] - 1$.

In the notion of unforgeability captured by the formal definition below, as in many other previous works ([1],[3],[10],[11],[17],[22], etc.), we allow a forger attempting to forge a ciphertext on behalf of the attacked user U to know the receiver's private key. In fact, the attacker has to come with the intended receiver's private key sk_R as a part of the forgery. The motivation is to prove that

no attacker can forge a ciphertext intended to any receiver on behalf of a given sender. In particular, no dishonest user can produce a ciphertext intended to himself and try to convince a third party that it emanates from a honest user.

Definition 3. *We say that a signcryption scheme is strongly existentially unforgeable against chosen-message attacks (SC-SUF-CMA) if no PPT adversary has a non-negligible advantage in the following game:*

1. *The challenger generates a key pair* (sk_U, pk_U) *and* pk_U *is given to the forger* $\mathcal{F}$.
2. *The forger* $\mathcal{F}$ *queries the oracles* $\mathtt{Signcrypt}_{sk_U}(.,.)$ *and* $\mathtt{De\text{-}signcrypt}_{sk_U}(.)$ *exactly as in the previous definition. Again, these queries can also be produced adaptively.*
3. *At the end of the game,* $\mathcal{F}$ *produces a ciphertext* σ *and a key pair* (sk_R, pk_R) *and wins the game if the result of the operation* $\mathtt{De\text{-}signcrypt}(\sigma, sk_R)$ *is a tuple* (m, s, pk_U) *such that* (m, s) *is a valid signature for the public key* pk_U *such that* σ *was not the output of a signcryption query* $\mathtt{Signcrypt}(m, sk_U, pk_R)$ *made during the game.*

Recall that, in the corresponding notion of conventional (i.e. non-strong) unforgeability for signcryption schemes, the attacker cannot win if the outputted ciphertext was the result of any signcryption query. In our context, as in [1],[17],[11], and many other works, the forger is allowed to have obtained the forged ciphertext as the result of a signcryption query for a different receiver's public key than the one corresponding to the claimed forgery. The only constraint is that, for the message m obtained by de-signcryption of the alleged forgery with the chosen private key sk_R, the outputted ciphertext σ was not obtained as the result of a $\mathtt{Signcrypt}(m, sk_U, pk_R)$ query.

In [10], Boyen also proposed additional security notions for signcryption schemes. One of the most important ones was the notion of ciphertext anonymity that can be viewed as an extension to authenticated encryption schemes of the notion of key privacy already considered by Bellare et al in [6]. Intuitively, in the context of public key encryption, a scheme is said to have the key privacy property if ciphertexts convey no information about the public key that was used to create them. In the signcryption setting, we say that the ciphertext anonymity (or key privacy) property is satisfied if ciphertexts contain no information about who created them nor about to whom they are intended. This notion is a transposition into the non-identity based setting of the one presented in [10]. It can be described like that.

Definition 4. *A signcryption scheme is said to satisfy the ciphertext anonymity property (also called key privacy or key indistinguishability: we call this notion SC-INDK-CCA for short) if no PPT distinguisher has a non-negligible advantage in the following game:*

1. *The challenger generates two key pairs* $(sk_{R,0}, pk_{R,0})$ *and* $(sk_{R,1}, pk_{R,1})$. $pk_{R,0}$ *and* $pk_{R,1}$ *are given to the distinguisher* $\mathcal{D}$.

2. *$\mathcal{D}$ adaptively performs queries $\mathbf{Signcrypt}(m, sk_{R,c}, pk_R)$, for arbitrary recipient keys pk_R, and $\mathbf{De\text{-}signcrypt}(\sigma, sk_{R,c})$ for $c = 0$ or $c = 1$.*
3. *Once stage 2 is over, $\mathcal{D}$ outputs two private keys $sk_{S,0}$ and $sk_{S,1}$ and a plaintext $m \in \mathcal{M}$. The challenger then flips two coins $b, b' \leftarrow_R \{0,1\}$ and computes a challenge ciphertext $\sigma = \mathbf{Signcrypt}(m, sk_{S,b}, pk_{R,b'})$ which is sent to $\mathcal{D}$.*
4. *$\mathcal{D}$ adaptively performs new queries as in stage 2 with the restriction that, this time, it is disallowed to ask the de-signcryption of the challenge σ with the private keys $sk_{R,0}$ or $sk_{R,1}$.*
5. *At the end of the game, $\mathcal{D}$ outputs bits d, d' and wins if $(d, d') = (b, b')$. Its advantage is defined to be $Adv^{\text{indk-cca}}(D) := Pr[(d, d') = (b, b')] - 1/4$.*

Again, this notion captures the security against insider attacks since the distinguisher is allowed to choose a set of two private keys among which the one used as sender's key to create the challenge ciphertext is picked by the challenger. The above definition can be viewed as a transposition to the non-identity based setting of the definition of ciphertext anonymity proposed by Boyen ([10]) as well as an extension of the definition of key privacy ([6]) to the authenticated encryption context. We introduce another notion called 'key invisibility' which is close to the concept (formalized by Galbraith and Mao in [14]) of invisibility for undeniable signatures. Intuitively, this notion expresses the impossibility to decide whether a given ciphertext was actually created using a given particular sender's private key and a given particular receiver's public key.

Definition 5. *We say that a signcryption scheme satisfies the key invisibility (we denote this notion by SC-INVK-CCA for short) if no PPT distinguisher has a non-negligible advantage in the following game:*

1. *The challenger generates a private/public key pair (sk_U, pk_U). pk_U is given to the distinguisher $\mathcal{D}$.*
2. *$\mathcal{D}$ adaptively performs queries $\mathbf{Signcrypt}(m, sk_U, pk_R)$, for arbitrary recipient keys pk_R, and $\mathbf{De\text{-}signcrypt}(\sigma, sk_U)$.*
3. *Once stage 2 is over, $\mathcal{D}$ outputs a private key sk_S and a plaintext $m \in \mathcal{M}$. The challenger then flips a coins $b \leftarrow_R \{0,1\}$. If $b = 0$, then the challenger returns an actual challenge ciphertext $\sigma = \mathbf{Signcrypt}(m, sk_S, pk_U)$ to $\mathcal{D}$. If $b = 1$, then the challenger returns a random σ uniformly taken from the ciphertext space $\mathcal{C}$.*
4. *$\mathcal{D}$ adaptively performs new queries as in stage 2 with the restriction that, this time, it cannot require the de-signcryption of the challenge σ with the private keys sk_U.*
5. *At the end of the game, $\mathcal{D}$ outputs bits d and wins if $d = b$. Its advantage is defined as $Adv^{\text{invk-cca}}(D) := 2Pr[d = b] - 1$.*

Again, we allow the distinguisher to choose which private key is used as a part of the challenge to take insider attacks into account.

Galbraith and Mao ([14]) showed that anonymity and invisibility are essentially equivalent security notions for undeniable signatures. While one can prove

in the same way that key privacy and key invisibility are also essentially equivalent for some particular encryption schemes, such an equivalence turns out to be unclear in the signcryption case. In fact, one cannot prove that a distinguisher against the key invisibility implies a distinguisher against the key privacy with the same advantage (because two random coins are used by the challenger in the definition of key privacy and a single one for key anonymity). However, we can prove that, for signcryption schemes satisfying some particular properties (that is, for a given message and a given sender's private key, the output of the signcryption algorithm must be uniformly distributed in the ciphertext space when the receiver's public key is random), we can prove that key invisibility implies key privacy. This will be showed in [19]. In the next section we propose a scheme that satisfies both of them (in addition to the usual notions of semantic security and unforgeability) in the random oracle model.

4 A Diffie-Hellman Based Signcryption Scheme with Key Privacy

This section presents a signcryption scheme whose unforgeability under chosen-message attacks is tightly related to the hardness of the computational Diffie-Hellman problem in Gap Diffie-Hellman groups. Our solution relies on the BLS signature ([9]) whose security is enhanced by a random quantity U which is used for encryption purposes but also acts as a random salt to provide a tighter security reduction to the Diffie-Hellman problem in $\mathbb{G}_1$ in the proof of unforgeability.

We assume that both the sender and the receiver agreed on public parameters: security parameters k and ℓ, cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order $q \geq 2^k$ such that ℓ is the number of bits required to represent elements of $\mathbb{G}_1$, a generator P of $\mathbb{G}_1$ and a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. They also agree on cryptographic hash functions $H_1 : \{0,1\}^{n+2\ell} \rightarrow \mathbb{G}_1$, $H_2 : {\mathbb{G}_1}^3 \rightarrow \{0,1\}^{\ell}$ and $H_3 : \{0,1\}^{\ell} \rightarrow \{0,1\}^{n+\ell}$ where n denotes the size of plaintexts (i.e. the message space is $\mathcal{M} = \{0,1\}^n$). The scheme consists of the following three algorithms (we recall that the symbol $\oplus$ denotes the bitwise exclusive OR).

Keygen: user u picks a random $x_u \leftarrow_R \mathbb{Z}_q$ and sets his public key to $Y_u = x_u P \in \mathbb{G}_1$. His private key is x_u. We will denote the sender and the receiver respectively by $u = S$ and $u = R$ and their key pair by (x_S, Y_S) and (x_R, Y_R).

Signcrypt: to signcrypt a plaintext $m \in \{0,1\}^n$ intended to R, the sender S uses the following procedure

1. Pick a random $r \leftarrow_R \mathbb{Z}_q$ and compute $U = rP \in \mathbb{G}_1$.
2. Compute $V = x_S H_1(m, U, Y_R) \in \mathbb{G}_1$.
3. Compute $W = V \oplus H_2(U, Y_R, rY_R) \in \{0,1\}^{\ell}$ and then scramble the plaintext together with the sender's public key: $Z = (m||Y_S) \oplus H_3(V) \in \{0,1\}^{n+\ell}$.

The ciphertext is given by $\sigma = \langle U, W, Z \rangle \in \mathbb{G}_1 \times \{0,1\}^{n+2\ell}$.

De-signcrypt: when receiving a ciphertext $\sigma = \langle U, W, Z \rangle$, the receiver R has to perform the steps below:

1. Compute $V = W \oplus H_2(U, Y_R, x_R U) \in \{0,1\}^{\ell}$.
2. Compute $(m||Y_S) = Z \oplus H_3(V) \in \{0,1\}^{n+\ell}$. Reject σ if Y_S is not a point on the curve on which $\mathbb{G}_1$ is defined.
3. Compute $H = H_1(m, U, Y_R) \in \mathbb{G}_1$ and then check if $\hat{e}(Y_S, H) = \hat{e}(P, V)$. If this condition does not hold, reject the ciphertext.

The consistency of the scheme is easy to verifiy. To prove to a third party that the sender S actually signed a plaintext m, the receiver just has to forward it m and (U, V, Y_R). The third party can then compute H as in the step 3 of de-signcrypt and perform the signature verification as in the same step 3. We note that, in the signcryption algorithm, the recipient's public key must be hashed together with the pair (m, U) in order to achieve the provable strong unforgeability.

As pointed out in [15], in some applications, it is interesting for the origin of a signcrypted text to be publicly verifiable (by firewalls for example). In some other applications, it is undesirable: indeed as explained in [10], in some cases, it is better for a signcrypted text not to convey any information about its sender nor about its intended receiver. This property, called anonymity of ciphertexts, is provided by the above scheme as shown in the next section.

From an efficiency point of view, we can easily verifiy that the above scheme is at least as efficient and more compact than any sequential composition of the BLS signature ([9]) with any other Diffie-Hellman based chosen ciphertext secure encryption scheme ([2],[4],[12],[13],[23],etc.): indeed only three scalar multiplications in $\mathbb{G}_1$ are required for the signcryption operation while 1 multiplication and 2 pairings must be performed in the de-signcryption process. A sequential combination of the BLS signature with the encryption scheme proposed in [2] would involve an additional multiplication at decryption. If we take $\ell \approx k \geq 160$ (by working with an appropriate elliptic curve), we see that ciphertexts are about 480 bits longer than plaintexts. Any combination of the BLS signature with a CCA-secure El Gamal type cryptosystem would result in longer final ciphertexts. With the same choice of parameters, a composition of the BLS signature with the lenth-saving El Gamal encryption scheme ([2]) would result in ciphertexts that would be 640 bits longer than plaintexts.

5 Security Analysis

In this section, we first show that an adversary against the SC-IND-CCA security of the scheme implies a PPT algorithm that can solve the Diffie-Hellman problem in $\mathbb{G}_1$ with high probability. This fact is formalized by the following theorem.

Theorem 1. *In the random oracle model, if an adversary $\mathcal{A}$ has a non-negligible advantage ϵ against the SC-IND-CCA security of the above scheme when running in a time t and performing q_{SC} signcryption queries, q_{DSC} de-signcryption queries and q_{H_i} queries to oracles H_i (for $i = 1, \ldots, 4$), then there exists an*

algorithm $\mathcal{B}$ that can solve the CDH problem in the group $\mathbb{G}_1$ with a probability $\epsilon' \geq \epsilon - q_{H_3} q_{DSC}/2^{2k}$ in a time $t' < t + (4q_{DSC} + 2q_{H_2})t_e$ where t_e denotes the time required for one pairing evaluation.

Proof. The algorithm $\mathcal{B}$ runs $\mathcal{A}$ as a subroutine to solve the CDH problem in a polynomial time. Let (aP, bP) be a random instance of the CDH problem in $\mathbb{G}_1$. $\mathcal{B}$ simulates $\mathcal{A}$'s challenger in the game of definition 2 and starts it with $Y_u = bP \in \mathbb{G}_1$ as a challenge public key. $\mathcal{A}$ then adaptively performs queries as explained in the definition. To handle these queries, $\mathcal{B}$ maintains lists L_i to keep track of the answers given to oracle queries on H_i for $i = 1, 2, 3$. Hash queries on H_2 and H_3 are treated in the usual way: $\mathcal{B}$ first checks in the corresponding list if the oracle's value was already defined at the queried point. If it was, $\mathcal{B}$ returns the defined value. Otherwise, it returns a random element from the appropriate range and updates the corresponding list. When a hash query $H_1(m, U, Y_R)$ is performed, $\mathcal{B}$ first looks if the value of H_1 was previously defined for the input (m, U, Y_R). If it was, the previously defined value is returned. Otherwise, $\mathcal{B}$ picks a random $t \leftarrow_R \mathbb{Z}_q$, returns $tP \in \mathbb{G}_1$ as an answer and inserts the tuple (m, U, Y_R, t) into L_1.

Now, let us see how signcryption and de-signcryption queries are dealt with:

- For a signcryption query on a plaintext m with a recipient's public key Y_R both chosen by the adversary $\mathcal{A}$, $\mathcal{B}$ first picks a random $r \leftarrow_R \mathbb{Z}_q$, computes $U = rP \in \mathbb{G}_1$ and checks if L_1 contains a tuple (m, U, Y_R, t) indicating that $H_1(m, U, Y_R)$ was previously defined to be tP. If no such tuple is found, $\mathcal{B}$ picks a random $t \leftarrow_R \mathbb{Z}_q$ and puts the entry (m, U, Y_R, t) into L_1. $\mathcal{B}$ then computes $V = tY_u = t(bP) \in \mathbb{G}_1$ for the random t chosen or recovered from L_1. The rest follows as in the normal signcryption process: $\mathcal{B}$ computes rY_R (for the Y_R specified by the adversary), runs the H_2 simulation process to obtain $h_2 = H_2(U, Y_R, rY_R)$, and then computes $W = V \oplus h_2$ and $Z = (m||Y_u) \oplus h_3$ where h_3 is obtained by simulation of the H_3 oracle on the input *V. (U, W, Z)* is then returned as a signcryption of m from the sender of public key Y_u to the recipient of public key Y_R.
- For a de-signcryption query on a ciphertext $\langle U, W, Z \rangle$ and a sender's public key Y_S both chosen by $\mathcal{A}$, $\mathcal{B}$ proceeds as follows: it scans the list L_2, looking for tuples $(U, Y_u, S_i, h_{2,i})$ (with $0 \leq i \leq q_{H_2}$) such that $V_i = h_{2,i} \oplus W$ exists in an entry $(V_i, h_{3,i})$ of L_3 and, for the corresponding elements $h_{3,i}$, $(m_i, Y_{S,i}) = h_{3,i} \oplus Z \in \{0,1\}^{n+\ell}$ is such that there exists an entry $(m_i, U, Y_u, h_{1,i})$ in the list L_1. If no such tuples are found, the $\perp$ symbol is returned to $\mathcal{A}$. Otherwise, elements $(m_i, U, V_i, S_i, h_{1,i})$ satisfying those conditions are kept for future examination. If one of them satisfies both $\hat{e}(P, S_i) = \hat{e}(U, Y_u)$ and $\hat{e}(Y_{S,i}, h_{1,i}) = \hat{e}(P, V_i)$, then $\langle m_i, (U, V_i) \rangle$ is returned as a message-signature pair together with the sender's public key $Y_{S,i}$.

At the end of the first stage, $\mathcal{A}$ outputs two plaintexts m_0 and m_1 together with an arbitrary sender's private key x_S and requires a challenge ciphertext built under the recipient's public key Y_u. $\mathcal{B}$ ignores m_0 and m_1 and randomly picks two binary strings $W \leftarrow_R \{0,1\}^\ell$ and $Z \leftarrow_R \{0,1\}^{n+\ell}$. A challenge ciphertext

$\sigma = \langle U, W, Z \rangle = \langle aP, W, Z \rangle$ is then sent to $\mathcal{A}$ that then performs a second series of queries at a second stage. These queries are handled by $\mathcal{B}$ as those at the first stage. As done in many other papers in the literature, it is easy to show that $\mathcal{A}$ will not realize that σ is not a valid signcryption for the sender's private key x_S and the public key Y_u unless it asks for the hash value $H_2(aP, bP, abP)$. In that case, the solution of the Diffie-Hellman problem would be inserted in L_2 exactly at that moment and it does not matter if the simulation of $\mathcal{A}$'s view is no longer perfect.

At the end of the game, $\mathcal{A}$ produces a result which is ignored by $\mathcal{B}$. The latter just looks into the list L_2 for tuples of the form $(aP, bP, D_i, .)$. For each of them, $\mathcal{B}$ checks whether $\hat{e}(P, D_i) = \hat{e}(aP, bP)$ and, if this relation holds, stops and outputs D_i as a solution of the CDH problem. If no tuple of this kind satisfies the latter equality, $\mathcal{B}$ stops and outputs "failure".

Now to assess $\mathcal{B}$'s probability of success, let us denote by $\mathtt{AskH}_2$ the event that $\mathcal{A}$ asks the hash value of abP during the simulation. As done in several papers in the literature (see [8] or [10]), as long as the simulation of the attack's environment is perfect, the probability for $\mathtt{AskH}_2$ to happen is the same as in a real attack (i.e. an attack where $\mathcal{A}$ interacts with real oracles). In a real attack we have

$$\Pr[b = b'] \leq \Pr[b = b' | \neg \mathtt{AskH}_2] \Pr[\neg \mathtt{AskH}_2] + \Pr[\mathtt{AskH}_2] = \frac{1}{2} + \frac{1}{2} \Pr[\mathtt{AskH}_2]$$

and then we have $\epsilon = 2\Pr[b = b'] - 1 \leq \Pr[\mathtt{AskH}_2]$. Now, the probability that the simulation is not perfect remains to be assessed. The only case where it can happen is when a valid ciphertext is rejected in a de-signcryption query. It is easy to see that for every pair $(V_i, h_{3,i})$ in L_3, there is exactly one pair $(h_{1,i}, h_{2,i})$ of elements in the range of oracles H_1 and H_2 providing a valid ciphertext. The probability to reject a valid ciphertext is thus not greater than $q_{H_3}/2^{2k}$. The bound on $\mathcal{B}$'s computation time derives from the fact that every de-signcryption query requires at most 4 pairing evaluations while the extraction of the solution from L_2 implies to compute at most $2q_{H_2}$ pairings. □

The above security proof makes use of the pairing's bilinearity to handle de-signcryption queries and thus avoids the use of constructions such as [2], [23], [13], [12] that would increase the ciphertext's length or imply additional computation in the de-signcryption operation (this is one of the interests in working with Gap Diffie-Hellman groups). This results in a worst-case bound on algorithm $\mathcal{B}$'s computation time that seems to be loose: to extract the solution of the CDH problem, $\mathcal{B}$ might have to compute up to $2q_{H_2}$ pairings if $\mathcal{A}$ only queries oracle H_2 on tuples of the form $(aP, bP, .)$. If we allow up to 2^{60} H_2-queries, this appears to be a loose bound at first sight. But we stress that, heuristically, if $\mathcal{A}$ asks many hash queries of tuples $(aP, bP, .)$ that are not valid Diffie-Hellman tuples, that means it has no better strategy to find information about the challenge ciphertext than computing the XOR of the ciphertext's W-component with hash values of random tuples. Such a strategy would not be more efficient for $\mathcal{A}$ than an exhaustive search of the solution to the Diffie-Hellman instance embedded in

the challenge ciphertext. An attacker having a non-negligible advantage against the semantic security would ask much less than 2^{60} hash queries of invalid Diffie-Hellman tuple. We can thus expect that, at the end of the simulation, L_2 only contains a limited number of entries $(aP, bP, .)$.

We note that the bound on $\mathcal{B}$'s probability of success is tight: if we allow $q_{DSC} \leq 2^{30}$ and $q_{H_3} \leq 2^{60}$, with $k \geq 160$, we obtain $q_{H_3} q_{DSC}/2^{2k} \leq 2^{-230}$ which is a negligible function of the parameter k.

The following theorem claims the strong unforgeability of the scheme.

Theorem 2. *In the random oracle model, if there exists an adversary $\mathcal{F}$ that has a non-negligible advantage ϵ against the SC-SUF-CMA security of the scheme when running in a time t, making q_{SC} signcryption queries, q_{DSC} de-signcryption queries and at most q_{H_i} queries on oracles H_i (for $i = 1, \ldots, 4$), then there exists an algorithm $\mathcal{B}$ that can solve the Diffie-Hellman problem in $\mathbb{G}_1$ with a probability $\epsilon' > \epsilon - q_{SC} q_{H_1}/2^k - q_{DSC} q_{H_3}/2^{2k}$ in a time $t' < t + 4q_{DSC} t_e$ where t_e denotes the time required for a pairing evaluation.*

Proof. given in the full paper ([19]). □

This time, we obtain bounds that are explicitly tight. With $k \geq 160$, if we allow $q_{H_3} < 2^{60}$ and $q_{H_1} < 2^{50}$, $q_{DSC} < 2^{30}$ we have $q_{SC} q_{H_1}/2^k < 1/2^{80}$ and we still have $q_{DSC} q_{H_3} < 2^{-230}$. We thus have a negligible degradation of $\mathcal{B}$'s probability of success when compared to the adversary's advantage. The bound on $\mathcal{B}$'s running time is also reasonably tight for $q_{DSC} < 2^{30}$.

The theorem below claims the ciphertext anonymity property of the scheme.

Theorem 3. *In the random oracle model, assume there exists a PPT distinguisher $\mathcal{D}$ that has a non-negligible advantage against the SC-INDK-CCA security of the scheme when running in a time t, performing q_{SC} signcryption queries, q_{DSC} de-signcryption queries and q_{H_i} queries to oracle H_i (for $i = 1, \ldots, 4$). Then there exists an algorithm $\mathcal{B}$ that solves the CDH problem with an advantage $\epsilon' > \epsilon - 1/2^{n+\ell-1} - q_{DSC} q_{H_3}/2^{2k}$ when running in a time $t' < t + (4q_{DSC} + 2q_{H_2}) t_e$ where t_e denotes the time required for one pairing evaluation.*

Proof. given in the full paper ([19]). □

Again, the bound on $\mathcal{B}$'s computation time might seem to be meaningless but, as for the proof of theorem 1, we can argue that a distinguisher performing many H_2 queries on invalid Diffie-Hellman tuples would have no better strategy than an exhaustive search for Diffie-Hellman instances embedded in the challenge-ciphertext. However, if we look at the proofs of semantic security and ciphertext anonymity for the scheme described in [10], although no bound is explicitly given for the running time of solvers for the bilinear Diffie-Hellman problems, these bounds are not tighter than ours. Furthermore, the proof of ciphertext anonymity provided in [10] leads to a significant degradation of the solver's advantage when compared to the distinguisher's one.

We close this section with the following theorem related to the key invisibility.

Theorem 4. *In the random oracle model, if there exists a distinguisher $\mathcal{D}$ having a non-negligible advantage ϵ against the SC-INVK-CCA security of the scheme when running in a time t and performing q_{H_i} queries to oracles H_i, for $i = 1, \ldots, 4$, q_{SC} signcryption queries and q_{DSC} de-signcryption queries, then there exists an algorithm $\mathcal{B}$ that solves the CDH problem with an advantage $\epsilon' > \epsilon - 1/2^{n+\ell-1} - q_{DSC}q_{H_3}/2^{2k}$ in a time $t' < t + (4q_{DSC} + 2q_{H_2})t_e$ where t_e is the time required for a pairing evaluation.*

Proof. given in the full paper ([19]). □

6 Conclusions

We proposed a new Diffie-Hellman based signcryption scheme satisfying strong security requirements. It turns out to be the discrete log based signcryption protocol whose unforgeability is the most tightly related to the Diffie-Hellman problem (except the construction in [17], all provably secure solutions are built on signatures having a security proof relying on the forking lemma ([24], [25]) and the CCA-security of [17] relies on stronger assumptions than the present scheme). By heuristic arguments, we argued that the reduction from an adaptive chosen ciphertext adversary to a solver for the Diffie-Hellman problem is also efficient. We also introduced a security notion called 'key invisibility' that can be shown to imply 'key privacy' in some cases (see [19] for details).

References

1. J.-H. An, Y. Dodis and T. Rabin. On the Security of Joint Signature and Encryption. In *Advances in Cryptology - Eurocrypt'02,* LNCS 2332, pp. 83–107. Springer, 2002.
2. J. Baek, B. Lee and K. Kim. Secure Length-Saving ElGamal Encryption under the Computational Diffie-Hellman Assumption. In *Proceedings of ACISP'00,* LNCS 1841, pp. 49–58. Springer, 2000.
3. J. Baek, R. Steinfeld and Y. Zheng. Formal Proofs for the Security of Signcryption. In *Proceedings of PKC'02,* LNCS 2274, pp. 80–98. Springer, 2002.
4. J. Baek and Y. Zheng. Simple and Efficient Threshold Cryptosystem from the Gap Diffie-Hellman Group. Available at http://citeseer.nj.nec.com/567030.html.
5. F. Bao and R.-H. Deng. A Signcryption Scheme with Signature Directly Verifiable by Public Key. In *Proceedings of PKC'98,* LNCS 1998, pp. 55–59, 1998.
6. M. Bellare, A. Boldyreva, A. Desai and D. Pointcheval. Key-Privacy in Public-Key Encryption. In *Advances in Cryptology - Asiacrypt'01,* LNCS 2248, pp. 566–582. Springer, 2001.
7. M. Bellare, P. Rogaway, *Random oracles are practical: A paradigm for designing efficient protocols,* Proc. of the 1^{st} ACM Conference on Computer and Communications Security, pp. 62-73, 1993.
8. D. Boneh and M. Franklin. Identity Based Encryption From the Weil Pairing. In *Advances in Cryptology - Proceedings of Crypto'01,* LNCS 2139, pp. 213–229. Springer, 2001.
9. D. Boneh, B. Lynn and H. Shacham. Short Signatures from the Weil Pairing. In *Advances in Cryptology - Proceedings of Asiacrypt'01,* LNCS 2248, pp. 514–532. Springer, 2001.

10. X. Boyen. Multipurpose Identity-Based Signcryption: A Swiss Army Knife for Identity-Based Cryptography. In *Advances in Cryptology - Crypto '03,* LNCS 2729, pp. 382–398. Springer, 2003.
11. Y. Dodis, M.-J. Freedman and S. Walfish. Parallel Signcryption with OAEP, PSS-R and other Feistel Paddings. 2003. Available at http://eprint.iacr.org/2003/043/.
12. E. Fujisaki and T. Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In *Advances in Cryptology - Crypto'99,* LNCS 1666, pp. 537–554. Springer, 1999.
13. E. Fujisaki and T. Okamoto. How to Enhance the Security of Public-Key Encryption at Minimum Cost. In *Proceedings of PKC'99,* LNCS 1560, pp. 53–68. Springer, 1999.
14. S. Galbraith and W. Mao. Invisibility and Anonymity of Undeniable and Confirmer Signatures. In *Topics in Cryptology - CT-RSA 2003,* LNCS 2612, pp. 80-97. Springer, 2003.
15. C. Gamage, J. Leiwo and Y. Zheng. Encrypted Message Authentication by Firewalls. In *Proceedings of PKC'98,* LNCS 1560, pp. 69–81. Springer, 1998.
16. E.-J. Goh and S. Jarecki. A Signature Scheme as Secure as the Diffie-Hellman Problem. In *Advances in Cryptology - Eurocrypt'03,* LNCS 2656, pp. 401–415. Springer, 2003.
17. I.-R. Jeong, H.-Y. Jeong, H.-S. Rhee, D.-H. Lee and I.-L. Jong. Provably Secure Encrypt-then-Sign Composition in Hybrid Signcryption. In *Proceedings of ICISC'02,* LNCS 2587, pp. 16-34. Springer, 2002.
18. A. Joux and K. Nguyen. Separating Decision Diffie-Hellman from Diffie-Hellman in Cryptographic Groups. In *Journal of Cryptology,* vol. 16-Number 4, pp. 239–247. Springer, 2003.
19. B. Libert and J.-J. Quisquater, Efficient Signcryption with Key Privacy from Gap Diffie-Hellman Groups. Full paper, available on http://eprint.iacr.org.
20. J. Malone-Lee and W. Mao. Two Birds One Stone: Signcryption using RSA. In *Topics in Cryptology - CT-RSA 2003,* LNCS 2612, pp. 211-225. Springer, 2003.
21. T. Okamoto and D. Pointcheval. The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes. In *Proceedings of PKC'01,* LNCS 1992. Springer, 2001.
22. J. Pieprzyk and D. Pointcheval. Parallel Authentication and Public-Key Encryption. In *Proceedings of ACISP'03,* LNCS 2727, pp. 383–401. Springer, 2003.
23. D. Pointcheval. Chosen-Ciphertext Security for any One-Way Cryptosystem . In *Proceedings of PKC'00,* LNCS 1751, pp. 129–146. Springer, 2000.
24. D. Pointcheval and J. Stern. Security Proofs for Signature Schemes. In *Advances in Cryptology - Eurocrypt'96,* LNCS 1992, pp. 387–398. Springer, 1996.
25. D. Pointcheval and J. Stern. Security Arguments for Digital Signatures and Blind Signatures. In *Journal of Cryptology,* vol. 13-Number 3, pp. 361–396. Springer, 2000.
26. J.-B. Shin, K. Lee and K. Shim. New DSA-verifiable Signcryption Schemes. In *Proceedings of ICISC'02,* LNCS 2587, pp. 35–47. Springer, 2002.
27. R. Steinfeld and Y. Zheng. A Signcryption Scheme Based on Integer Factorization. In *Proceedings of ISW'00,* LNCS 1975, pp. 308–322, Springer, 2000.
28. B.-H. Yum and P.-J. Lee. New Signcryption Schemes Based on KCDSA. In *Proceedings of ICISC'01,* LNCS 2288, pp. 305–317, Springer, 2001.
29. Y. Zheng. Digital signcryption or how to achieve cost (signature & encryption) $<<$ cost(signature) + cost(encryption). In *Advances in Cryptology - Crypto'97,* LNCS 1294, pp. 165–179, Springer, 1997.

Algebraic Attacks over $GF(2^k)$, Application to HFE Challenge 2 and Sflash-v2

Nicolas T. Courtois

Axalto Cryptography Research & Advanced Security,
36-38 rue de la Princesse, BP 45, 78430 Louveciennes Cedex, France
courtois@minrank.org
http://www.nicolascourtois.net

Abstract. The problem MQ of solving a system of multivariate quadratic equations over a finite field is relevant to the security of AES and for several public key cryptosystems. For example Sflash, the fastest known signature scheme (cf. [1]), is based on MQ equations over $GF(2^7)$, and Patarin's 500 $ HFE Challenge 2 is over $GF(2^4)$. Similarly, the fastest alleged algebraic attack on AES due to Courtois, Pieprzyk, Murphy and Robshaw uses a MQ system over $GF(2^8)$.

At present very little is known about practical solvability of such systems of equations over $GF(2^k)$. The XL algorithm for Eurocrypt 2000 was initially studied over $GF(p)$, and only recently in two papers presented at CT-RSA'02 and ICISC'02 the behaviour of XL is studied for systems of equations over $GF(2)$. In this paper we show (as expected) that XL over $GF(2^k)$, $k > 1$ (never studied so far) does not always work very well. The reason is the existence of additional roots to the system in the extension field, which is closely related to the remark made by Moh, claiming that the XSL attack on AES cannot work. However, we explain that, the specific set of equations proposed by Murphy and Robshaw already contains a structure that removes the problem. From this, we deduce a method to modify XL so that it works much better over $GF(2^k)$.

In addition we show how to break the signature scheme Sflash-v2 recently selected by the European consortium Nessie, by three different methods derived from XL. Our fastest attack is in 2^{58}. All the three attacks apply also to HFE Challenge 2, and our best attack is in 2^{63}.

Key Words: Multivariate quadratic equations, MQ problem, overdefined systems of multivariate equations, XL algorithm, Gröbner bases, algebraic attacks on AES, XSL, Murphy-Robshaw equations on AES.

1 Introduction

In the perpetual search for hard problems on which to base cryptographic security, there is a growing interest in so called "multivariate problems". These problems are usually NP-hard. In terms of scalability of the systems, the best

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 201–217, 2004.

problems are those for which all known attacks are exponential: it is then sufficient to increase slightly the parameter sizes, to keep up with progress in the attacks, or with an increase in the speed of computers. One of such problems is the problem MQ, of solving a system of multivariate quadratic equations over a small finite field. Several public key cryptosystems based on MQ have been proposed, for example the HFE family [30,9]. In this paper we study generic attacks that solve the underlying MQ problem independently of the existence of the trapdoor. They apply also to random quadratic equations.

At Crypto'99, Shamir and Kipnis present a surprising method called relinearization for solving overdefined systems of multivariate quadratic equations. They point out that, if such a system of equations is overdefined (much more equations than needed), then it can be solved much faster than expected. Subsequently, at Eurocrypt 2000 [32], Courtois, Klimov, Patarin and Shamir, present a new algorithm called XL, (and also FXL) that can be seen as an improved version of relinearization.

From [32] and still at present, very little is known about the exact complexity and behaviour of XL. Initially in [32] it was studied mainly over $GF(p)$. Recently a lot of interest emerged in solving MQ systems over $GF(2)$ and $GF(2^k)$, due to the Courtois-Pieprzyk method to attack AES by such means [15,26]. At CT-RSA 2002 Courtois and Patarin study the XL algorithm over $GF(2)$ and show it works much better than expected from [32] or from the naive criticism of it published on the internet [24]. At ICISC 2002, Courtois studies the extension of XL to equations of degree higher than 2, and again demonstrates that it works very well, allowing to cryptanalyse the stream cipher Toyocrypt, see [7]. The object of this paper is to study rather MQ over fields of the form $GF(2^k), k > 1$. Such equations appear for example in the signature schemes Flash, Sflash and Sflash-v2 published at CT-RSA 2002, out of which Sflash-v2 has been selected by Nessie (in company of ECDSA and RSA-PSS). Also, in the fastest known alleged attack on AES due to Courtois-Pieprzyk and Murphy-Robshaw [15,26], the equations are quadratic over $GF(2^8)$.

2 Notation and Conventions Used in This Paper

The MQ Problem

In this paper we consider the problem of solving a system of m multivariate quadratic equations with n variables over a finite field $GF(q)$. We use very similar notations than in [32] and [14]. The input variables are denoted by x_i and belong to $GF(q)$ with $q = 2^k$. The equations are denoted by l_i and are quadratic (which means they can also include linear and constant terms). Our system to solve will be:

$$\mathcal{A}: \begin{cases} l_1(x_1, \ldots x_n) & = 0 \\ & \vdots \\ l_m(x_1, \ldots x_n) & = 0 \end{cases}$$

Given m, n, q we call MQ the problem of finding one (not necessarily all) solutions to such a system chosen at random. Typically in cryptographic applications, k can be between 4 and 8 and m, n can between 26 and 1600 (for AES, see [15,26]). The MQ problem is NP-hard, see [20].

Remark: In XL description in [32,14] the powers of variables are taken in $GF(q)$, i.e. reduced modulo q to the range $1, \ldots, q-1$, because of the equation $x_i^q = x_i$ of the finite field $GF(q)$. Thus if $q = 2$ there would be no powers of x_i bigger than 1. For us it makes no difference, as in all cases studied in this paper, we have $q \geq 16$ and we will never generate or manipulate equations of degree equal or bigger than $q - 1$.

Instances of MQ That Will Be Used in This Paper

If $m > n$ the system is said to be overdefined. Similarly as in [32,14], we will see that for a fixed n, the bigger is m, the more easy becomes the MQ problem. If $m < n$ the system is said to be underdefined, and efficient algorithms for the underdefined MQ has been studied in [5]. In general, following [32,14], we expect that the hardest case of MQ is when $m \approx n$.

In practice, if we have a system with $n > m$, as in the Sflash public key [12], we will start by fixing some variables to arbitrary values, get a system with $m \geq n$, and the try to solve it. (When over $GF(2^k)$, it is unclear if one can take advantage from the initial $n > m$, cf. [5].)

For all our MQ systems we will always insure/assume that **the system has one and unique solution,** we refer to Section 4.1 or to the end of Section 5.1 to see why it is very important. To have one unique solution happens frequently in cryptographic applications of MQ, and it is also the average number of solutions of a random MQ with $m = n$. Moreover, in practice, for systems that have several solutions, we can always reduce to a system having one solution, by guessing a few variables.

Manipulating the Equations

Because the right hand of all our equations is always 0, it is very useful to identify a multivariate polynomial and an equation that says it is equal to 0. Thus the equation $l_i(x_1, \ldots x_n) = 0$ can be simply called the equation l_i, and the equation $x_1 \cdot l_2(x_1, \ldots x_n) = 0$ can be called simply $x_1 l_2$.

We say that the equations of the form $\prod_{j=1}^{k} x_{i_j} \cdot l_i = 0$, with all the i_j being pairwise different, are of type $x^k l$, and we call $x^k l$ the set of all these equations. For example the initial equations $\mathcal{A}$ are of type l. We observe that each solution x that satisfies all the equations l_i, also does satisfy all the equations of type $x^k l$, for any $k \geq 0$. Similarly we denote by x^k the set of all terms of degree exactly K, $\prod_{j=1}^{K} x_{i_j}$. By extension we define $x^0 = \{1\}$, the constant monomial.

Let $D \in \mathbb{N}$. We consider all the polynomials $\prod_j x_{i_j} \cdot l_i$ of total degree $\leq D$. Let $\mathcal{I}_D$ be the set of equations they span. $\mathcal{I}_D$ is the linear space generated by all the $x^k l$, $0 \leq k \leq D-2$. We have $\mathcal{I}_D \subset \mathcal{I}$, $\mathcal{I}$ being the ideal spanned by the l_i

We call $\mathcal{T}$ the set of monomials, including the constant monomial, that appear in all the equations of $\mathcal{I}_D$, $\mathcal{T} = \bigcup_{i=0}^{D} x^i$.

3 The Basic Principle of XL

Let D be the parameter of XL algorithm. Following [32,14]:

Definition 3.0.1 (The XL algorithm). Execute the following steps:

1. **Multiply:** Generate all the products $\prod_{j=1}^{k} x_{i_j} \cdot l_i \in \mathcal{I}_D$ with $k \leq D-2$.
2. **Linearize:** Consider each monomial in the x_i of degree $\leq D$ as a new variable and perform Gaussian elimination on the equations obtained in 1. The ordering on the monomials must be such that all the terms containing one variable (say x_1) are eliminated last.
3. **Find $\mathbf{x_1}$:** Assume that step 2 yields at least one univariate equation in the powers of x_1. Solve this equation over the finite fields (e.g., with Berlekamp's algorithm). There may be several roots.
4. **Recover the other variables:** For each root x_1 substitute it to the expanded equations and, directly from the Gaussian reduction done in step 3, find the values of all the other monomials, in particular for all the other variables x_i.

4 The Necessary Condition for XL to Work

We will always assume $q = 2^k, k > 1$. We also always assume $D < q-1$, because we will have $q \geq 16$ and and D will remain quite small (XL is exponential in D). The XL algorithm consists of multiplying the initial m equations l_i by all possible monomials of degree up to $D-2$, so that the total degree of resulting equations is D. With the notations introduced above, this set of equations is called $\mathcal{I}_D$. Let R be the number of equations generated in $\mathcal{I}_D$ and T be the number of all monomials. When $D < q-1$ we have:

$$T = |\mathcal{T}| = \sum_{i=0}^{D} |x^i| = \sum_{\lambda=0}^{D} \binom{n+\lambda-1}{\lambda} = \binom{n+D}{D}$$

$$R = |\mathcal{I}_D| = m\left(\sum_{\lambda=0}^{D-2} \binom{n+\lambda-1}{\lambda}\right) = m\binom{n+D-2}{D-2}$$

It is likely that not all of these equations are linearly independent, and we denote by $Free$ the exact dimension of $\mathcal{I}_D$. We have $Free \leq R$. We also have necessarily $Free \leq T$.

The basic principle of XL is the following: one monomial in T can be generated in many different ways when different equations are multiplied by different monomials. Therefore T grows slower than R and for some D we will have $R \geq T$. Then we expect that $Free \approx T$, as obviously it cannot be bigger than T. In [32],

when $Free \geq T - D$, it is possible to obtain one equation with only one variable x_1, and XL will succeed. (However in [14] two improved versions of XL are introduced: XL' and XL2, that will work when $Free < T - T'$, for some T' that may be substantially bigger then D.)

Simplified Analysis of XL from [32]

In Section 6 of [32], R is evaluated as $R = m \cdot \frac{n^{D-2}}{(D-2)!}$ and T is evaluated as $\frac{n^D}{D!}$. The authors state that "if most of the equations are linearly independent" then XL will succeed as long as $R \geq T$, which gives that:
$m \geq \frac{n^2}{D(D-1)}$, and thus they obtain the (approximative) bound $D \geq \frac{n}{\sqrt{m}}$.

4.1 General Theory and Moh's Comments on XL

In [24], Moh states that "From the theory of Hilbert-Serre, we may deduce that the XL program will work for many interesting cases for D large enough". According to [23], in XL we always have $Free \leq T - \alpha$. and when D is sufficiently big, we have $Free = T - \alpha$. Here α is the number of solutions to the system, including not only the solutions when $x_i \in Gf(q)$, but also when the x_i lie in an algebraic extension of the field $GF(q)$, or projective solutions (points at infinity). Thus, on the one side, under our condition that our system has one and unique solution, and if there is no projective solutions or in an extension field, XL should work and for D large enough we should have $Free = T - 1$. On the other side, this condition is necessary, and when the system has several solutions, $Free = T - 1$ *is* never achieved and the basic XL cannot work. Thus, in Section 4 of [24], Moh shows an interesting example on which the XL always fails. However:

- For XL over $GF(2)$, it is shown in [14] that this kind of counter-example cannot occur, because of the added equations $x_i^2 = x_i$ that make that the system has no points at infinity, and the additional solutions in the algebraic closure of $GF(2)$ are excluded.
- In this paper we work over $GF(2^k), k \neq 1$ and we will face this problem. In Section 6 we will see that XL will not work well when $m = n$, then in Section 7 we will present a new version of XL, called XLF, that will work even in this case. (In addition, we will see that in practice, if 2^k is not too big, and only then, two other already known versions of XL can also circumvent this problem.)

5 Important Remarks About XL Algorithm over $GF(2^k)$

Let $Free$ be the dimension of $\mathcal{I}_D$, i.e. the maximum number of equations that are linearly independent. Very little is known about the value of $Free$ for $D \geq 3$. In the paper that describes XL, the authors demonstrate that XL works with a series of computer simulations over $GF(127)$ (and some more are given in the

extended version of the paper [32]). In [14,7] the authors study the XL algorithm over $GF(2)$. They do many computer simulations and are able to predict the exact value *Free* obtained in these simulations. In this paper we will do the same for XL over $GF(2^k), k > 1$.

5.1 The Behaviour of XL - Upper Bounds

In general it is not always possible to have $Free = R$. In many cases the equations generated by XL are not all linearly independent. One reason for this is that *Free* cannot exceed T, as the equations lie in a linear space spanned by all the T monomials. We have therefore always

$$Free \leq \min(T, R)$$

Moreover, it is possible to see that if the system is not contradictory, and has one solution, then:

$$Free \leq \min(T - 1, R)$$

This can be shown by contradiction: if $Free = T$ then by elimination of $T - 1$ non-constant monomials, some liner combination of the given equations will be 1, and if there is a solution to these equations, by substituting it, we get 0 = 1.

5.2 The Behaviour of XL - Interesting Cases

As we will see in the present paper, the behaviour of XL over $GF(2^k)$, when k is not too small, (e.g. $k = 7$) is very similar to the general behaviour of XL over a big field $GF(p)$ studied in details (with many computer simulations) in [32]:

- XL works very well for (even slightly) overdefined systems of equations, i.e. when m exceeds n by even a small value, cf. Appendix A.
- However when $m \approx n$, and as long as the XL parameter D is smaller than the cardinal of the field, it is possible to see that XL does **not** work very well for systems of quadratic equations over $GF(2^k)$.

A different behaviour is observed for XL over a very small finite field (such as $GF(2)$ or $GF(3)$): XL works much better and there is no "problem" at all when $m \approx n$. Detailed explanation and many computer simulations for this case are given in [14] and in the appendix of [7].

6 Our Computer Simulations on XL

In all our simulations we pick a random system of linearly independent quadratic (non-homogenous) equations $y_i = f_i(x_1, \ldots, x_n)$ and pick a random input $x = (x_1, \ldots, x_n)$. Then we modify the constants in the system in order to have a system that has a solution (and gives 0 in x). The system solve is then of the form $l_i(x_0, \ldots, x_{n-1}) = 0$, for $i = 1, \ldots m$.

In Appendix A we show that for overdefined systems of equations over $GF(2^k)$, i.e. when $m > n + \varepsilon$, XL works very well. Below we study the hard case, when $m \approx n$.

6.1 Simulations on XL over $GF(2^k)$ when $m = n$

Table 1. XL over $GF(2^7)$ for $m = n$

n	2	2	2	3	3	3	3	3	4	**4**	4	4	4
m	2	2	2	3	3	3	3	3	4	**4**	4	4	4
D	2	3	4	2	4	6	7	8	4	**5**	10	15	**16**
R	2	6	12	3	30	105	168	252	60	**140**	1980	2860	12240
T	6	10	15	10	35	84	120	165	70	**126**	1001	1365	4845
$Free$	2	6	11	3	27	76	112	157	54	**110**	985	1349	4829
$\frac{Free}{T-D}$	0.50	0.86	1.00	0.38	0.87	0.97	0.99	1.00	0.82	**0.91**	0.99	0.99	1.00
$Success$			OK					OK					OK

Legend:
n number of variables.
m number of equations.
D we generate equations of total degree $\leq D$ in the x_i.
R number of equations generated (independent or not).
T number of monomials of degree $\leq D$.
Free number of linearly independent equations among the R equations.
$\diamond$ Note: XL will work when $Free \geq T - D$.

It is very interesting to observe the column in bold characters: though already for $D = 5$ XL gives $R > T$ and therefore it could work, it will not work until we have $D = 16$. The difference is quite big: the complexity of the attack grows exponentially in D.

We see that for $m = n$ and over $GF(2^7)$ the XL algorithm works very poorly. In [32], for simulations over $GF(127)$, it appears that the minimum degree is $D = 2^n$. We observe the same here. The reason for this is, following [32], that for $m = n$ the system has many solutions not only in the base field, but also in the algebraic closure.

It is interesting to see that basic XL over $GF(2^7)$ becomes impractical already for $m = n = 5$: in this case, doing XL with $D = 2^5 = 32$ would give a complexity of about 2^{49}, more than exhaustive search in $2^{7 \cdot 5} = 2^{35}$. Later we will improve XL to handle such systems much faster.

6.2 Simulations on XL over $GF(2^k)$ when $m = n + \varepsilon$

We will see that, similarly as in [32], the behaviour of XL will dramatically improve when m becomes slightly bigger than n. We do not longer need $D = 2^n$ and XL works about as soon as R becomes larger than T.

7 XLF - New Version of XL for $m \approx n$ and $GF(2^k)$

In Section 6.1 we saw that XL does not work very well when $m = n$ and over a large field $GF(2^k)$. From the analysis done in [32], we expect that this is due

Table 2. XL over $GF(2^7)$ for $m = n + \varepsilon$ (notations as for Table 1)

n	4	4	4	4	4	4	5	5	5	5
m	4	4	5	5	6	6	6	6	7	7
D	15	16	4	5	3	4	4	5	3	4
R	2860	12240	75	175	30	90	126	336	42	147
T	1365	4845	70	126	35	70	126	252	56	126
$Free$	1349	4829	65	125	30	69	111	246	42	125
$\frac{Free}{T-D}$	0.99	1.00	0.98	1.03	0.94	1.05	0.91	1.00	0.79	1.02
$Success$		OK		OK		OK		OK		OK

to existence of many additional solutions to our system of equations that lie in an extension field. In this section we introduce a new version of XL, called XLF, designed specifically to handle this problem over fields $GF(2^k)$. XL stands for multiply (X) and linearize (L), the new method is called XLF, which stands for multiply (X) and linearize (L) and apply Frobenius mappings (F). The basic idea of XLF is borrowed from the Murphy-Robshaw representation of AES [26]. Each variable x that appears in the system of equations will be duplicated k times, instead of x_i, we will have k variables denoted by $(x_i), (x_i^2), (x_i^4), \ldots, (x_i^{2^{k-1}})$. Each equation $0 = \sum_{ij} \alpha_{ij} x_i x_j$ will be also duplicated k times: we will write: $0 = \sum_{ij} \alpha_{ij}^2 (x_i^2)(x_j^2)$ etc. After doing XL expansion we get k times as many equations of degree D and k times as many variables as in the regular XL execution. Then we add some new equations that relate the new variables to each other. For example, we add $k \cdot n$ quadratic equations as follows: for each i we have $(x^2) = (x) \cdot (x)$ up to $(x) = (x^{2^{k-1}}) \cdot (x^{2^{k-1}})$. If $D \geq 4$ we have also kn equations of type $(x^4) = (x) \cdot (x) \cdot (x) \cdot (x)$ etc. Since the equations we added are only equalities between monomials, we may as well identify these monomials, which is equivalent to counting less monomials. In the extended version of this paper we give a precise list of all the monomials that are identified, and formulas to compute the resulting reduced number of monomials T.

7.1 Comparing XLF and XL

It is easy to see that by this simple trick, all the solutions with $x_i \notin GF(2^k)$ will be removed, because they cannot satisfy the added equations. We conjecture that XLF will work as long as R becomes somewhat bigger than T in the ordinary XL, (for example twice as big). This belief is motivated by the paper [14] where it is shown that the equations of the field $GF(2)$ make XL always work as long as $R > 1.1T$.

XLF is expected to work where the original XL fails, as for $m \approx n$ XL does not work well when $R > T$, as shown in Section 6. XLF uses k times as many equations, and k times as many monomials as XL. We expect therefore that the complexity of XLF will be only about k^ω bigger than the expected complexity of XL (if the XL itself does not work). Indeed, our simulations (Table 3) show that XLF works very well when XL fails, i.e. even when $m = n$.

Table 3. XLF algorithm over $GF(2^7)$ for $m = n$ (notations as for Table 1).

n	2	2	2	3	3	3	3	3
m	2	2	2	3	3	3	3	3
D	2	3	4	2	3	4	5	6
R	14	42	84	21	84	210	420	735
T	22	50	64	43	113	176	323	449
$Free$	14	42	61	21	84	168	315	445
$\frac{Free}{T-D}$	0.70	0.89	1.02	0.51	0.76	0.98	0.99	1.00
$Success$			OK					OK

n	4	4	4	4	4	5	5	5	5	5	5
m	4	4	4	4	4	5	5	5	5	5	5
D	4	5	6	7	8	3	4	5	6	7	8
R	420	980	1960	3528	5880	210	735	1960	4410	8820	16710
T	386	778	1226	2066	2976	346	736	1618	2843	5153	8128
$Free$	350	742	1218	2058	2970	210	630	1505	2800	5110	8120
$\frac{Free}{T-D}$	0.92	0.96	0.999	0.999	1.00	0.61	0.86	0.93	0.99	0.99	1.00
$Success$					OK						OK

We see that XLF behaves much better than XL for solving systems of equations over $GF(2^k)$ with $m = n$. For example, with XL, we need $D = 2^5 = 32$ to solve a system of 5 equations with 5 unknowns, while with XLF $D = 8$ is enough. This is an important improvement, because the complexity of both XL and XLF is very much the same with an important factor that is exponential in D.

7.2 Relation with Algebraic Attacks on AES

The idea of XLF algorithm introduced in this paper is closely related to the question of feasibility of an algebraic attack on AES [15]. In the Murphy-Robshaw representation of AES, see [26] for details, the equations are written over GF(256), and have the same structure as in XLF: for each variable (x) there is a variable that is always equal to (x^2), and for each equation, the square of it also present in the system.

These equations may be combined with the Courtois-Pieprzyk XSL attack on AES, (XSL is different from XL and beyond the scope of this paper). From the values of R and T obtained in XSL it seems that, if sufficiently many equations are linearly independent, AES 128 bits would be broken in about 2^{100}, see [15,26]. However, on a web page entitled "AES is not broken" [25], T.T.Moh unwillingly acknowledges that the XL algorithm will work, but objects for XSL and AES as follows: " new considerations of the XL method to the smallest field $GF(2)$ with the well-known trick of adding the equations $x_i^2 + x_i = 0$ to make the the component at infinity empty to satisfy the requirement of our Proposition 2" (...) "Note that this trick can not be used for the AES situation, since the

corresponding equations would be $x_i^{256} + x^i = 0$, the degrees 256 would be too high for practical purpose."

This is very interesting, because as far as we know, it is the **only** somewhat mathematically founded argument that have been put forward so far, suggesting that the Courtois-Pieprzyk-Murphy-Robshaw attack in 2^{100} might not work on AES. Yet as we have seen above, this argument is void: the structure of equations makes that each of the variables must lie in $GF(256)$. This excludes additional roots in extension fields that would make the attack fail. Moreover, it is also easy to see that with such equations there will be no points at infinity: if we homogenise the equations $(x_i^2) = (x_i) * (x_i)$ with a new variable (a), we get $(a) * (x_i^2) = (x_i) * (x_i)$, and then if $a = 0$, all the x_i will be 0, which is a contradiction.

Consequences for AES: Results on XL certainly not prove that the XSL attack on AES works. Yet the Moh argument saying it shouldn't work does not apply at all to this specific Courtois-Pieprzyk-Murphy-Robshaw system of equations. More generally, this paper shows that it is in general risky, and difficult to predict whether an algebraic attack will or will not work. We have seen that for (somewhat) deeply mathematical reasons XL does not work very well for Sflash. Yet, as we will see later, a subtle and finally quite minor modification of XL, such as XLF or XL', is enough to make it work and break Sflash.

8 New Attacks on Sflash

In this section we present three new methods that allow to break Sflash in less than the Nessie security requirement of 2^{80} Triple-DES computations.

8.1 Applying XLF to Sflash

In Sflash we have $n = 37$ and $m = 26$. Equations are quadratic over $GF(2^7)$. We fix 11 arbitrary variables to 0 and still expect to have on average one solution. Then we apply XLF, the new version of XL. For $D = 7$ we have $R = 7 \cdot 4417686$ and $T \approx 7 \cdot 4272048$. Though XL does certainly fail here, we expect that XLF may work. For $D = 7$ the complexity would be about $T^\omega \approx \mathbf{2^{67}}$. Even if we were too optimistic, and XLF works only for $D = 10$, then we still have an attack in $T^\omega \approx 2^{83}$ CPU clocks which is less than 2^{80} triple-DES computations required by Nessie.

8.2 Another Attack on Sflash Using XL' Method from [14]

In this section we present yet another and even simpler method to break Sflash. Instead of XL, we apply the XL' algorithm from [14]. With classical XL, for $D = 7$ we have $R = 4417686$ and $T = 4272048$, however in practice, and *Free* does not take the value $\geq T - D$ for a very long time. This makes XL fail so far. Still, as shown by all simulations of Section 6.1, *Free* remains very close to $T - D$, and from this we expect that the XL' version of XL described in [14] will

work. We have $n = 26$ and $m = 26$. We count all the monomials contain **only** the first 5 variables: let T' be their number, we have $T' = \binom{5+D}{D} = 792$. It seems very likely that the rank, usually close *to* $T - D$, will be at least $T - T' + 5$. Then we are able to eliminate **all** the monomials that contain **any** of the remaining $n - 5 = 26 - 5 = 21$ variables, and get a system of 5 equations of degree $D = 7$ with 5 variables, with $T' = 792$ monomials. Such a system can be solved by exhaustive search in about $2^{7 \cdot 5} \cdot 792 \cdot 5 \approx 2^{47}$. The total complexity of the attack will be $\left(2^{47} + T^{\omega}\right) \approx \mathbf{2^{58}}$ CPU clocks.

8.3 Another Attack on Sflash with Modified FXL

In this attack we will use the idea of FXL from [32]: guess values of few variables in Sflash, solve the system by XL, and then solve the system by XL. FXL leads very quickly to an overdefined system of equations and from [32] and following our experiments done in Section 6.2, we expect that after fixing a few variables XL will work.

Moreover, we will be able to do only once most of the Gaussian reduction that in FXL is done each time, which will give better results over basic FXL from [32]. We proceed as follows:

1. We start with MQ with $m = n = 26$ and over $GF(2^7)$.
2. We fix $f = 4$ variables (this is the optimal choice we have found).
3. We have 22 variables said of "type a" and $f = 4$ variables "of type b".
4. We multiply all the equations by all the products of degree up to $D = 6$ of the variables of "type a".
5. The number of equations is $R = 26\binom{22+D-2}{D-2} = 388700$.
6. In these equations we will eliminate all monomials of degree exactly $D = 6$ in the variables of "type a". Their number is exactly $T' = \binom{22+D-1}{D} = 296010$. They do not depend on the variables of "type b", and can be eliminated once for all.
7. Thus we get $R - T' = 92690$ equations that are of degree $D - 1 = 5$ in the variables of "type a".
8. If we fix a random value for the four variables of "type b", then we get a system of $R - T' = 92690$ equations with $T'' = \binom{22+5}{5} = 80730$ monomials that is sufficiently defined, as $9269 > 80730$.
9. We expect that if the guess for the four variables of "type b" is correct, then the system has a solution and the rank of this system is at most $80730 - 1$. However if the guess is wrong, we expect the system to be contradictory and the rank to be 80730.
10. We expect that on average exactly one guess will be correct.
11. The complexity to find the right values for the four variables of "type b" with Strassen's version of the Gaussian reduction is about:

$$2^{7 \cdot 4} \cdot 7/64 \cdot (80730)^{\log_2(7)} \approx \mathbf{2^{71}}.$$

Remark: It is possible to see that the matrices generated in our attacks are somewhat sparse and that they can probably still be (slightly) improved by using sparse linear algebra.

9 Application of Our Attacks to HFE Challenge 2

The HFE Challenge 2 was published by Patarin in the extended version of [30], with a price of 500 $. In the extended version of this paper we apply exactly "as they are" our 3 attacks from Section 8 Results are given in Table 4 and our best attack on HFE Challenge 2 gives 2^{63}.

10 Conclusion and Perspectives

The problem MQ of solving a set of multivariate quadratic equations over a finite field arises in cryptography (allowing to propose new cryptographic primitives), but also in cryptanalysis (for example for AES). In this paper we have studied the XL algorithm over $GF(2^k)$. We show that it works very well for overdefined equations and fails when $m \approx n$. Then we present XLF, a modified version of XL that works also in this case.

Using XLF, and also with two other versions of XL known as XL' and FXL, we present three new attacks on Sflash, a signature scheme accepted by the European Nessie consortium. All these three new attacks are faster than 2^{80}, and the fastest requires about 2^{58} CPU clocks. They also apply to Patarin's 500 $ HFE Challenge 2, and the best gives 2^{63}.

In our results, one can notice that XLF is not the best method to break Sflash and HFE Challenge 2. This is because 2^k is still not too big. It is possible to see that, when 2^k is very big, XLF, introduced in this paper, will be **the only method known** to solve efficiently systems of quadratic equations over $GF(2^k)$ and with $m = n$. To summarize:

Table 4. Summary of the results of this paper.

	XL from [32]	XLF - new	XL' from [14]	improved FXL
Sflash-v2	2^{282}	2^{67}	$\mathbf{2^{58}}$	2^{71}
Sflash-v3 [13], $m=56$	2^{458}	2^{110}	2^{102}	2^{100}
HFE Challenge 2	2^{122}	2^{76}	2^{70}	$\mathbf{2^{63}}$
General MQ, $m \approx n, k$ big	fails	**works**	fails	fails

In Appendix A of this paper we show that, as in [14], we succeed to predict perfectly the behaviour of XL for $D < 6$, and this is sufficient to cryptanalyse current versions of Sflash and HFE Challenge 2. In general, the asymptotic behaviour of XL can be studied by the theory of Gröbner bases, see [17,18,16,2]. We conjecture that complexity of solving MQ systems over a finite field with $m \approx n$ must grow exponentially with n, and even for equations over $GF(2)$, the easiest case, it can be shown that applying Buchberger algorithm to ideals generated in XL has single exponential worst case complexity, see [16] or [2].

Consequences for Sflash and HFE. We did not exhibit any structural weakness of these schemes. We simply showed that the proposed parameter sizes

are insufficient for the hardness of the generic one-way problem. These schemes will resist all the attacks described in the present paper if we increase parameters m and n. Thus in Table 4 above we see that the latest updated version Sflash-v3 from [13] remains very secure.

Potential consequences for other algebraic attacks such as XSL attack on AES. We showed that for systems of low degree equations over fields $GF(2^k)$, it is not hard to avoid additional solutions in the algebraic extension or at infinity, that would make algebraic attacks fail. The Frobenius-based transformation method (with adding new variables and new equations), inspired by [26] and developed in this paper, may be of independent interest: it can potentially be applied to various systems of equations solved by various methods. For example equations can be derived from a block cipher, to be later solved by XSL-type method [15]. This simple trick (not needed in [15] nor in [26]) can transform an attack that does not work, into an attack that does work, while increasing the size of equations only k times.

Note: The extended version of this paper is available from the author.

References

1. Mehdi-Laurent Akkar, Nicolas Courtois, Louis Goubin, Romain Duteuil: *A Fast and Secure Implementation of Slash,* PKC 2003, LNCS 2567, Springer, pp. 267-278.
2. B. Barkee, D. C. Can, J. Ecks, T. Moriarty, R. F. Ree: *Why You Cannot Even Hope to use Gröbner Bases in Public Key Cryptography: An Open Letter to a Scientist Who Failed and a Challenge to Those Who Have Not Yet Failed,* in Journal of Symbolic Computation 18, 1994, S. 497-501
3. Don Coppersmith, Shmuel Winograd: "Matrix multiplication via arithmetic progressions"; J. Symbolic Computation (1990), 9, pp. 251-280.
4. Nicolas Courtois, Magnus Daum and Patrick Felke: *On the Security of HFE, HFEv- and Quartz,* PKC 2003, LNCS 2567, Springer, pp. 337-350.
5. Nicolas Courtois, Louis Goubin, Willi Meier, Jean-Daniel Tacier: *Solving Underdefined Systems of Multivariate Quadratic Equations,* PKC 2002, LNCS 2274, Springer, pp. 211-227.
6. Nicolas Courtois: *The security of Hidden Field Equations (HFE);* Cryptographers' Track Rsa Conference 2001, LNCS 2020, Springer, pp. 266-281.
7. Nicolas Courtois: *Higher Order Correlation Attacks, XL algorithm and Cryptanalysis of Toyocrypt,* ICISC 2002, LNCS 2587, pp. 182-199, Springer. An updated version (2002) is available at `http://eprint.iacr.org/2002/087/`.
8. Jacques Patarin, Nicolas Courtois, Louis Goubin: *C*-+ and HM - Variations around two schemes of T. Matsumoto and H. Imai;* Asiacrypt'98, Springer.
9. Jacques Patarin, Louis Goubin, Nicolas Courtois: *Quartz,* 128*-bit long digital signatures;* Cryptographers' Track Rsa Conference 2001, LNCS 2020, pp.282-297, Springer. See [10] for the updated Quartz specification.
10. Jacques Patarin, Louis Goubin, Nicolas Courtois: *Quartz,* 128*-bit long digital signatures*; An updated version of Quartz specification. available at `http://www.cryptosystem.net/quartz/`
11. Jacques Patarin, Louis Goubin, Nicolas Courtois: *Flash, a fast multivariate signature algorithm;* Cryptographers' Track Rsa Conference 2001, LNCS 2020, pp. 298-307, Springer.

12. Nicolas Courtois, Louis Goubin and Jacques Patarin: Second updated version of Sflash specification (Sflash-v2). Available at http://www.cryptosystem.net/sflash/
13. Nicolas Courtois, Louis Goubin and Jacques Patarin: SFLASHv3, a fast asymmetric signature scheme. New, third version of Sflash specification (Sflash-v3), proposed after this paper was written. Available on eprint . iacr.org/2003/211/.
14. Nicolas Courtois and Jacques Patarin, *About the XL Algorithm over GF*(2), Cryptographers' Track RSA 2003, LNCS 2612, pages 141-157, Springer 2003.
15. Nicolas Courtois and Josef Pieprzyk, *Cryptanalysis of Block Ciphers with Overdefined Systems of Equations,* Asiacrypt 2002, LNCS 2501, pp.267-287, Springer, a preprint with a different version of the attack is available at http://eprint.iacr.org/2002/044/.
16. Magnus Daum: *Das Kryptosystem HFE und quadratische Gleichungssysteme über endlichen Körpern,* Diplomarbeit, Universität Dortmund, 2001. Available from daum@itsc.ruhr-uni-bochum.de
17. Jean-Charles Faugère: *A new efficient algorithm for computing Gröbner bases* (F_4), Journal of Pure and Applied Algebra 139 (1999) pp. 61-88. See www.elsevier.com/locate/jpaa
18. Jean-Charles Faugère: A new *efficient algorithm for computing Gröbner bases without reduction to zero (F5),* Workshop on Applications of Commutative Algebra, Catania, Italy, 3-6 April 2002, ACM Press.
19. Jean-Charles Faugère: Report on a successful attack of HFE Challege 1 with Gröbner bases algorithm F5/2, announcement that appeared in sci.crypt newsgroup on the internet in April 19th 2002.
20. Michael Garey, David Johnson: *Computers and Intractability, a guide to the theory of NP-completeness,* Freeman, see in particular p. 251.
21. Henri Gilbert, Marine Minier: *Cryptanalysis of SFLASH.* Eurocrypt 2002, LNCS 2232, Springer, pp. 288-298, 2002.
22. Antoine Joux, Jean-Charles Faugère: *Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems Using Gröbner Bases,* Crypto 2003, LNCS 2729, pp. 44-60, Springer.
23. Mireille Martin-Deschamps, private communication, University of Versailles.
24. T.T. Moh: *On The Method of XL and Its Inefficiency Against TTM,* invited talk at the American Mathematical Society regional meeting at the University of Notre Dame, April 8, 2000. available at http://eprint.iacr.org/2001/047/.
25. T.T. Moh: *On The Courtois-Pieprzyk's Attack on Rijndael,* September 18 2002, available at http://www.usdsi.com/aes.html.
26. S. Murphy, M. Robshaw: *Essential Algebraic Structure within the AES,* Crypto 2002, LNCS 2442, Springer.
27. NESSIE Portfolio of recommended cryptographic primitives, available at www.cosic.esat.kuleuven.ac.be/nessie/deliverables/decision-final.pdf
28. NESSIE Security Report, revised final version 2.0, available at https://www.cosic.esat.kuleuven.ac.be/nessie/deliverables/D20-v2.pdf
29. Jacques Patarin: *Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88*; Crypto'95, Springer, LNCS 963, pp. 248-261, 1995.
30. Jacques Patarin: *Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): two new families of Asymm. Algorithms,* Eurocrypt'96, Springer, pp. 33-48.
31. Adi Shamir, Aviad Kipnis: *Cryptanalysis of the HFE Public Key Cryptosystem;* In Advances in Cryptology, Proceedings of Crypto'99, Springer, LNCS.

32. Adi Shamir, Jacques Patarin, Nicolas Courtois, Alexander Klimov, *Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations,* Eurocrypt'2000, LNCS 1807, Springer, pp. 392-407.

A More Computer Simulations - Predicting the Behaviour of XL

In this section we will show that XL works very well for even slightly overdefined systems of equations over $GF(2^k)$, i.e. when m exceeds n by even a small value. Moreover, we will show, as in [14], how to predict the behaviour of XL, and this prediction will in many cases remain valid also when $m = n$. (the case $m \approx n$ is studied in section 6).

As before, in these simulations we pick a random system of linearly independent quadratic (non-homogenous) equations $y_i = f_i(x_1, \ldots, x_n)$ and pick a random input $x = (x_1, \ldots, x_n)$. Then we modify the constants in the system to have a system that has (at least) one solution x.

A.1 The Behaviour of XL over $GF(2^k)$ for $D = 3$

We have always $Free \leq \min(T - 1, R)$. We have done various computer simulations with $D = 3$ and in our simulations, for $D = 3$, we have always $Free = \min(T - 1, R)$ or $Free = \min(T - 1, R) - 1$.

In the following table we fix n and try XL on a random system of m linearly independent equations with growing m and with a fixed D.

Table 5. XL over $GF(2^7)$ for $D = 3$ (notations as for Table 1)

n	10	10	10	10	10	20	20	20	20	20
m	10	15	20	25	26	20	40	60	80	85
D	3	3	3	3	3	3	3	3	3	3
R	110	165	220	275	286	420	840	1260	1680	1785
T	286	286	286	286	286	1771	1771	1771	1771	1771
$Free$	110	165	220	275	285	420	840	1260	1680	1770
$Expected$	110	165	220	275	285	420	840	1260	1680	1770
$\frac{Free}{T-D}$	0.39	0.58	0.77	0.96	1.00	0.24	0.48	0.71	0.95	1.00
$Success$					OK					OK

A.2 The Behaviour of XL over $GF(2^k)$ for $D = 4$

When $D = 4$ we do not have $Free = \min(T, R)$ anymore.

We see that for $D = 4$ most of the equations are linearly independent. We observed that we have always:

$$\text{For } D = 4, \quad Free = \min\left(T - 1, R - \binom{m}{2}\right).$$

Table 6. XL over $GF(2^7)$ for $D = 4$ (notations as for Table 1)

n	5	5	5	5	10	10	10	10
m	5	6	7	8	10	15	17	18
D	4	4	4	4	4	4	4	4
R	105	126	147	168	660	990	1122	1188
T	126	126	126	126	1001	1001	1001	1001
$Free$	95	111	125	125	615	885	986	1000
$Expected$	95	111	125	125	615	885	986	1000
$\frac{Free}{T-D}$	0.76	0.91	1.02	1.02	0.62	0.89	0.99	1.00
$Success$				OK				OK

The fact that $Free = R - \binom{m}{2}$ when $R - \binom{m}{2} \leq T$, means that, in all cases, there are $\binom{m}{2}$ linear dependencies between the equations in R. As in [14], we are able to explain the origin (and the exact number) of these linear dependencies: Let l_i be the equations names (not expanded, just written as "l_i"), and let $[l_i]$ denote the expanded expression of these equations as quadratic polynomials. Then we have:

$$l_i[l_j] = [l_i]l_j$$

For each $i \neq j$, the above equation defines a linear dependency between the equations of XL. This explains the $\binom{m}{2}$ dependencies.

Example: For example if $l_1 = x_1x_3 + x_4$ and $l_5 = x_2x_1 + x_4x_7$ then the notation $l_1[l_5] = [l_1]l_5$ denotes the following linear dependency between the $l_ix_jx_k$:

$$l_1x_2x_1 + l_1x_4x_7 = l_5x_1x_3 + l_5x_4.$$

A.3 The Behaviour of XL over $GF(2^k)$ for $D = 5$

Following the method from [14] and used in the previous chapter, we will try to predict the exact number of linearly independent equations that will be obtained for $D = 5$. First of all, we have the $\binom{m}{2}$ linear dependencies of type $l_i[l_j] = [l_i]l_j$ that are the same that existed for $D = 4$. In addition we have dependencies like:

$$l_i[l_j]x_k = [l_i]l_jx_k$$

It gives $n \cdot \binom{m}{2}$ dependencies. By inspection we check that for $D = 5$ we are unable to generate any more dependencies. From the above, we expect that:

$$\text{For } D = 5, \quad Free = \min\left(T - 1, R - (n+1)\binom{m}{2}\right).$$

Is that all the linear dependencies ? Apparently yes.

All our simulations confirm the above formula.

Table 7. XL over $GF(2^7)$ for $D = 5$ (notations as for Table 1)

n	5	5	5	10	10	10
m	5	6	7	10	15	16
D	5	5	5	5	5	5
R	280	336	392	2860	4290	4576
T	252	252	252	3003	3003	3003
$Free$	220	246	250	2365	3002	3002
$Expected$	220	246	250	2365	3002	3002
$\frac{Free}{T-D}$	0.88	0.98	1.00	0.79	0.99	1.00
$Success$			OK			OK

A.4 The Behaviour of XL over $GF(2^k)$ when $D \geq 6, \ldots$

As in [14], it is possible to continue and give formulas for *Free* when $D = 6$ etc. These formulas are expected to predict the behaviour of XL for any D for overdefined systems with $m > n + \varepsilon$. The results given here are very similar than for fields $GF(2)$ in [14], except that in [14] the formulas work also when $m = n$, which is the hard case here.

The exact formula for all D is unknown. This formula is probably not very simple, due to entanglement of linear dependencies: so far we only subtracted linear dependencies, yet for a larger D dependencies among these dependencies will appear, etc. Apparently for XL over $GF(2)$ the exact number of linearly independent equations can be computed from the work of Jean-Charles-Faugère [17,18], extending the so called Buchberger criteria, however we do not know if the problem is solved for XL over $GF(2^k), k > 1$.

Secret Exponent Attacks on RSA-type Schemes with Moduli $N = p^r q$

Alexander May

Faculty of Computer Science, Electrical Engineering and Mathematics
University of Paderborn
33102 Paderborn, Germany
alexx@uni-paderborn.de

Abstract. We consider RSA-type schemes with modulus $N = p^r q$ for $r \geq 2$. We present two new attacks for small secret exponent d. Both approaches are applications of Coppersmith's method for solving modular univariate polynomial equations [5]. From these new attacks we directly derive partial key exposure attacks, i.e. attacks when the secret exponent is not necessarily small but when a fraction of the secret key bits is known to the attacker. Interestingly, all of these attacks work for public exponents e of arbitrary size. Additionally, we present partial key exposure attacks for the value $d_p = d \bmod p-1$ which is used in CRT-variants like Takagi's scheme [11]. Our results show that RSA-type schemes that use moduli of the form $N = p^r q$ are more susceptible to attacks that leak bits of the secret key than the original RSA scheme.

Keywords: $N = p^r q$, Coppersmith's method, Partial Key Exposure Attacks

1 Introduction

We investigate attacks on cryptographic schemes that use public moduli of the form $N = p^r q$ for some constant $r > 1$. Moduli of this type have recently been used in different cryptographic designs. Fujioke, Okamoto and Uchiyama [6] presented an electronic cash scheme using a modulus $N = p^2 q$. Furthermore, Okamoto and Uchiyama [10] designed an elegant public-key crypto scheme that is provably as secure as factoring a modulus $N = p^2 q$. A fast CRT-RSA variant using moduli of the form $N = p^r q$ was introduced by Takagi [11] in 1998. The larger one chooses r in Takagi's scheme, the more efficient is the scheme for a fixed bit-size of the modulus N.

Consider an RSA-type scheme with public key (N, e), where $N = p^r q$ for some fixed $r > 1$ and p, q are of the same bit-size. The secret key d satisfies $ed = 1 \bmod \phi(N)$, where $\phi(N)$ is Euler's totient function. We denote by $\mathbb{Z}^*_{\phi(N)}$ the multiplicative group of invertible integers modulo $\phi(N)$.

In 1999, Boneh, Durfee and Howgrave-Graham [3] showed that schemes with moduli of the form $N = p^r q$ are more susceptible to attacks that leak bits of p than the original RSA-scheme. Using Coppersmith's method for solving

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 218–230, 2004.

univariate modular equations [5], they showed that it suffices to know a fraction of $\frac{1}{r+1}$ of the MSBs of p to factor the modulus. It is an interesting task, whether schemes with $N = p^r q$ are also more susceptible to attacks that leak bits of the secret exponent d. In most side-channel attack scenarios (see for instance [7,8]), it is more reasonable to assume that an adversary gains knowledge of a fraction of the secret key bits than knowledge of the prime factor bits.

Intuitively, one should expect that crypto-systems with moduli of the form $N = p^r q$, $r > 1$ are more vulnerable to secret key attacks than the original RSA-scheme, since for a fixed bit-size of N the amount of secret information encoded in the prime factors is smaller than in RSA. Hence, these schemes should be more susceptible to small secret key attacks like the Wiener attack [12] and the Boneh-Durfee attack [1]. Likewise, these schemes should be more susceptible to so-called partial key exposure attacks that use the knowledge of a fraction of the secret key bits like the Boneh-Durfee-Frankel attack [2] and the Blömer-May attack [4].

In contrast to this intuition, it was stated in the work of Takagi [11] that RSA-type schemes with $N = p^r q$ seem to be less vulnerable to attacks for small decryption exponents d than the original RSA-scheme. Namely, Takagi showed a generalized Wiener-bound of $d \leq N^{\frac{1}{2(r+1)}}$. However, we introduce two attacks with improved bounds for the size of d. Both new attacks are applications of Coppersmith's method for solving modular univariate polynomial equations [5].

Our first attack directly uses the results of Boneh, Durfee and Howgrave-Graham [2] for factoring $N = p^r q$. It yields an improved bound of

$$d \leq N^{\frac{r}{(r+1)^2}} \quad \text{for } r \geq 2.$$

Let us compare the results for $r = 2$: Takagi requires that $d \leq N^{\frac{1}{6}}$ whereas our new method works whenever $d \leq N^{\frac{2}{9}}$.

Our second method makes use of Coppersmith's method in the univariate case and leads to the bound

$$d \leq N^{\left(\frac{r-1}{r+1}\right)^2} = N^{1-\frac{4r}{(r+1)^2}} \quad \text{for } r \geq 2.$$

Interestingly in contrast to the previous bounds, this new bound converges to N for growing r instead of converging to 1. It improves upon our first attack for all parameter choices $r \geq 3$: The second attack requires that $d \leq N^{\frac{1}{4}}$ in the case $r = 3$ compared to $d \leq N^{\frac{3}{16}}$ for our first method. Thus, our first attack is only superior to the other methods in the case $r = 2$. On the other hand, moduli of the form $N = p^2 q$ are frequently used in cryptography and therefore they represent one of the most important cases.

Interestingly, the new attacks for small decryption exponents d have two new features which the original Wiener attack and the Boneh-Durfee attack do not possess:

- One cannot counteract the new attacks by choosing large public exponents e, since the attacks are independent of the value of e. In comparison, the Wiener bound $d \leq N^{\frac{1}{4}}$ and the Boneh-Durfee bound $d \leq N^{0.292}$ require

that $e < \phi(N)$. It is known that the attacks cannot be applied for any size of d if $e > N^{1.5}$ or $e > N^{1.875}$, respectively.
- The new attacks immediately imply a partial key exposure attack for d with known most significant bits (MSBs). Namely, it makes no difference in the attacks whether the most significant bits of d are zero (and thus d is a small decryption exponent) or are known to the attacker. In contrast, Wiener's attack and the Boneh-Durfee attack for small decryption exponents do not work when the MSB's are non-zero but known. In addition, the new attacks also provide partial key exposure attacks for known least significant bits (LSBs).

Using the first attack, we are able to prove that a fraction of

$$1 - \frac{r}{(r+1)^2} \text{ of the MSBs } or \text{ LSBs of } d$$

suffice to find the factorization of $N = p^r q$. The second attack yields partial key exposure attacks that require only a fraction of

$$\frac{4r}{(r+1)^2} \text{ of the MSBs } or \text{ LSBs of } d$$

in order to factor N.

The resulting partial key exposure attacks share the same property as the underlying attacks for small decryption exponents d: They do not rely on the size of the public exponent e. Note that all partial key exposure attacks mentioned in the literature [2,4] are dependent on e and do not work for arbitrary $e \in \mathbb{Z}^*_{\phi(N)}$. The new methods are the first partial key exposure attacks that work for all public exponents e.

The reason that all former attacks on RSA-type schemes depend on the size of e is that they all compute the parameter k in the RSA key equation $ed - 1 = k\phi(N)$. In contrast, our new attacks do not require the computation of k. Thus, k must not be a small parameter and hence the parameters e and d can be increased (thereby increasing k) without affecting the usability of the attacks.

The reason that our new attacks do not require the direct computation of k is mainly that for moduli $N = p^r q$ the group order of the multiplicative group Z^*_N is $\phi(N) = p^{r-1}(p-1)(q-1)$. Thus for $r \geq 2$, $\phi(N)$ and N share the common divisors p and p^{r-1}, respectively, and this can be used in the attacks by constructing polynomials with small roots modulo p (our first attack) and modulo p^{r-1} (our second attack), respectively. But looking at the equation $ed-1 = k\phi(N)$ modulo p (respectively modulo p^{r-1}) removes the unknown parameter k.

We want to point out that these new attacks are normally not a threat to Takagi's scheme [11]. Since Takagi's CRT-decryption process only makes use of the values $d_p = d \bmod p - 1$ and $d_q = d \bmod q - 1$, it suffices to choose an d which satisfies $ed = 1 \bmod (p-1)(q-1)$. For this kind of public-key/secret-key pair (e, d), our previous attacks do not apply. Even worse, normally one would not even store the value of d but only the values of d_p and d_q for the decryption

process. Therefore, it is reasonable to assume that an attacker may only get bits of d_p or d_q. Hence, it is an interesting task to derive partial key exposure attacks for known bits of d_p (respectively d_q).

We show that the partial key exposure attacks of Blömer and May [4] for moduli $N = pq$ generalize to the case $N = p^r q$. Interestingly, the results are again much better for $r > 1$. Namely, we present attacks that need only a fraction of

$$\frac{1}{r+1} \text{ of the MSBs } or \text{ LSBs of } d_p$$

when the public exponent e is small. This shows that Takagi's scheme is also more susceptible to attacks that leak bits of d_p than normal CRT-RSA.

The paper is organized as follows: In Section 2, we review Coppersmith's method for modular univariate polynomial equations [5]. Here, we introduce a reformulation of Coppersmith's orginal theorem that unifies all known applications (see [2,3,4,5]) of the method in the univariate case. As an example, we derive the result of Boneh, Durfee and Howgrave-Graham [3] for factoring $N = p^r q$ as a direct application of Coppersmith's theorem. The first attack for small d and the corresponding partial key exposure attacks are presented in Section 3. In Section 4, we describe our second attack. The partial key exposure attacks for d_p are presented in Section 5.

2 Coppersmith's Method and the Result of BDH

Let us recall Coppersmith's theorem for solving modular univariate polynomial equations [5]. Here, we give the theorem in a slightly more general form than originally stated. However, one can prove the theorem in a completely analogous way to the reasoning in the original proof of Coppersmith. We give the details of the proof in the full version of the paper.

Theorem 1 (Coppersmith) *Let N be an integer of unknown factorization, which has a divisor $b \geq N^\beta$. Let $f_b(x)$ be an univariate, monic polynomial of degree δ. Furthermore, let c_N be a function that is upper-bounded by a polynomial in $\log N$. Then we can find all solutions x_0 for the equation $f_b(x) = 0 \bmod b$ with*

$$|x_0| \leq c_N N^{\frac{\beta^2}{\delta}}$$

in time polynomial in $(\log N, \delta)$.

Coppermith formulated Theorem 1 for the special case where $N = b$. Then the bound for the solutions becomes $|x_0| \leq c_N N^{\frac{1}{\delta}}$. However, the above formulation of Coppersmith's theorem has some advantages: For instance, it is not hard to see that the result of Boneh, Durfee and Howgrave-Graham [3] for factoring $N = p^r q$ with known bits is a direct application of Theorem 1 using the polynomial $f_{p^r}(x) = (x + \tilde{p})^r$.

In fact, the following theorem is stated in the original work of Boneh, Durfee and Howgrave-Graham for the special case $k = 1$, but we formulate it in a slightly more general way, since we will use this generalization in Section 3.

Theorem 2 (BDH) *Let $N = p^r q$, where r is a known constant and p, q are of the same bit-size. Let k be an (unknown) integer that is not a multiple of $p^{r-1}q$. Suppose we know an integer $\tilde{p}$ with*

$$|kp - \tilde{p}| \leq N^{\frac{r}{(r+1)^2}}.$$

Then N can be factored in polynomial time.

Let us interpret the result of Theorem 2. In order to factor N it suffices to find an integer $\tilde{p}$ which is within the range $N^{\frac{r}{(r+1)^2}}$ of some multiple of p (which is not a multiple of N). In the following section, we present our first new attack that constructs an integer $\tilde{p}$ with the above property whenever d is sufficiently small.

3 The Attack Modulo p

We present our first attack for small decryption exponents d and afterwards extend this approach to partial key exposure attacks.

Theorem 3 *Let $N = p^r q$, where $r \geq 2$ is a known constant and p, q are primes of the same bit-size. Let $(e, d) \in \mathbb{Z} \times \mathbb{Z}^*_{\phi(N)}$ be the public-key/secret-key pair satisfying $ed = 1 \mod \phi(N)$. Suppose that*

$$d \leq N^{\frac{r}{(r+1)^2}}.$$

Then N can be factored in probabilistic polynomial time.

Proof: We know that $\phi(N) = p^{r-1}(p-1)(q-1)$ and therefore the key pair (e, d) satisfies the equation

$$ed - 1 = kp^{r-1}(p-1)(q-1) \quad \text{for some } k \in \mathbb{N}. \tag{1}$$

Let E be the inverse of e modulo N, i.e. $Ee = 1 + cN$ for some $c \in \mathbb{N}$. If E does not exist then $\gcd(e, N)$ must be a non-trivial divisor of N.

Note that each possible non-trivial divisor p^s, $p^s q$ or q $(1 \leq s \leq r)$ does immediately yield the complete factorization of N: p^s can be easily factored by guessing s and taking the s^{th} root over the integers. On the other hand, $p^s q$ yields $\frac{N}{p^s q} = p^{r-s}$ which reduces this case to the previous one. Similarly, q gives us p^r.

Hence, let us assume wlog that the inverse E of e modulo N exists. Multiplying equation (1) by E leads to

$$d - E = (Ekp^{r-2}(p-1)(q-1) - cp^{r-1}qd)p.$$

Thus, E is a multiple of p up to an additive error of $d \leq N^{\frac{r}{(r+1)^2}}$. In order to apply Theorem 2, it remains to show that the expression $Ekp^{r-2}(p-1)(q-1) - cp^{r-1}qd$ is not a multiple of $p^{r-1}q$. Since $p^{r-1}q$ divides the second term, this is equivalent

to show that $Ek(p-1)(q-1)$ is not a multiple of pq. By assumption, we have $\gcd(E,N) = 1$ and thus it remains to prove that pq does not divide $k(p-1)(q-1)$. Assume $k(p-1)(q-1) = c'pq$ for some $c' \in \mathbb{N}$. Then equation (1) simplifies to

$$ed - 1 = c'N.$$

On the other hand we know that $eE - 1 = cN$. Combining both equalities we obtain that $d = E \bmod N$. Since $d, E < N$ we have $d = E$ even over $\mathbb{Z}$. It is a well-known fact that the knowledge of the secret key d yields the factorization of N in probabilistic polynomial time (see for instance [9], Chapter 4.6.1).

We briefly summarize our factorization algorithm.

(Mod p)-attack for small d using a modulus $N = p^r q$

INPUT: (N, e), where $N = p^r q$ and $ed = 1 \bmod \phi(N)$ for some $d \leq N^{\frac{r}{(r+1)^2}}$.

1. Compute $E = e^{-1} \bmod N$. If the computation of E fails, output p, q.
2. Run the algorithm of Theorem 2 on input E. If the algorithm's output is p, q then EXIT.
3. Otherwise set $d = E$ and run a probabilistic factorization algorithm on input (N, e, d).

OUTPUT: p, q

Since every step of the algorithm runs in (probabilistic) polynomial time, this concludes the proof of the theorem. □

Theorem 3 gives us a polynomial time factoring algorithm whenever a certain amount of the MSBs of d are zero. The following corollary shows how the proof of Theorem 3 can be easily generalized such that the result does not only hold if the MSBs of d are zero but instead if they are known to the attacker. This gives as a partial key exposure attack for known MSBs with an analogous bound.

Corollary 4 *(MSB) Let $N = p^r q$, where $r \geq 2$ is a known constant and p, q are primes of the same bit-size. Let $(e, d) \in \mathbb{Z} \times \mathbb{Z}^*_{\phi(N)}$ be the public-key/secret-key pair satisfying $ed = 1 \bmod \phi(N)$. Given $\tilde{d}$ such that*

$$|d - \tilde{d}| \leq N^{\frac{r}{(r+1)^2}}.$$

Then N can be factored in probabilistic polynomial time.

Proof: The key-pair (e, d) satisfies the equality

$$e(d - \tilde{d}) + e\tilde{d} - 1 = kp^{r-1}(p-1)(q-1) \quad \text{for some } k \in \mathbb{N}.$$

Let $E = e^{-1} \bmod N$, i.e. $Ee = 1 + cN$ for some $c \in \mathbb{N}$. If E does not exist, we obtain the factorization of N. Multiplying the above equation by E yields

$$(d - \tilde{d}) + E(e\tilde{d} - 1) = (Ekp^{r-2}(p-1)(q-1) - cp^{r-1}q(d - \tilde{d}))p.$$

Thus, $E(e\tilde{d} - 1)$ is a multiple of p up to an additive error of $|d - \tilde{d}| \leq N^{\frac{r}{(r+1)^2}}$. The rest of the proof is completely analogous to the proof of Theorem 3. □

Corollary 4 implies that one has to know roughly a fraction of $1 - \frac{r}{(r+1)^2}$ of the MSBs of d for our partial key exposure attack. We can also derive a partial key exposure attack for known LSBs with an analogous bound.

Corollary 5 *(LSB) Let $N = p^r q$, where $r \geq 2$ is a known constant and p, q are primes of the same bit-size. Let $(e, d) \in \mathbb{Z} \times \mathbb{Z}^*_{\phi(N)}$ be the public-key/secret-key pair satisfying $ed = 1 \bmod \phi(N)$. Given d_0, M with $d = d_0 \bmod M$ and*

$$M \geq N^{1 - \frac{r}{(r+1)^2}}.$$

Then N can be factored in probabilistic polynomial time.

Proof: Let us write $d = d_1 M + d_0$, were the unknown d_1 satisfies $d_1 = \frac{d - d_0}{M} < \frac{N}{M} \leq N^{\frac{r}{(r+1)^2}}$. We have the key equation

$$ed_1 M + ed_0 - 1 = kp^{r-1}(p-1)(q-1) \quad \text{for some } k \in \mathbb{N}.$$

Multiply the equation by $E = (eM)^{-1} \bmod N$. We see that $E(ed_0 - 1)$ is a multiple of p up to an additive error of $|d_1| < N^{\frac{r}{(r+1)^2}}$. The rest of the proof is analogous to the proof of Theorem 3. □

4 Attack Modulo p^{r-1}

Our first attack applied Theorem 2 which in turn uses a polynomial with small roots modulo p. In our second attack we will construct a polynomial with a small root modulo p^{r-1} and directly apply Coppersmith's method in the univariate case (Theorem 1). This approach yields better results than the first one whenever $r \geq 3$.

Theorem 6 *Let $N = p^r q$, where $r \geq 2$ is a known constant and p, q are primes of the same bit-size. Let $(e, d) \in \mathbb{Z} \times \mathbb{Z}^*_{\phi(N)}$ be the public-key/secret-key pair satisfying $ed = 1 \bmod \phi(N)$. Suppose that*

$$d \leq N^{\left(\frac{r-1}{r+1}\right)^2}.$$

Then N can be factored in probabilistic polynomial time.

Proof: The key pair (e, d) satisfies the equation

$$ed - 1 = kp^{r-1}(p-1)(q-1) \quad \text{for some } k \in \mathbb{N}.$$

Let E be the inverse of e modulo N, i.e. $Ee = 1 + cN$ for some $c \in N$. In the case that E does not exist, $\gcd(e, N)$ yields the complete factorization of N as shown in the proof of Theorem 3. Multiplying our equation by E leads to

$$d - E = (Ek(p-1)(q-1) - cdpq)p^{r-1}.$$

This gives us a simple univariate polynomial

$$f_{p^{r-1}}(x) = x - E$$

with the root $x_0 = d$ modulo p^{r-1}.

Thus, we have a polynomial $f_{p^{r-1}}$ of degree $\delta = 1$ with a root x_0 modulo p^{r-1}. In order to apply Theorem 1, we have to find a lower bound for p^{r-1} in terms of N.

Since p and q are of the same bit-size, we know that $p \geq \frac{1}{2}q$. Hence $p^{r-1} = \frac{N}{pq} \geq \frac{N}{2p^2}$. This gives us

$$p^{r-1} \geq \left(\frac{1}{2}N\right)^{\frac{r-1}{r+1}} \geq \frac{1}{2}N^{\frac{r-1}{r+1}}.$$

Thus, we can choose $\beta = \frac{r-1}{r+1} - \frac{1}{\log N}$ and apply Theorem 1 with the parameter choice β, δ and $c_N = 4$. We can find all roots x_0 that are in absolute value smaller than

$$4N^{\frac{\beta^2}{\delta}} = 4N^{(\frac{r-1}{r+1})^2 - \frac{2(r-1)}{(r+1)\log N} + \frac{1}{\log^2 N}} \geq 4N^{(\frac{r-1}{r+1})^2 - \frac{2}{\log N}} = N^{(\frac{r-1}{r+1})^2}.$$

Hence, we obtain the value $x_0 = d$. We can run a probabilistic factorization algorithm on input (N, e, d) in order to obtain the factorization of N in expected polynomial time.

Remark 7 *Another (deterministic) polynomial time method to find the factorization of N could be the computation of* $\gcd(ed - 1, N)$. *Since* $ed - 1 = kp^{r-1}(p-1)(q-1)$, *the computation yields a non-trivial divisor of N iff pq does not divide* $k(p-1)(q-1)$, *which is unlikely to happen. As shown in the proof of Theorem 3, a non-trivial divisor of N reveals the complete factorization of the modulus. So in practice, one might try this alternative gcd-method first and if it fails, one applies a probabilistic algorithm on the key-pair* (N, e, d).

Let us summarize our new factorization algorithm.

(Mod p^r)-attack for small d using a modulus $N = p^r q$

INPUT: (N, e), where $N = p^r q$ and $ed = 1 \bmod \phi(N)$ for some $d \leq N^{\left(\frac{r-1}{r+1}\right)^2}$.

1. Compute $E = e^{-1} \bmod N$. If E does not exist, compute $\gcd(e, N)$ and output p, q.
2. Apply the algorithm of Theorem 1 on input N, $f_{p^{r-1}} = x - E$, $\beta = \frac{r-1}{r+1} - \frac{1}{\log N}$ and $c_N = 2$. This gives us the value d.
3. If the computation $\gcd(ed - 1, N)$ yields the factorization, EXIT.
4. Run a probabilistic factorization algorithm on input (N, e, d).

OUTPUT: p, q

Every step of the algorithm can be computed in probabilistic polynomial time, which concludes the proof of Theorem 6 □

Similar to the first attack (the (Mod p)-attack) for small decryption exponent d, we can also easily derive partial key exposure attacks for the new attack of Theorem 6. The proof of Theorem 6 shows that in order to find the factorization of N, it suffice to find a linear, univariate polynomial $f_{p^{r-1}}(x) = x + c$ with a root $x_0, |x_0| \leq N^{\left(\frac{r-1}{r+1}\right)^2}$ modulo p^{r-1}.

We will show that this requirement is satisfied in the following partial key exposure attacks. Instead of using small decryption exponents $d < N^{\left(\frac{r-1}{r+1}\right)^2} = N^{1-\frac{4r}{(r+1)^2}}$, the attacker has to know a fraction of roughly $\frac{4r}{(r+1)^2}$ of the bits of N in order to succeed.

Corollary 8 *(MSB) Let $N = p^r q$, where $r \geq 2$ is a known constant and p, q are primes of the same bit-size. Let $(e, d) \in \mathbb{Z} \times \mathbb{Z}^*_{\phi(N)}$ be the public-key/secret-key pair satisfying $ed = 1 \bmod \phi(N)$. Given $\tilde{d}$ with*

$$|d - \tilde{d}| \leq N^{\left(\frac{r-1}{r+1}\right)^2}.$$

Then N can be factored in probabilistic polynomial time.

Proof: We know that

$$e(d - \tilde{d}) + e\tilde{d} - 1 = 0 \bmod \phi(N),$$

and $\phi(N)$ is a multiple of p^{r-1}. Multiply the equation by $E = e^{-1} \bmod N$, which gives us the desired linear polynomial

$$f_{p^{r-1}}(x) = x + E(e\tilde{d} - 1)$$

with the small root $x_0 = d - \tilde{d}$, $|x_0| \leq N^{\left(\frac{r-1}{r+1}\right)^2}$ modulo p^{r-1}. The rest of the proof is analogous to the proof of Theorem 6. □

In a similar fashion, we derive a partial key exposure attack for known LSBs.

Corollary 9 *(LSB) Let $N = p^r q$, where $r \geq 2$ is a known constant and p, q are primes of the same bit-size. Let $(e, d) \in \mathbb{Z} \times \mathbb{Z}^*_{\phi(N)}$ be the public-key/secret-key pair satisfying $ed = 1 \bmod \phi(N)$. Given d_0, M with $d = d_0 \bmod M$ and*

$$M \geq N^{\frac{4}{(r+1)^2}}.$$

Then N can be factored in probabilistic polynomial time.

Proof: Let us write $d = d_1 M + d_0$. Then the unknown parameter satisfies $d_1 < \frac{N}{M} \leq N^{\left(\frac{r-1}{r+1}\right)^2}$. For the key-pair (e, d) we have

$$e(d_1 M + d_0) - 1 = 0 \bmod \phi(N),$$

where $\phi(N)$ is a multiple of p^{r-1}. Multiplying this equation by $E = (eM)^{-1}$ modulo N gives us the desired linear polynomial

$$f_{p^{r-1}}(x) = x + E(ed_0 - 1)$$

with the small root d_1 modulo p^{r-1}. The rest of the proof is analogous to the proof of Theorem 6. □

5 Partial Key Exposure Attacks for $d_p = d$ Modulo $p - 1$

The partial key exposure attacks that we consider in this section for moduli $N = p^r q$ can be considered as a generalization of the results of Blömer and May [4]. The attacks are an application of the theorem of Boneh, Durfee and Howgrave-Graham (Theorem 2).

We derive simple partial key exposure attacks for small public exponents e in both cases: known MSBs and known LSBs. The new attacks are a threat to schemes that use CRT-decoding (for instance Takagi's scheme [11]) in combination with small public exponents.

Let us state our LSB-attack.

Theorem 10 *Let $N = p^r q$, where $r \geq 1$ is a known constant and p, q are primes of the same bit-size. Let e be the public key and let d_p satisfy $ed_p = 1 \bmod p-1$. Given d_0, M with $d_0 = d_p \bmod M$ and*

$$M \geq 2N^{\frac{1}{(r+1)^2}}.$$

Then N can be factored in time $e \cdot poly(\log(N))$.

Proof: Let us consider the RSA key equation

$$ed_p - 1 = k(p-1) \quad \text{for some } k \in \mathbb{Z}.$$

Since $d_p < (p-1)$, we obtain the inequality $k < e$. Let us write $d_p = d_1 M + d_0$. We can bound the unknown d_1 by $d_1 < \frac{p}{M} \leq N^{\frac{r}{(r+1)^2}}$. Our equation above can be rewritten as

$$ed_1 M + ed_0 + k - 1 = kp.$$

Compute the inverse E of eM modulo N, i.e. $EeM = 1 + cN$ for some $c \in \mathbb{N}$. If E does not exist, we obtain from $\gcd(eM, N)$ the complete factorization of N as shown in Theorem 3. Multiplying our equation with E leaves us with

$$d_1 + E(ed_0 + k - 1) = (Ek - cp^{r-1}qd_1)p.$$

Thus, $E(ed_0 + k - 1)$ is a multiple of p up to some additive error $d_1 \leq N^{\frac{r}{(r+1)^2}}$. Since the parameter k is unknown, we have to do a brute force search for k in the interval $[1, e)$. In order to apply Theorem 2, it remains to show that the term $(Ek - cp^{r-1}qd_1)$ is not a multiple of $p^{r-1}q$. This is equivalent to the condition that $p^{r-1}q$ does not divide Ek, but we know that $\gcd(E, N) = 1$ and thus $p^{r-1}q$ must not divide k. But $p^{r-1}q$ cannot divide k in the case $e \leq p^{r-1}q$ and otherwise we can easily check the condition by computing $\gcd(k, N)$ for every possible k. The algorithm of Theorem 2 yields the factorization of N for the correct guess of k.

We briefly summarize our factorization algorithm.

Algorithm LSB-Attack for d_p and moduli $N = p^r q$

INPUT: $-$ (N, e), where $N = p^r q$ and d_p satisfies $ed_p = 1 \bmod p - 1$
$-$ d_0, M with $d_0 = d_p \bmod M$ and $M \geq 2N^{\frac{1}{(r+1)^2}}$

1. Compute $E = (eM)^{-1} \bmod N$. If the computation of E fails, find the factors p, q of N using $\gcd(eM, N)$.
2. FOR $k = 1$ TO e
   (a) If $\gcd(k, N) > 1$ find the factors p, q.
   (b) Run the algorithm of Theorem 2 on input $E(ed_0 + k - 1)$. If the algorithm's output is p, q then EXIT.

OUTPUT: p, q

The running time of the algorithm is $e \cdot \text{poly}(\log N)$, which concludes the proof. □

Note that our method from Theorem 10 is polynomial time for public exponents of the size $\text{poly}(\log(N))$ and requires only a $\frac{1}{(r+1)^2}$-fraction of the bits (in

terms of the size of N), which is a $\frac{1}{r+1}$-fraction of the bits of d_p. The following theorem gives us a similar result for partial key exposure attacks with known MSBs, but in contrast the method is polynomial time for all public exponents $e < N^{\frac{r}{(r+1)^2}}$.

We show that an approximation of d_p up to $N^{\frac{r}{(r+1)^2}-\alpha}$ suffices to find the factorization of N. Note that d_p is of size roughly $N^{\frac{1}{r+1}}$. Hence in the case $\alpha = 0$, a fraction of $\frac{1}{r+1} - \frac{r}{(r+1)^2} = \frac{1}{(r+1)^2}$ of the bits is enough (in terms of the size of N).

Theorem 11 *Let $N = p^r q$, where $r \geq 1$ is a known constant and p, q are primes of the same bit-size. Let $e = N^\alpha$, $\alpha \in [0, \frac{r}{(r+1)^2}]$ be the public key and let d_p satisfy $ed_p = 1 \bmod p-1$. Given $\tilde{d}$ with*

$$|d_p - \tilde{d}| \leq N^{\frac{r}{(r+1)^2}-\alpha}.$$

Then N can be factored in polynomial time.

Proof: We know that

$$ed_p - 1 = k(p-1) \quad \text{for some } k \in \mathbb{N},$$

with $k < e$. The term $e\tilde{d}$ is an approximation of kp up to an additive error of

$$|kp - e\tilde{d}| = |e(d_p - \tilde{d}) + k - 1| \leq |e(d_p - \tilde{d})| + |k-1|$$

$$\leq N^{\frac{r}{(r+1)^2}} + N^\alpha \leq 2N^{\frac{r}{(r+1)^2}}.$$

Thus, one of the terms $e\tilde{d} \pm N^{\frac{r}{(r+1)^2}}$ satisfies the bound of Theorem 2. Note that the algorithm of Theorem 2 can be applied since $k < e < N^{\frac{r}{(r+1)^2}}$ and thus k cannot be a multiple of $p^{r-1}q = \Omega(N^{\frac{r}{r+1}})$.

Let us briefly summarize the factorization algorithm.

MSB-Attack for d_p and moduli $N = p^r q$

INPUT: – (N, e), where $N = p^r q$ and d_p satisfies $ed_p = 1 \bmod p-1$
– $\tilde{d}$ with $|d_p - \tilde{d}| \leq N^{\frac{r}{(r+1)^2}-\alpha}$, where $\alpha = \log_N(e)$.

1. Compute $\tilde{p} = e\tilde{d}$.
2. Run the algorithm of Theorem 2 on input $\tilde{p} + N^{\frac{r}{(r+1)^2}}$. If the algorithm's output is p, q then EXIT.
3. Otherwise run the algorithm of Theorem 2 on input $\tilde{p} - N^{\frac{r}{(r+1)^2}}$.

OUTPUT: p, q

The algorithm runs in time polynomial in log*(N),* which concludes the proof. □

References

1. D. Boneh, G. Durfee, "Cryptanalysis of RSA with private key d less than $N^{0.292}$", IEEE Trans. on Information Theory, Vol. 46(4), 2000
2. D. Boneh, G. Durfee, Y. Frankel, "An attack on RSA given a small fraction of the private key bits", Advances in Cryptology - AsiaCrypt '98, Lecture Notes in Computer Science Vol. 1514, Springer-Verlag, pp. 25–34, 1998
3. D. Boneh, G. Durfee, and N. Howgrave-Graham, "Factoring $N = p^r q$ for large r", Advances in Cryptology - Crypto '99, Lecture Notes in Computer Science Vol. 1666, Springer-Verlag, pp. 326–337, 1999
4. J. Blömer, A. May, "New Partial Key Exposure Attacks on RSA", Advances in Cryptology - Crypto 2003, Lecture Notes in Computer Science Vol. 2729, pp. 27–43, Springer Verlag, 2003
5. D. Coppersmith, "Small solutions to polynomial equations and low exponent vulnerabilities", Journal of Cryptology, Vol. 10(4), pp. 223–260, 1997.
6. A. Fujioke, T. Okamoto, Miyaguchi, "ESIGN: An Efficient Digital Signature Implementation for Smartcards", Advances in Cryptology - Eurocrypt '91, Lecture Notes in Computer Science Vol. 547, Springer Verlag, pp. 446-457, 1991
7. P. Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS and other systems", Advances in Cryptology - Crypto '96, Lecture Notes in Computer Science Vol. 1109, Springer Verlag, pp. 104–113, 1996
8. P. Kocher, J. Jaffe and B. Jun, "Differential power analysis", Advances in Cryptology – Crypto '99, Lecture Notes in Computer Science Vol. 1666, Springer Verlag, pp. 388–397, 1999
9. D. Stinson, "Cryptography Theory and Practice", Second Edition, CRC Press, 2002
10. T. Okamoto, S. Uchiyama, "A new public key cryptosystem as secure as factoring", Advances in Cryptology - Eurocrypt '98, Lecture Notes in Computer Science Vol. 1403, Springer Verlag, pp. 308–318, 1998
11. T. Takagi, "Fast RSA-type cryptosystem modulo $p^k q$", Advances in Cryptology - Crypto '98, Lecture Notes in Computer Science Vol. 1462,Springer-Verlag, pp.318–326, 1998
12. M. Wiener, "Cryptanalysis of short RSA secret exponents", IEEE Transactions on Information Theory, Vol. 36, pp. 553–558, 1998.

General Group Authentication Codes and Their Relation to "Unconditionally-Secure Signatures"

Reihaneh Safavi–Naini[1], Luke McAven[1], and Moti Yung[2]

[1] School of Information Technology and Computer Science,
University of Wollongong, Wollongong 2522, Australia.
[rei,lukemc]@uow.edu.au
[2] Department of Computer Science,
Columbia University, New York, NY 10027, USA.
moti@cs.columbia.edu

Abstract. Strong notions of security for unconditionally secure digital signature schemes (USDS) were recently proposed where security is defined based on notions of security in computationally–secure digital signatures. The traditional area of unconditionally secure authentication, however, is that of "authentication codes" (*A*–codes). Relations between primitives is central to cryptographic research. To this end, we develop a novel "general group–based *A*–code" framework which includes known types of group *A*–codes and their extensions, including the newly proposed USDS, and also allows other models to be systematically described and analysed. In particular, information theoretic analysis of these codes can be applied to USDS, establishing fundamental bounds on USDS parameters.

A second contribution herein is a modular algebraic method of synthesising group codes from simpler *A*–codes, such that security of the group code follows directly from the component codes. We demonstrate our approach by constructing and analysing a USDS satisfying the 'strongest security notion'.

1 Introduction

Digital signatures are the basic authentication primitive in modern cryptography. They are known to be equivalent to the existence of one–way functions, and thus to rely on computational assumptions [12]. There are, however, settings where reliance on computational assumptions is inappropriate (typically for small mutually distrusting groups of entities that do not know each others computational or technological advantages, e.g. advances in quantum computations, as is the setting between nations). The alternative model for secure authentication, when there is no assumption regarding adversaries computational power, has been *A*–codes as suggested by Simmons [16]. This was indeed motivated by authentication procedures between the USA and USSR regarding treaty verification.

In recent years a number of unconditionally secure digital signature schemes, both in interactive [2] and non–interactive settings, have been proposed. We consider a non–interactive setting where a trusted Key Distribution Centres (KDC)

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 231–247, 2004.

or trusted authority (TA) generates and distributes the key information of system participants. The two main approaches satisfying these assumptions are due to Johansson (J99) [11], who considered a variant of multireceiver authentication codes with an untrusted sender and an arbiter, and called it *Unconditionally Secure Digital Signature (USDS),* and Hanaoka, Shikata, Zheng, and Imai [7, 15] (referred to as HSZI00 and SHZI02, respectively) who recently proposed a range of new security notions for USDS. In Eurocrypt 2002 [15], the authors formalised their approach independent of the theory of *A*–codes, and proposed the 'strongest notion' of security for USDS without reference to these codes. They constructed a USDS that provided the 'strongest security notion'.

To understand these proposals we develop a unified framework allowing evaluation of USDS schemes within the domain of *A*–codes. We view the work and scenarios of HSZI00/SHZI02 as providing motivation for studying *A*–code generalisations. One may mistake the use of new notions in HSZI00/SHZI02 to mean extensions of this theory cannot capture the new settings (and that perhaps a new type of theory, similar to that for conditionally secure signatures, is needed). We believe our work puts things in order, in this respect.

A second contribution of this paper is proposing a modular algebraic method for synthesising group *A*–codes. This is particularly important because constructing *A*–codes for complex authentication scenarios can become a formidable task and approaches that allow 're–use' of proven secure schemes as building blocks will provide an attractive option.

Review of *A*-codes

Unconditionally secure authentication codes were first constructed in [6] and then modelled and analysed in [16]. The original *A*–codes were symmetric key primitives for communication between two honest participants, secure against spoofing attacks. Simmons derived the first information theoretic bound on impersonation, bounds on higher order spoofing were later obtained [10, 18].

Simmons [17] also considered A^2–codes in which sender and receiver are distrusted. The sender may *deny* a sent message, and a receiver may substitute a received message or try to ascribe a message to the sender. A^2–codes are asymmetric primitives in which the sending and receiving keys differ. Simmons showed the need for a trusted *arbiter,* with the key information of the sender and receiver, to resolve disputes. In A^3–codes [1, 3] the trust in the arbiter is reduced and the arbiter may attempt to construct fraudulent messages.

Group–based *A*–codes were introduced by [4] and extended by [5, 11, 13, 14]. In multireceiver *A*–codes *(MRA)* [4] a sender constructs an authenticated message that is verifiable by each member of a verifier group. The sender is trusted but receivers may collude to construct a fraudulent message on behalf of the sender. In (J99) [11] the senders are distrusted, and the resulting system was called an *Unconditionally Secure Digital Signature (USDS).* In this model the sender may deny his constructed message. We call this model an MRA^2–code, since the trust assumption is most similar to A^2–codes.

Requirements of a USDS

In a USDS scheme signers and verifiers are distrusted. They may try to forge signed messages, or repudiate their own signed messages. Let $\mathcal{U}$ denote a set of distrusted participants. Any $U_i \in \mathcal{U}$ can sign a message that is verifiable by all $U_j \in \mathcal{U}$. An important property of standard digital signatures is that if U_i obtains a signed message from U_j he can convince U_ℓ that the message is from U_j; this is called transferability. We require the following properties to be satisfied.

Transferability: U_j can convince any $U_k \in \mathcal{U}, k \neq \{i, j\}$, the message is from U_i.
Unforgeability: A colluding subset $C \subset \mathcal{U}$ has a negligible probability of constructing a fraudulent message that is acceptable by a group member U_k as signed by U_i, where:

(i) $U_i \in C$, and can deny the message, *(non–repudiation).*
(ii) $U_i \notin C$, and the message is not generated by U_i.

These properties match the requirements of the first unconditionally secure signature (interactive) protocol [2], and are closest to those achieved in computationally secure signature schemes. An important difference between computationally and unconditionally secure digital signatures is that in USDS verification cannot be a public process, and so secret keys are needed, which, as noted in [11, 13, 15], must be different for each group member.

Our Results

We propose a common framework for modelling and analysing asymmetric group *A*–codes and USDS schemes. We introduce *authentication* and *verification oracles* which adversaries interact with to obtain spoofing information. We also introduce *authentication scenarios* and outline a general way of expressing security goals and adversary's power. We give a generalised bound that applies in such scenarios. Our work suggests numerous variations on defining security goals of a group–based *A*–code and adversaries power. Critically, the framework allows information theoretic *security* and *efficiency* evaluations for USDS.

We also propose a methodical approach to synthesising complex group–based USDS systems with provable security, starting from simple component systems with provable security. This approach is algebraic and while sometimes providing less efficient constructions it avoids some disadvantages of combinatorial synthesis. Furthermore, security proofs follow from security of components.

The rest of the paper is organised as follows: In section 2 we recall parts of *A*–code theory. In section 3 we propose our model of asymmetric group authentication codes (USDS) and show how previous USDS models fit in this framework. Section 4 contains the new design methodology with concrete constructions, while section 5 sketches our general framework for group authentication. Finally, section 6 contains our concluding comments.

2 Preliminaries

An authentication code may be represented as a 4–tuple, $C = (\mathcal{S}, \mathcal{M}, \mathcal{E}, f)$, where $\mathcal{S}, \mathcal{M}, \mathcal{E}$ are the sets of source states, messages and keys, respectively. The function $f : \mathcal{S} \times \mathcal{E} \rightarrow \mathcal{M}$ takes a *source state* s, a key e and generates the corresponding *message* m. The function f defines two algorithms; an *authentication algorithm* used by the sender to generate an authenticated message, and a *verification algorithm* used by the receiver to verify a received message. There is also a *key generation* algorithm that generates key information for the system. We use *systematic Cartesian A*–codes, wherein the messages are of the form (s, t), where the tag t is used to authenticate the source state s. Such an authentication code is represented as a 3–tuple $C = (\mathcal{S}, \mathcal{A}, \mathcal{E})$ with $t = e(s), e \in \mathcal{E}, s \in \mathcal{S}$, and $\mathcal{A}$ being the set of tags (or authenticators). *A*–codes are symmetric key systems and the secret key is shared by sender and receiver, who are assumed to be trusted.

An attacker may inject a fraudulent message into the system (an *impersonation attack),* or construct a fraudulent message m' after observing a valid message m (a *substitution attack*). In both cases the attacker succeeds if the fraudulent message is accepted. The best success probability of the attacker in the two attacks are denoted P_I and P_S, respectively. A message m is *valid* for a key e if $m \in \mathcal{M}(e)$, where e is the key shared by the sender and receiver. Security of an *A*–code is defined by the attackers best success probability in the attacks.

$$P_I = \max_{m \in M} p(m \text{ is valid for } e) \qquad P_S = \max_{m' \in M \setminus \{m\}} p(m' \text{ is valid for } e | m) \ .$$

An *A*–code has ϵ–*security* if the success probability of any attack is at most ϵ.

In A^2–codes one considers signer's *denial attack* and receiver's *impersonation* and *substitution* attacks. In A^3–codes [1, 3] fraud by the arbiter is treated also.

Authentication systems may provide security for more than one message. In *spoofing of order* t, the attackers have access to up to t authenticated messages. Order 0 and 1 spoofing are impersonation and substitution, respectively. Codes that provide security for t–messages are denoted as tA, tA^2 and tA^3–codes.

Efficiency parameters of an *A*–code include participants key sizes, and the length of the authenticator. Performance bounds provide fundamental limits on these parameters for a given level of security, or alternatively bound the security level for a given set of parameters. Two types of bounds are derived for *A*–codes: *information theoretic bounds* on the success probability of attacks in terms of information theoretic measures, and *combinatorial bounds* on the sizes of the key spaces and authenticator in the system. Information theoretic bounds for *A*–codes were given in [6, 16] and later derived for other models [11].

Group–based *A*–codes (the subject of this work) were introduced in [4] and developed by numerous authors [5, 11, 13, 14]. *Multireceiver A–codes (MRA–codes)* allow a single trusted sender to send a message to a group of receivers such that each receiver can individually verify the message. A (ϵ, w, n)–MRA–code is an *MRA*–code for which the success probability of the best attack (impersonation and substitution) for a colluding group of w verifiers is less than ϵ. Information theoretic bounds and constructions for such codes are given in [13].

2.1 Constructions

Numerous constructions of A–codes have been proposed (for example [4, 13, 14, 16]). We briefly recall constructions to be used in this paper.

Polynomial A–code [4] (C_0) Consider the A–code defined by the function $f(x) = a + bx$, where $(a, b) \in \mathbb{F}_q^2$ is the key and the authenticator for the source state $s \in \mathbb{F}_q$ is given by $f(s)$. This code satisfies $P_I = P_S = 1/q$.

Polynomial (ϵ, w, n)–MRA–code [4] (C_1) The sender has two polynomials $f(x)$ and $g(x)$, both of degree at most w, with coefficients over F_q, the finite field with q elements. Each receiver U_i is given $(u_i, f(u_i), g(u_i))$, where $u_i \in \mathbb{F}_q$ is public and $u_i \neq u_j, i \neq j$. To authenticate a source state s, the sender constructs the tag $\alpha(x) = f(x) + sg(x)$ and appends it to s. The receiver U_i accepts a message $(s, \alpha(x))$ as authentic if $f(u_i) + sg(u_i) = \alpha(u_i)$. The construction has $\epsilon = 1/q$ and is *optimal* with respect to tag length, and key sizes.

3 Asymmetric Authentication in Groups: USDS

We consider systems where no participant is trusted (except, possibly the arbiter), and where participants' keys are only known to themselves, hence the term asymmetric. We focus on single signer schemes.

3.1 A General Framework for Single Signer Group A-codes

There is a set $\mathcal{U} = \{U_0, U_1, \ldots, U_n, U_A\}$ of distrusted participants, each with secret key information. The set $\mathcal{U}$ contains n verifiers, an arbiter U_A, and a signer U_0. A message signed by U_0 is acceptable to all verifiers. We assume the arbiter has the algorithm and key information of a verifier, so the arbiter's key information is the same as a verifier's. Arbitration is performed by applying the verification algorithm to a 'suspect' signed message and using the result to resolve the dispute following arbitration rules.

Each user has a distinct identity encoded in the source state: for example the source state can be the concatenation of the user's identity and the information signed. The signer wants to sign $s \in \mathcal{S}$ so any verifier can verify the signature.

The adversary can corrupt a group C, of at most w verifiers, and possibly the signer and/or the arbiter. This is the model in earlier group–based A–codes and USDS. Including the arbiter assesses security under extreme attack conditions. One assumes, however, the arbiter follows the correct arbitration rules.

We consider the following types of attacks.

1. $U_0 \in C$. A *denial attack* where U_0 signs a message, then denies it. Colluders succeed if, following arbitration, the message is deemed not from U_0.
2. $U_0 \notin \mathcal{U}$. In this case the attack is one of the following types.
   - *spoofing attack:* The collusion constructs a message valid for a verifier.
   - *framing attack:* The colluders construct a message attributable to U_0 and acceptable to a verifier. We note the verifier, in this case, may be part of the collusion.

In spoofing attacks colluders succeed if their fraudulent message is acceptable to a target verifier. The message may or may not be valid for (constructible by) U_0.

We remark that HSZI00 introduced an attack against transferability, called 'transfer with a trap'. We show in section 3.2 that this attack has less chance of success than the above attacks and therefore need not be considered separately.

The above requirements are reminiscent of *MRA*–codes and thus we will use the term MRA^2–codes and MRA^3–codes when the arbiter is, or is not, trusted. With a trusted arbiter, a signer's denial attack succeeds if the colluders construct a message m where $m \notin \mathcal{M}(e_T)$, e_T being the key, e_A the arbiter's key distinct from all e_i, which denote the key of U_i.

We use E_i, E_T, E_A and E_C to denote sets of keys associated with verifier U_i, signer U_0, arbiter U_A, and collusion E_C, respectively. The success in denial attacks can be measured by the probability of a verifier U_i accepting the message, $m \in \mathcal{M}(e_i)$, but the arbiter not, i.e., $m \notin \mathcal{M}(e_A)$. In verifier's spoofing attack the message must be valid for a verifier U_j and so $m \in \mathcal{M}(e_j)$, while in verifier's framing attack $m \in \mathcal{M}(e_T)$ and $m \in \mathcal{M}(e_i)$ for some verifier U_i.

Security of an MRA^2 code against the above attacks can be defined using probabilities, $P_D^{t_{V_1},t_{V_2}}$, $P_{RS}^{t_A,t_{V_1},t_{V_2}}$, and $P_{RS}^{t_A,t_{V_1},t_{V_2}}$. In the first attack the collusion includes the signer, in the last two it does not. Each probability is obtained as the best success probability of colluders. The superscripts represent colluders ability to collect information on uncorrupted verifiers' keys by oracle interaction.

Colluders Information

Colluders have their key information. In traditional A–codes colluders may also have access to prior *authenticated messages* sent over the channel. We model such observations by queries to oracles that implement users algorithms with users key information. We consider two types of oracles.

Authentication oracles (A–oracles) implement the authentication algorithm with the signer's key. When presented with an *Authentication query (A–query),* consisting of a source state $s \in \mathcal{S}$, the A–oracle generates the signed message $m = (s, t)$ (or just the signature t).

The impersonation and substitution attacks in traditional A–codes correspond to the case that 0 and 1 A–queries are allowed, respectively.

Verification oracles (V–oracles) implement the verification oracle with a particular verifier's key (as in SHZI02). On input (s, t), the V–oracle generates a TRUE/FALSE result. The queries to this oracle are called *V–queries.*

If the arbitration algorithm is different for the verifier's algorithm, we also need to consider an arbitration oracle.

In symmetric A–codes, A–oracles and V–oracles have the same information; i.e. they implement the same algorithm with the same keys but in asymmetric systems, the oracles have different keys.

A V–query against a verifier U_i gives information about the verification key of U_i. If verifiers use the same verification algorithm with different keys chosen using

the same algorithm (for example random selection with uniform distribution), then the average information from a query will be the same for the two queried verifiers.

Attacks will be against a *target verifier*. The V–queries against this verifier will intuitively be expected to be more 'useful' than a query against a non–targeted verifier. Thus we define **Type V_1–queries (V_2–queries)**] as being made to a non–targeted (targeted) verifier.

Security Evaluation

Let $e_C = \{e_j : j \in C\}$ be the colluders key set. $P_D^{t_{V_1},t_{V_2}}$, $P_{RS}^{t_A,t_{V_1},t_{V_2}}$ and $P_{RF}^{t_a,t_{V_1},t_{V_2}}$ denote success probabilities given t_A A–queries, t_{V_1} V_1–queries to each non–targeted verifier and t_{V_2} V_2–queries. Let $Q(t_A, t_{V_1}, t_{V_2})$ and $R(t_A, t_{V_1}, t_{V_2})$ denote the sequence of queries and responses, respectively and let $(Q,R)(t_A, t_{V_1}, t_{V_2})$ denote the pair of queries and responses.

$$P_D^{t_{V_1},t_{V_2}} = \max_{U_i} \max_{C \subset \mathcal{U}} \max_{\substack{e_C, m \\ m \notin M(e_t) \\ Q(t_{V_1},t_{V_2})}} P(m \text{ is valid for } U_i,\ \text{invalid for } U_A | e_C, (Q,R)(t_{V_1}, t_{V_2}))$$

$$P_{RS}^{t_A,t_{V_1},t_{V_2}} = \max_{\substack{C \subset \mathcal{U} \\ U_i \notin C}} \max_{e_C, m} \quad P(m \text{ is valid for } U_i | e_C, (Q,R)(t_A, t_{V_1}, t_{V_2}))$$

$$P_{RF}^{t_A,t_{V_1},t_{V_2}} = \max_{\substack{C \subset \mathcal{U} \\ U_i}} \max_{e_C, m} \quad P(m \text{ is valid for } U_0 | e_C, (Q,R)(t_A, t_{V_1}, t_{V_2}))$$

We say a system is $(\epsilon, w, n, t_A, t_{V_1}, t_{V_2})$–*secure* if the success chance of the best attack when t_A queries of type A, t_{V_1} queries of type V_1 and t_{V_2} queries of type V_2 are allowed, is at most ϵ.

Adaptive and non–adaptive queries

In the model we allow the queries to be asked in an arbitrary order. The success probability considers all possible interactions involving t A–queries, t_{V_1} V_1–queries and t_{V_2} V_2–queries and is maximised as the attacker's best strategy.

MRA^3–codes are similarly defined but the arbiter may in the collusion. In our model we assume the arbiter has the key information of a verifier. This means security of an MRA^3–code against a collusion containing U_A and w verifiers can be achieved by a $(\epsilon, w+1, n)$–MRA^2–code. Generally, success probability of the collusion attacks involving a dishonest arbiter must be considered.

A–queries and *V*–queries

Although distinguishing among the query type is important for efficiency of constructions, we can guarantee some security against V–queries even if we only consider A–queries. The following Lemma shows protection against V_1–queries can be obtained by constructing codes providing protection against larger collusions.

Lemma 1. *An $(\epsilon, w, n, t, 0, 0)$–$MRA^2$ provides ϵ–security against collusions of size $w - v$, assuming colluders can have t A–queries and any number of V_1–queries against v verifiers.*

This result follows since the information gained by V_1–queries to U_i at most equals the key held by U_i, which would be yielded up were U_i in the collusion,

V_2–queries provide information on the target verifier's key. For secure codes, one expects to obtain less information from queries resulting in FALSE compared to those giving TRUE. This is since the probability of the former type of queries is expected to be higher than that of the latter.

3.2 Security Notions in HSZI00 and SHZI02

One main aim of developing our framework is to unify USDS, including SHZI02. We address this here. HSZI00 correctly recognised the inadequacy of *MRA* and *DMRA*–codes as USDS and argued that multireceiver *A*–codes make sense only in a broadcast environment [7, p.132] and [8, p.69].

The term 'multireceiver' in the *A*–code context refers to the property: *any receiver who receives the authenticated message can verify it.* This is exactly as required in signature schemes. Multireceiver schemes do not 'require' that the signed message be received simultaneously by all group members. Rather they guarantee that *if* any group member receives the signed message then they can verify it. However, as noted earlier, *MRA*–systems assume a trusted sender and so do not provide security against attacks by collusions including a distrusted signer. The model proposed in section 3.1 assumes the signer is distrusted.

The following Lemma shows we need not consider 'transfer with a trap' (so named by HSZI00) attack. In a 'transfer with a trap' colluders construct a forged message that is acceptable to U_i and not U_j or U_A, and so when U_i presents the message to U_j, U_i is trapped. Here the colluders may include the signer.

Lemma 2. *The success probability in 'transfer with a trap' is at most equal to* $\max\{P_D^{t_{V_1},t_{V_2}}, P_{RS}^{t_A,t_{V_1},t_{V_2}}\}$.

Proof. If the signer is part of the collusion the attack succeeds if *(i)* the message satisfies the requirement for a successful denial attack, and *(ii)* is furthermore unacceptable to some receiver U_j. If the signer is not part of the collusion the attack succeeds if *(i)* the message satisfies the requirement for a successful spoofing attack, and *(ii)* is not acceptable to both the receiver U_j and the arbiter U_A. Success in transfer with a trap requires two conditions to be satisfied and thus has less chance of success than plain denial or spoofing attacks, respectively.

SHZI02 introduced a wide range of new security notions closely following computational models. They considered the 'strongest security notion' for their proposed construction. In our model of asymmetric group *A*-codes, we consider the most powerful collusion, with the most useful information, using their best strategy, with success defined by success against a single verifier. The most powerful collusion includes the signer and the arbiter, with their key information and access to oracle queries, and the attack goal is constructing 'a message' acceptable to 'a verifier' (in SHZI02 notation, existential forgery and existential acceptance). This is the same as the 'strongest security notion' in SHZI02.

Other types of forgeries in SHZI02 are *Total break* and *selective forgery* which are harder to achieve and, while expressible in our framework, are of less interest. Similarly, colluders information can be restricted to key information only *(Key–only attacks);* i.e. disallow queries. As mentioned earlier, we consider all valid query sequences (§3.1), so adaptive queries need not be considered.

SHZI02 define other security goals (*Total* and *selective* acceptance), both harder to achieve than the *existential acceptance* considered in our model and used in the 'strongest security notion'.

SHZI02 [15] note *"the strongest signature scheme is one secure against existential acceptance forgery under adaptive chosen message attack and adaptive chosen signature attacks",* and use this model for their constructions. The security model of MRA^3–codes, matches this definition. In section 5 we give a language to express a wide range of security models in authentication scenarios. The value of particular scenarios depends on practical applications.

Information theoretic bounds

Establishing the relationship between USDS in HSZI00 and SHZI02 models and multireceiver codes allows us to derive information theoretic bounds for USDS. We give bounds for the attacks defined in section 3.1. Since the arbiter is treated as having a verifier's information, the bounds for arbiter inclusive attacks are the same as the bounds for a collusion of size $w+1$. These bounds consider A–queries only and so the query set is $Q(t_A)$, with $(Q, R)(t_A)$ the message and response set. We use $M' = M \backslash Q(t_A)$ to denote the rest of the message space and E_C for the keyspace of colluders.

$$P_D \geq 2^{-I(M;E_i,E_A|E_C)} \qquad P_{RS}^{t_A} \geq 2^{-I(M';E_i|E_C,(Q,R)(t_A))}$$
$$P_{RF}^{t_A} \geq 2^{-I(M';E_T|E_C,(Q,R)(t_A))}$$

The bounds when V–queries are considered remains an open problem.

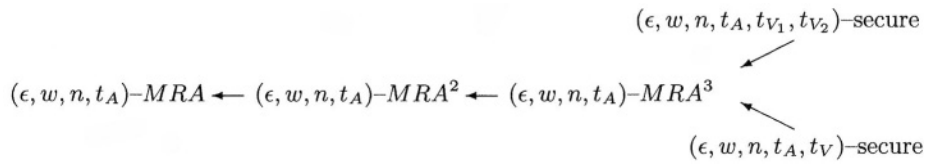


Fig. 1. The relationship between different types of security notions for authentication codes. We use $A \rightarrow B$ to imply that a code of type A satisfies the security requirements of a code of type B. All codes, except for (ϵ, w, n, t_A)–MRA, are types of USDS. The $(\epsilon, w, n, t_A, t_V)$ code satisfies the strongest security notions of SHZI02 with t_A A–queries, t_V V_1 queries and $t_V - 1$ V_2 queries. We note the two rightmost USDS are essentially the same and the distinction lies in seperating V_1 and V_2–queries.

4 Constructions

In constructing group A–codes the challenge is to have *secure and efficient* constructions. Optimal constructions meet minimum requirements for keys and have the shortest signature length, but are rare andinflexible. ϵ–security gives guaranteed security without the highest efficiency, but with the advantage of providing flexibility and a wide range of constructions.

Proof of security for systems with complex security goals is generally difficult. We give two algebraic methods of constructing group–based A–codes from simpler A–codes. The constructions use *polynomial codes* where signature generation and verification can be expressed by evaluation of multivariate polynomials over a finite field F_q with q elements. We assume all polynomials are in $F_q[x_1, \cdots x_n]$, the ring of polynomials over the finite field F_q. Constructions $\mathbf{C_0}$ and $\mathbf{C_1}$ are polynomial codes. Polynomial codes are generally efficient and often optimal.

A polynomial code can be expressed in terms of polynomials generated by the trusted authority (TA) during the **Key generation (KeyGen)** phase. The signer receives a signing polynomial $A(x, z)$ for generating signatures. Each receiver U_i gets a verification polynomial and some identification information u_i. The identifier may be public (private) if the sender is trusted (distrusted).
Signature generation (SigGen): The signature of a source s is $\alpha(z) = A(s, z)$. We assume authentication codes without secrecy so the signed message is $(s, \alpha(z))$.
Signature verification (SigVer): A receiver U_i accepts a signed message iff $\alpha(z)|_{z=u_i} = V_i(x)|_{x=s}$.

4.1 A Systematic Approach to Constructing Group A–codes

We use multiple instances of a component code, combined using powers of a single variable, or using distinct variables for each instance. We consider two synthesis algorithms, Σ_1 and Σ_2.

Synthesis Algorithm: Σ_1

KeyGen: The TA generates $k + 1$ instances of the component authentication code. For each instance j, a component signing key $A_j(x, z)$ and component verification keys, $V_{ij}(x)$ for each verifier i are generated, such that $V_{ij}(x) = A_j(x, u_i)$ where $u_i \in \mathbb{F}_q$ is U_i's identifier. The TA gives U_0 the polynomial

$$B(x, z, y) = \sum_{j=0}^{k} A_j(x, z) y^j$$

and each verifier U_i another identifier u_i', if necessary, and a polynomial

$$W_i(x) = \sum_{j=0}^{k} V_{ij}(x)(u_i')^j = B(x, u_i, u_i') \ .$$

SigGen: The signature of a source state s is $\alpha(z, y) = B(s, z, y)$.
SigVer: A receiver U_i accepts a signed message iff $\alpha(z, y)|_{z=u_i, y=u_i'} = W_i(x)|_{x=s}$.

Discussion and Example for Σ_1

Σ_1 can be used to construct codes that provide protection for multiple receivers, construct asymmetric codes from symmetric codes, and construct dynamic sender codes from single sender codes.

We shall consider synthesis of an *MRA*–code from a two party *A*–code. The approach also be used to construct HSZI00 (dynamic sender) from a (ϵ, w, n, t_A)–secure code providing protection against collusions of size w and t *A*–queries.

Let the component code be $\mathbf{C_0}$, where the signer has $A(x) = a + bx$ and $V(x) = A(x)$. Using Σ_1 we obtain an authentication code as follows.
KeyGen: The TA generates $k+1$ instances of the code $\mathbf{C_0}$, specified by $A_j(x) = a_j + xb_j, 0 \leq j \leq k$. The TA gives U_0 the polynomial

$$B(x, y) = \sum_{j=0}^{k} (a_j + b_j x) y^j$$

and each verifier U_i an identifier u_i and verification polynomial

$$W_i(x) = \sum_{j=0}^{k} (a_i + b_i x)(u_i)^j \ .$$

SigGen: The signature for a source state s is $\alpha(y) = B(s, y)$.
SigVer: User U_i accepts $(s, \alpha(y))$ iff $\alpha|_{y=u_i} = W_i(x)|_{x=s}$.

The above construction is the same as the $(\epsilon = 1/q, k, n)$–secure *MRA*–code of [4]. This follows since the signature generation function can be written as $B_i(x, y) = \sum_j a_j y^j + x \sum_i b_j y^j = f(y) + xg(y)$. If u_i is only known to the receiver, we have an (ϵ, k, n)–MRA^2–code, with $\epsilon = 1/(q - k)$, since the signer cannot deny a signature.

Synthesis Algorithm: Σ_2

KeyGen: The TA generates $k + 1$ instances of the component authentication code. For instance j, a signing key $A_j(x, z)$ and verification keys, $V_{ij}(x) = A_j(x, u_i)$, for each verifier i, are generated. The TA gives U_0 the polynomial

$$B(x, z, \mathbf{Y}) = \sum_{j=0}^{k} A_j(x, z) \mathbf{Y}_j$$

and each verifier U_i an identifier u_i, randomly generated vector $\mathbf{v_i} \in \mathbb{F}_q^{k+1}$ also written as $\mathbf{v_i} = (v_{i0}, v_{i1}, \ldots, v_{ik})$, and a verification polynomial

$$W_i(x) = \sum_{j=0}^{k} V_{ij}(x) v_{ij} \ .$$

SigGen: The signature of a source state s is $\alpha(z, \mathbf{Y}) = B(s, z, \mathbf{Y})$
SigVer: A receiver U_i accepts a signed message iff $\alpha|_{(z=u_i, \mathbf{Y}=\mathbf{v_i})} = W_i(x)|_{x=s}$.

Discussion and Example for Σ_2

This algorithm allows one to construct asymmetric codes from symmetric ones, multireceiver codes from single receiver codes, or dynamic codes from single sender codes. Again we consider constructing an *MRA*–code from a two party *A*–code. As before we use $\mathbf{C_0}$ as the component code.
KeyGen: The TA randomly generates $k+1$ instances of the code $\mathbf{C_0}$, specified by the polynomial $A_j(x) = a_j + xb_j, 0 \leq j \leq k$. The TA gives U_0 the polynomial

$$B(x, \mathbf{Y}) = \sum_{j=0}^{k} A_j(x)\mathbf{Y}_j$$

and each U_i an identifier $u_i \in \mathbb{F}_q$, a randomly generated vector $\mathbf{v_i} \in \mathbb{F}_q^{k+1}$, and a polynomial

$$W_i(x) = \sum_{j=0}^{k} V_{ij}(x)v_{ij} \ .$$

SigGen: The signature for a source state s is $\alpha(\mathbf{Y}) = B(s, \mathbf{Y})$.
SigVer: User U_i accepts $(s, \alpha(\mathbf{Y}))$ iff $\alpha(\mathbf{Y})|_{\mathbf{Y}=\mathbf{v_i}} = W_i(s)$.

Theorem 1. *The above construction is an (ϵ, w, n)–MRA–code. The authenticator and key sizes for signer and user are $k+1, 2(k+1)$ and $k+3$ respectively. In this case $\epsilon = 1/q$.*

Intuitively this result follows since each copy of the two party code provides security for a single colluder and for each colluder one copy of the code is added. Compared to $\mathbf{C_1}$, obtained using Σ_1, this construction has a larger key size for verifiers but the same signer key size and the same signature length.

Σ_2 construction can also be used to provide protection against *V*–queries. This property will be used in synthesising SHZI02 (§4.3). To show this property we re–visit the construction above and show it can be seen as an $(\epsilon, 0, n, 1, t_{V_1} = k+1, t_{V_2} = k)$–secure code. That is, a code where signer is distrusted but verifiers are trusted. This is dual to traditional *MRA*–codes where the signer is trusted and verifiers collude. The most powerful attack is the signer's denial attack against a verifier. The signer does not know the identity vector $\mathbf{v_i}$ and has to construct a pair $(s, \alpha'(\mathbf{Y}))$ such that (i) $\alpha'(\mathbf{v_j}) = W_j(s')$ and (ii) $\alpha(\mathbf{v_j}) \neq B(s, \mathbf{Y})$. He can have k V_2–queries. The V_1 queries give information about the key information of other verifiers only. The signer attempts to construct a message $(s, \alpha'(\mathbf{Y}))$ such that (i) $\alpha'(\mathbf{v_j}) = W_j(s')$ and (ii) $\alpha(\mathbf{v_j}) \neq B(s, \mathbf{Y})$.

Each V_2–query gives a tag $\alpha_i\mathbf{Y}, 0 \leq i \leq k-1$ such that $\alpha_i(\mathbf{v_j}) \neq W_j(s')$, i.e. a source state, tag pair unacceptable to U_j. The adversary can choose k α_i so $\alpha_i(\mathbf{v_j}) = \alpha_l(\mathbf{v_j})$ if and only if $i = l$, so each tag tests a different value against

$W_j(s')$. Each of the tags used reduces the possible values of $W_j(s')$ by 1. Thus the probability of the adversary choosing a tag acceptable to U_j is $\epsilon = 1/(q-k)$.

This shows one may apply Σ_2 to $\mathbf{C_0}$ to obtain either a $(\epsilon, k, n, 1, 0, 0)$–secure or a $(\epsilon, 0, n, 1, k+1, k)$–secure code. Indeed, though we shall not give details here, the Σ_2 synthesis gives an $(\epsilon, k_1, n, 1, k_2+1, k_2)$–secure code, where $k_1 + k_2 = k$.

4.2 Construction of USDS

Σ_2 can be applied to the A^2–code and A^3–codes in [9] to construct MRA^2 and MRA^3–codes from $\mathbf{C_1}$. We omit the details and instead show how to use a synthesis approach similar to Σ_1 on source states rather than on identities to synthesise MRA^2 and MRA^3–codes that protect against higher number of queries. That is we show how to construct a $(\epsilon, w, n, t_A, 0, 0)$–secure code from a $(\epsilon, w, n, 1, 0, 0)$–secure code. A similar argument applies to MRA^3–codes when the arbiter has the key information of a verifier.

Theorem 2. *The construction $\mathbf{C_1}$ is an (ϵ, w, n)–secure MRA^2–code if u_i are known only to U_i. We have $\epsilon = w/(q-w)$.*

We call this construction $\mathbf{C_1^2}$. The security proof uses the knowledge that the strongest collusion consists of the signer and w verifiers whose aim is to construct an authenticator $\alpha(x)$ (a polynomial of degree w) such that $\alpha(u_j) = f(u_j) + sg(u_j)$ for some j. The result follows since while colluders know $f(x)$ and $g(x)$ they cannot determine the identity u_j of U_j. The construction guarantees ϵ–security if for given security ϵ and w we have $q \geq w(1 + 1/\epsilon)$. To construct an (ϵ, w, n, t_A)–secure MRA^2–code we use $t_A + 1$ copies of $\mathbf{C_1^2}$ and apply a modified version of Σ_1. (Similarly for MRA^3 from $\mathbf{C_1^3}$.)

KeyGen: The TA generates $t+1$ independent $\mathbf{C_1^3}$, $f_i(x) + zg_i(x)$, and gives U_0

$$B(x, y, z) = \sum_{k=0}^{t} (f_k(x) + zg_k(x))y^k = \sum_{k=0}^{t} \sum_{i=0}^{w} \sum_{j=0}^{1} a_{kij} x^i z^j y^k \ .$$

The TA gives verifier U_i a private $u_i \in F_q$ and $B(u_i, y, z)$. The arbiter has the key information of a verifier, that is $B(u_a, y, z)$ where u_a is the arbiters identifier.

SigGen: The signature of a source state $s \in \mathbb{F}_q$ is $\alpha(x, z) = B(x, s, z)$.

SigVer: User U_i accepts the message as authentic iff $\alpha|_{x=u_i} = B(u_i, y, z)|_{y=s}, \forall z$.

The key sizes for the signer and each verifier are $2(t+1)(w+1)$ and $2t+3$, respectively. The tag length is $2(w+1)$. As before appropriate choices of parameters can provide ϵ–security for any chosen ϵ.

Theorem 3. *The above construction is a MRA^3–codes that protects against t A–queries with $\epsilon = w/(q-w)$.*

This code is similar to a generalised $\mathbf{C_1}$ construction given in [13, §5.1] as an *MRA*–code protecting against multiple *A*–queries.

4.3 USDS Constructions: The SHZI02 Model

SHZI02 gave a construction that satisfies their proposed 'strongest security notion'. We construct a code with the same security level using the synthesis methodology above. The main advantage of this description is that the security proof can be straightforwardly derived from that of the underlying codes.

The SHZI02 model uses the same setting as MRA^3–codes. For an attack against U_j, by a collusion of w out of n verifiers, the adversary may have (i) t A–queries, (ii) t' V_1 queries from each verifier other than U_j, and (iii) $t'-1$ V_2–queries rejected by U_j.

The synthesis has two steps: *(i)* constructing an $(\epsilon,0,2,t,0,0)$–secure code, and *(ii)* constructing a code with $(\epsilon,w,n,t,t',t'-1)$–security.

We start from $\mathbf{C_0}$: a component code that is $(\epsilon,0,2,1,0,0)$–secure. The key is a pair of random numbers $(a,b)\in\mathbb{F}_q^2$ shared by the signer and verifier. Using the synthesis akin to Σ_1, described in the previous section, we take $t+1$ copies, thus $A_i(x)=a_i+b_ix$, we obtain an $(\epsilon,0,2,t,0,0)$–secure code, where the polynomial held by the signer and by each verifier (noting they are still all trusted), is

$$B(x,y)=\sum_{i=0}^{t}A_i(x)y^i \;=f(y)+xg(y)$$

where $f(y)=\sum_{i=0}^{t}a_iy^i$ and $g(y)=\sum_{i=0}^{t}b_iy^i$.

The signature for a source state s is $\alpha(x)=B(x,s)$, and a message is accepted if $\alpha(x)=B(x,s)$. Let this $(\epsilon,0,2,t,0,0)$–secure code be the component code, and apply Σ_2 to $t'+w+1$ copies $B_i(x,y),0\le i\le t'+w$. The TA gives U_0

$$C(x,y,\mathbf{Y})=\sum_{j=0}^{w+t'}B_j(x,y)\mathbf{Y}_j$$

and verifier U_i a randomly chosen identity $\mathbf{v_i}\in\mathbb{F}_q^{w+t'+1}$ and verifying polynomial

$$W_i(x,y)=C(x,y,\mathbf{v_i})=\sum_{j=0}^{w+t'}B_j(x,y)\mathbf{v_{ij}}\ .$$

SigGen: The signature for a source state s is $\alpha(x,\mathbf{Y})=B(x,s,\mathbf{Y})$.
SigVer: User U_i accepts $(s,\alpha(x,\mathbf{Y}))$ iff $\alpha(x,\mathbf{Y})|_{\mathbf{Y}=\mathbf{v_i}}=W_i(x,y)|_{y=s},\forall x$.

We may write the complete key of the signer as

$$C(x,y,\mathbf{Y})=\sum_{i=0}^{t}\sum_{j=0}^{w+t'}\sum_{k=0}^{1}A_{ijk}\mathbf{Y}_jy^ix^k\ .$$

This *is* the construction of SHZI02, satisfying the 'strong security notion' and constructed using Σ_1 and Σ_2. We used Σ_1 to synthesise a t–message system from a 1–message code. We used Σ_2 to synthesise an asymmetric system secure

against collusions of up to size w, and t' V–queries. Collusions may include the signer, or arbiter in our model, and the arbiter has a verifier's key.

SHZI02 note this code meets the $1/q$ bound on security, although it is not known to be optimal. Rather than starting with $\mathbf{C_0}$ we could omit the Σ_1 step and use an optimal $(\epsilon, 0, 2, t, 0, 0)$–secure code, with signer polynomial $B(x) = \sum_{i=0}^{t} A_i y^i$ [9]. We omit details but synthesising this code using Σ_2 as above gives a $(1/(q-t'), w, n, t, t', t'-1)$–secure code. The authenticator, signer's and verifier's keys sizes are, $(w+t'+1)$, $(t+1)(w+t'+1)$ and $(w+t'+1)+(t+1)$, respectively, half those of the SHZI02 as formulated above. While information theoretic and combinatorial bounds are not yet known for these codes, it seems unlikely the construction of SHZI02, as developed above, is optimal.

5 Generalised Authentication Codes

A general setting for Generalised A–codes (GA–codes) consists of a set $\mathcal{U}$ of participants, each with some secret key information, such that any group member may sign a message and verify signed messages. To emphasise the new aspects of GA–codes, we assume there is one signer, the approach can be extended to dynamic signer systems. The set $\mathcal{U}$ contains n verifiers, an arbiter U_A, and the signer U_0. Let E_X denote the set of all possible keys values held by a set X of participants. We use $\mathcal{M}(E)$ to denote the set of messages valid under all the keys in E. An adversary corrupts some subset of participants that will form a *colluding set.* We assume these sets are staticly determined.

We consider codes without secrecy, where the authenticated message for a source state s can be written in the form (s, t), where t is a tag or authenticator.

Generalised oracles: We generalise the oracles of section 3.1 by defining *generalised A–oracles* and *generalised V–oracles* that can generate and verify, respectively, messages of defined *type.*

Message type: We say a message m is of type $\tau = (e_{i_1}, \ldots, e_{i_l}; e_{j_1}, \ldots, e_{j_{l'}})$, if

$$m \in \{\widehat{\mathcal{M}}(e_{i_1}) \cap \ldots \cap \widehat{\mathcal{M}}(e_{i_l})\} \backslash \{\widehat{\mathcal{M}}(e_{j_1}) \cap \ldots \cap \widehat{\mathcal{M}}(e_{j_{l'}})\} \tag{1}$$

where $\widehat{\mathcal{M}}(e) \subseteq \mathcal{M}(e)$. In other words m is valid for $(e_{i_1}, \ldots, e_{i_l})$ and not valid for $(e_{j_1}, \ldots, e_{j_{l'}})$. This captures exclusions of already 'used' queries from the message space. A message type is NULL if the types message space is empty.

A gA–oracle takes a source state and type τ, and generates an authenticated message (source state followed by the signature) of type τ, or outputs NULL, if it is not possible to generate such a message. A gV–oracle takes a message m and a type τ and produces a TRUE result if m is of type τ, and FALSE otherwise.

Since the status of a message with respect to the arbiter is also relevant, one may have messages known to be acceptable or unacceptable to the arbiter by considering inclusion in $\mathcal{M}(e_A)$. If the arbitration algorithm differs from the verification algorithm, arbiter queries need to be considered separately.

Collusion structure: The collusion structure is written as a pair (C, Φ_C), where C is a colluding set and Φ_C determines the oracle queries accessible to C. The set Φ_C contains a list of message types ϕ_i, multiplicities ℓ_i and a flag ρ_i that determines if the query is made to the gA–oracle or to the gV–oracle. For each i, ℓ_i messages of type ϕ_i may be queried to an oracle of type ρ_i. Let $\mathcal{R}(\phi_i)$ be the set of input and response pairs associated with the ϕ_i queries.

A $(\epsilon, w, n, t_A, t_V)$ *–threshold* collusion structure is a collusion structure in which a colluding set contains at most w verifiers and has access to up to t_A A–queries, up to $t_V - 1$ V_2–queries (from the targeted verifier) and up to t_V V_1–queries (from each other verifier). A collusion set may also include the signer, and/or the arbiter.

The **Goal of an attack** is specified by the type of message to be constructed by the colluders.

An **Authentication Scenario** $\sigma(\mathcal{C})$ is defined by a set of participants, a collusion structure, and the protection the system can provide against colluder's attacks. Performance of an authentication scenario against a colluding set C with goal type γ is measured by $P(\gamma|\Phi_C)$, the highest success chance of a collusion with message set Φ_C. The success probability of such an attack is defined as

$$P(\gamma; \Phi_C) = \max_{\mathcal{C}} \max_{e_C \in E_C} \max_{\phi_C \in \Phi_C} P(m \text{ is of type } \gamma | \mathcal{R}(\phi_C), e_C)$$

where $P(m(\gamma)|\mathcal{R}(\phi_C), e_C)$ is the probability of generating the message m of type γ given queries with responses, $\mathcal{R}(\phi_C) \in \mathcal{R}(\Phi_C)$, specified by the collusion structure ϕ_C, and key information $e_C \in E_C$. We note that for gA–queries only the space $\mathcal{R}$ reduces to the message space $\mathcal{M}(\Phi_C)$, as below.

Information Theoretic Bounds

The attack probability bounds for A, A^2, A^3, MRA, MRA^2, tA^3 and $tMRA^2$ codes, at least may be concisely represented using authentication scenarios;

$$P(\gamma; \Phi_C) \geq 2^{-I(M(E_\gamma); E_\gamma | M(\Phi_C), E_C)} .$$

6 Concluding Remarks

We proposed an extension of traditional A–codes and showed the resulting framework encompesses the recently proposed USDS schemes, and all the previously known ones, hence unifying all models and constructions in the area. Introducing the notion of V–queries suggests an interesting model for attacker's strategy in A–codes not previously considered. This is hence a rich area for research.

We also developed an algebraic method for synthesizing group A–codes from simpler component codes, which removes the shortcoming of previous synthesis constructions. We gave two general methods, called Σ_1 and Σ_2, and gave an example construction using each.

We believe our work fills a gap in understanding USDS and provides a unified framework for USDS and their future extensions.

Acknowledgements

This work is supported in part by an Australian Research Council Discovery Grant (ARC Ref No. A10012057).

References

[1] E. F. Brickell and D. R. Stinson 'Authentication codes with multiple arbiters.' *Eurocrypt'88* LNCS **330,** (Springer–Verlag, 1988) 51–5.
[2] D. Chaum and S. Roijakkers 'Unconditionally–secure digital signatures.' *Crypto'90* LNCS **537** (Springer, 1990) 206–15.
[3] Y. Desmedt and M. Yung 'Arbitrated unconditionally secure authentication can be unconditionally protected against arbiter's attack.' *Crypto'90* LNCS **537** (Springer, 1990) 177–88.
[4] Y. Desmedt, Y. Frankel and M. Yung 'Multi–receiver/multi–sender network security: efficient authenticated multicast/feedback.' *IEEE Infocom'92* (1992) 2045–54.
[5] H. Fujii, W. Kachen and K. Kurosawa 'Combinatorial bounds and design of broadcast authentication.' *IEICE Trans.* **E79–A**(4) (1996) 502–6.
[6] E. N. Gilbert, F. J. MacWilliams and N. J. A. Sloane 'Codes which detect deception.' *Bell System Tech. J.* **53**(3) (1974) 405–24.
[7] G. Hanaoka, J. Shikata, Y. Zheng and H. Imai 'Unconditionally secure digital signature schemes admitting transferability.' *Asiacrypt'00* LNCS **1976,** (2000) 130–42.
[8] G. Hanaoka, J. Shikata, Y. Zheng and H. Imai 'Efficient and unconditionally secure digital signatures and a security analysis of a multireceiver authentication code.' *PKC'02* LNCS **2274,** (2002) 64–79.
[9] T. Johansson 'Contributions to unconditionally secure authentication.' Ph.D. Thesis, (Lund University, Sweden, 1994).
[10] T. Johansson 'Lower bounds on the probability of deception in authentication with arbitration.' *IEEE Trans. Inform. Theory.* **40**(5) (1994) 1573–85.
[11] T. Johansson 'Further results on asymmetric authentication schemes' *Information and Computation* **151** (1999) 100–33.
[12] J. Rompel 'One–way functions are necessary and sufficient for secure signatures.' *STOC'90* (1990) 387–94.
[13] R. Safavi–Naini and H. Wang 'Multireceiver authentication codes: Models, bounds, constructions and extensions.' *Information and Computation* **151** (1999) 148–72.
[14] R. Safavi–Naini and H. Wang 'Broadcast authentication for group communication.' *Theoretical Computer Science* **269** (2001) 1–21.
[15] J. Shikata, G. Hanaoka, Y. Zheng and H. Imai 'Security Notions for Unconditionally Secure Signature Schemes' *Eurocrypt'02* LNCS **2332** (2002) 434–49.
[16] G. J. Simmons 'Authentication theory/coding theory' *Crypto'84* LNCS **196** (Springer–Verlag, 1984) 411–31.
[17] G. J. Simmons 'A Cartesian product construction for unconditionally secure authentication codes that permit arbitration' *J. Crypt.* **2**(2) (1990) 77–104.
[18] B. Smeets 'Bounds on the probability of deception in multiple authentication.' *IEEE Trans. Inform. Theory* **40**(5) (1994) 1586–91.
[19] Y. Wang and R. Safavi–Naini 'A^3–codes under collusion attacks' *Asiacrypt'99,* LNCS **1716** (Springer, 1999) 360–98.

From Digital Signature to ID-based Identification/Signature

Kaoru Kurosawa[1] and Swee-Huay Heng[2]

[1] Department of Computer and Information Sciences,
Ibaraki University,
4-12-1 Nakanarusawa, Hitachi, Ibaraki 316-8511, Japan
kurosawa@cis.ibaraki.ac.jp
[2] Department of Communications and Integrated Systems,
Tokyo Institute of Technology,
2-12-1 O-okayama, Meguro-ku, Tokyo 152-8552, Japan
shheng@crypt.ss.titech.ac.jp

Abstract. In this paper, we first formalize the concept of ID-based identification scheme. Secondly, we show a transformation from any digital signature scheme satisfying certain condition to an ID-based identification scheme. As an instance, we present the first provably secure ID-based identification scheme based on the hardness of discrete logarithm problem. (More precisely, the hardness of gap Diffie-Hellman (GDH) problem.) We further show that for the ID-based signature scheme which is obtained by the Fiat-Shamir heuristic, a tight security bound is easily derived due to our transformation.

Key words: ID-based cryptography, signature scheme, identification scheme, GDH group

1 Introduction

1.1 On ID-based

In the last few years, research on identity (ID)-based *encryption* schemes [4,8,5] and *signature* schemes [18,16,13,7] have been very active. In an ID-based scheme, the identity of each user is used as his public key string. Most of the schemes employed bilinear pairings in their constructions, motivated by the novel work of Boneh and Franklin [4].

On the other hand, an *identification* scheme enables prover holding a secret key to identify himself to a verifier holding the corresponding public key. Fiat and Shamir mentioned in the fundamental paper of identification scheme [11] that their scheme is ID-based. Since then, there have been a large number of practical identification protocols in the literature, to name a few [11,10,12,17,15]. However, to the best of our knowledge, there is no rigorous definition as well as security proof for "ID-based" identification schemes in the open literature.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 248–261, 2004.

1.2 On Equivalences, Relationships, and Dualities

Many current research focus on drawing equivalences, relationships and dualities between different primitives, and these discoveries lead to new understanding and novel constructions of the related primitives.

For example, we have the paper on "From identification to signatures via Fiat-Shamir transform" by Abdalla et al. [1], where the idea was initially presented by Fiat and Shamir in 1986 [11]. In [14], Kiayias and Yung introduced new design methodologies for group signatures that convert a traitor tracing scheme into a group signature scheme.

1.3 Our Contribution

In this paper, we first formalize the concept of ID-based identification scheme. The main differences of ID-based identification schemes from the usual identification schemes are that: (1) The adversary can choose a target identity ID of her choice to impersonate as opposed to a random public key; (2) The adversary can possess private keys of some users which she has chosen.

Note that Schnorr's identification scheme [17] is not ID-based because each user must publicize his public key. (In other words, he cannot use his identity as his public key string.) In Guillou and Quisquater (GQ) identification scheme [12], each user can use his identity as his public key string. However, we cannot prove the security as mentioned above. Hence it is not ID-based, either.

Secondly, we show a transformation from a digital signature scheme (DS) to an ID-based identification scheme, where we require that the signature scheme has a three-move honest verifier zero-knowledge proof on knowledge. We then prove that the resulting ID-based identification scheme is secure against impersonation under passive attacks if the underlying signature scheme is secure against existentially forgery on adaptive chosen message attacks. An ID-based identification scheme can be further transformed to an ID-based signature scheme, following the Fiat-Shamir transform paradigm [11,1]. That is,

$$\text{DS scheme} \rightarrow \text{ID-based identification scheme} \rightarrow \text{ID-based DS scheme.}$$

Tight security bounds are directly obtained by our transformation both for the ID-based identification scheme and the ID-based signature scheme if tight security proof is known for the underlying signature scheme.

As an instance, we present the first provably secure ID-based identification scheme based on the hardness of discrete logarithm problem. More precisely, it is based on the hardness of GDH problem. Our scheme uses Boneh et al.'s short signature scheme as a building block where the security is based on the GDH groups [6]. Similarly to Schnorr's (non ID-based) identification scheme, our scheme allows precomputation, reducing the real time computation of the prover to just one multiplication. It is thus particularly suitable for provers with limited computational ability.

We can further obtain an ID-based signature scheme from the ID-based identification scheme. The resulting signature scheme coincides with Cha and

Cheon's scheme [7]. However, we provide a tighter security bound due to our transformation: GDH signature scheme → ID-based identification scheme → ID-based signature scheme. This in turn improves the efficiency of the scheme since smaller modulus can be used for the same level of security.

We also prove that the proposed ID-based identification scheme is secure against active attacks under the one-more DH assumption, where the one-more DH assumption is a natural analogue of the one-more RSA inversion assumption introduced in [2].

Finally, we point out that we can easily obtain GQ type ID-based identification/signature schemes by combining the Full Domain Hash RSA (FDH-RSA) signature scheme with our transformation. By using the result of Coron [9], tight security bound is further obtained.

1.4 Organization

The rest of the paper is organized as follows. Section 2.1 recalls the formal definition of digital signature schemes. We give the definition of GDH groups, following with the GDH signature scheme proposed by Boneh et al. [6] in Section 2.2. We present the formal model and the security definition of ID-based identification schemes in Section 3. Next, we show how to transform a digital signature scheme to an ID-based identification scheme in Section 4. A security analysis of the transformation follows in Section 4.3. Subsequently, in Section 5 we present our proposed ID-based identification scheme and show that it is secure against impersonation under passive attacks. In Section 6 we present a tight security reduction of the ID-based signature scheme based on the GDH groups. In Section 7 we prove that the proposed ID-based identification scheme is also secure against active attacks under the one-more DH assumption. In Section 8 we briefly discuss the applicability of our proposed transformation method to GQ schemes. We conclude the paper in Section 9.

2 Digital Signature Scheme

2.1 Definition

The standard definition of digital signature schemes is decribed as follows.

Definition 1. *A digital signature scheme $\mathcal{DS}$ is denoted by a triple* (Gen, Sign, Verify) *of polynomial-time algorithms, called key generation algorithm, signing algorithm and verification algorithm, respectively. The first two algorithms are probabilistic.*

- **Key Generation.** *On input 1^k (throughout this paper, k denotes the security parameter), the algorithm produces a pair of matching public and secret keys (pk, sk).*
- **Signing.** *On input (sk, m), the algorithm returns a signature $\sigma = \mathsf{Sign}_{sk}(m)$, where m is a message.*

– **Verification.** *On input* (pk, m, σ)*, the algorithm returns* 1 *(accept) or* 0 *(reject). We require that* $\mathsf{Verify}_{pk}(m, \sigma) = 1$ *for all* $\sigma \leftarrow \mathsf{Sign}_{sk}(m)$.

Security. We consider signature schemes that are secure against existential forgery under adaptive chosen message attacks. A forger F takes as input a public key pk, where $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$, and tries to forge signatures with respect to pk. The forger is allowed to query messages adaptively to the signing oracle to obtain the corresponding signatures. A valid forgery is a message-signature pair (m, σ) such that $\mathsf{Verify}_{pk}(m, \sigma) = 1$ but m has never been queried by F.

Definition 2. *We say that a digital signature scheme* $\mathcal{DS}$ *is* (t, q_S, ϵ)*-secure against existential forgery on adaptive chosen message attacks if for any forger* F *who runs in time* t*,*

$$\Pr(F \text{ can output a valid forgery}) < \epsilon,$$

where F *can make at most* q_S *signing queries.*

In the random oracle model, we consider a hash function H as a random oracle. Definition 2 is naturally generalized to the random oracle model: We say that a digital signature scheme $\mathcal{DS}$ is (t, q_S, q_H, ϵ)-secure if the condition of Definition 2 is satisfied, where F can make at most q_H random oracle queries.

2.2 GDH Signature Scheme

Boneh et al. proposed a signature scheme based on the GDH groups [6]. Let G be a (additive) cyclic group G generated by P with prime order q.

Computational Diffie-Hellman (CDH) Problem. Given (P, aP, bP) for some $a, b \in Z_q^*$, compute abP.
Decisional Diffie-Hellman (DDH) Problem. Given (P, aP, bP, cP) for some $a, b, c \in Z_q^*$, decide whether $c \equiv ab \bmod q$. (We say that (P, aP, bP, cP) is a DH-tuple if $c \equiv ab \bmod q$.)

We say that G is a GDH group if the CDH problem is hard, but the DDH problem is easy.

Key Generation. On input 1^k, generate an additive group G with prime order q where q is k-bit long. Choose an arbitrary generator $P \in G$. Pick a random $s \in Z_q^*$ and set $Q = sP$. Choose a cryptographic hash function $H : \{0,1\}^* \rightarrow G$. The public key is (P, Q, H) and the secret key is s.
Signing. Given the secret key s, a message $m \in \{0,1\}^*$, compute the signature $\sigma = sH(m)$.
Verification. Given the public key (P, Q, H), a message m and a signature σ, compute $H(m)$ and verify that $(P, Q, H(m), \sigma)$ is a valid DH-tuple.

GDH groups are defined formally as follows [6].

Definition 3. G *is a* τ*-decision group for Diffie-Hellman if the DDH problem can be computed in time at most* τ*, where* $P + Q$ *is computed in one time unit.*

Definition 4. *The advantage of an algorithm A in solving the CDH problem in group G is*

$$\mathsf{AdvCDH}_A \stackrel{def}{=} \Pr[A(P, aP, bP) = abP : a, b \stackrel{R}{\leftarrow} Z_q^*]$$

where the probability is over the choice of a *and* b*, and the coin tosses of A. We say that an algorithm A* (t, ϵ)*-breaks CDH in G if A runs in time at most* t*, and* $\mathsf{AdvCDH}_A \geq \epsilon$*.*

Definition 5. *A prime order group G is a* (τ, t, ϵ)*-GDH group if it is a* τ*-decision group for Diffie-Hellman and no algorithm* (t, ϵ)*-breaks CDH on it.*

The security of the scheme is derived as follows.

Proposition 1. *[6, Theorem, page 517] If G is a* (τ, t', ϵ')*-GDH group of order* q*, then the above GDH signature scheme is* (t, q_S, q_H, ϵ)*-secure against existentially forgery on adaptive chosen-message attacks, where*

$$t \geq t' - 2c_A \log_2 q(q_H + q_S),$$
$$\epsilon \leq 2eq_S\epsilon'$$

and c_A *is a small constant. Here e is the base of the natural logarithm.*

3 ID-based Identification Scheme

In this section, we give a formal definition of ID-based identification schemes.

3.1 Model

An ID-based identification scheme $\mathcal{ID} = (\mathcal{S}, \mathcal{E}, \mathcal{P}, \mathcal{V})$ is specified by four probabilistic polynomial-time (PPT) algorithms, called setup algorithm, extract algorithm, proving algorithm and verification algorithm, respectively. $\mathcal{P}$ and $\mathcal{V}$ are interactive algorithms that implement the prover and verifier, respectively. Alternatively we call $(\mathcal{P}, \mathcal{V})$ an identification protocol.

- **Setup.** A probabilistic algorithm used by the private key generator (PKG) to set up all the parameters of the scheme. $\mathcal{S}$ takes as input 1^k and generates the global system parameters params and the master-key. The system parameters will be publicly known while the master-key will be known to the PKG only.
- **Extract.** A probabilistic algorithm used by the PKG to extract a private key corresponding to a given public identity. $\mathcal{E}$ eceives as input the master-key and a public identity ID, it returns the corresponding private key d.
- **Identification Protocol.** $\mathcal{P}$ receives as input (params, ID, d) and $\mathcal{V}$ receives as input (params, ID), where d is the private key corresponding to the public identity ID. After an interactive execution of $(\mathcal{P}, \mathcal{V})$, $\mathcal{V}$ outputs a boolean decision 1 (accept) or 0 (reject). A legitimate $\mathcal{P}$ should always be accepted.

Specifically, we consider the following ID-based identification scheme having three-move protocol which is commonly called *canonical.*

1. $\mathcal{P}$ sends a *commitment* CMT to $\mathcal{V}$.
2. $\mathcal{V}$ returns a *challenge* CH which is randomly chosen from some set.
3. $\mathcal{P}$ provides a *response* RSP.
4. On input (params,ID,CMT,CH,RSP), $\mathcal{V}$ accepts or rejects.

3.2 Security

The security of ID-based identification schemes is almost the same as the security of standard identification schemes. However, it must be strengthened a bit as follows: (1) The adversary can choose a public identity ID of her choice to impersonate as opposed to a random public key; (2) When an adversary attacks a public identity ID, she might already possess the private keys of some users $\mathsf{ID}_1, \mathsf{ID}_2, \ldots$ of her choice. The system should remain secure under such an attack. Hence, the definition must allow the adversary to obtain the private key associated with any identity ID_i of her choice (other than the public identity ID being attacked).

The adversary goal is impersonation: an adversary succeeds if it interacts with the verifier in the role of a prover with public identity ID and can convince the verifier to accept with non-negligible probability.

There are two type of attacks on the honest, private key equipped prover, namely passive attacks and active attacks. These attacks should take place and complete before the impersonation attempt. In the passive attacks, the adversary does not interact with the prover. What the adversary does is eavesdropping and she is in possession of transcripts of conversations between the prover and the verifier. In the active attacks, the adversary gets to play the role of a cheating verifier, interacting with the prover several times, in an effort to extract some useful information before the impersonation attempt.

We describe the two-phase game between a passive (active) impersonator I and the challenger C. In Phase 1, the impersonator is allowed to make some extract queries. In addition, it can also make either some transcript queries (for passive attacks) or request to act as a cheating verifier (for active attacks). In Phase 2, I starts its impersonation attempt, plays the role as a cheating prover of a public identity ID of its choice, trying to convince the verifier.

- **Setup.** The challenger takes as input 1^k and runs the setup algorithm $\mathcal{S}$. It gives I the resulting system parameters params and keeps the master-key to itself.
- **Phase 1.**
  1. I issues some extract queries $\mathsf{ID}_1, \mathsf{ID}_2, \ldots$. The challenger responds by running the extract algorithm $\mathcal{E}$ to generate the private key d_i corresponding to the public identity ID_i. It returns d_i to I. These queries may be asked adaptively.
  2. I issues some transcript queries (for passive attacks) on ID_i or requests to act as a cheating verifier corresponding to some ID_i (for active attacks).
  3. The queries on step 1 and step 2 above can be interleaved.

– **Phase 2.** I outputs a challenge identity ID on which it wishes to impersonate whereby I can act as a cheating prover now, trying to convince the verifier.

Definition 6. *We say that an ID-based identification scheme $\mathcal{ID}$ is (t, q_I, ϵ)-secure under passive (active) attacks if for any passive (active) impersonator I who runs in time t,*

$$\Pr(I \text{ can impersonate}) < \epsilon,$$

where I can make at most q_I extract queries.

4 Transformation from $\mathcal{DS}$ to $\mathcal{ID}$

In this section, we show a transformation of a digital signature scheme $\mathcal{DS}$ to an ID-based identification scheme $\mathcal{ID}$. First, we state the requirement that a digital signature scheme $\mathcal{DS}$ must fulfill. Next, we present the transformation following by the security analysis.

4.1 Requirement for $\mathcal{DS}$

We require that a digital signature scheme $\mathcal{DS}$ has a canonical (three-move) zero-knowledge interactive proof system (ZKIP) on knowledge of signatures as follows.

Let pk be a public key, m be a message and σ be a signature on m. The common input to *(P, V)* is (pk, m). The secret input to P is σ. Let *view* =(CMT,CH,RSP) be a transcript of the conversation between *(P, V)*. Let *View* be the random variable induced by *view*. We say that (CMT,CH,RSP) is acceptable if V accepts it.

Definition 7. *We say that a digital signature scheme $\mathcal{DS}$ has a Δ-challenge zero-knowledge (ZK) protocol if there exists a canonical protocol (P, V) as follows. For any (pk, m),*

Completeness. *If P knows σ, then* $\Pr(V \text{ accepts}) = 1$.

Soundness. – *The number of possible challenge* CH *is equal to Δ.*
– σ is *computed efficiently from any two acceptable transcripts* $(\text{CMT}, \text{CH}_1, \text{RSP}_1)$ *and* $(\text{CMT}, \text{CH}_2, \text{RSP}_2)$ *such that* $\text{CH}_1 \neq \text{CH}_2$.

Zero-knowledgeness. *(P, V) is perfectly ZK for the honest verifier. That is, there exists a simulator S such that its output follows the same probability distribution as View.*

4.2 Transformation

Any digital signature scheme $\mathcal{DS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ satisfying the above requirement can be used as a building block to implement a canonical ID-based identification scheme $\mathcal{ID} = (\mathcal{S}, \mathcal{E}, \mathcal{P}, \mathcal{V})$.

Firstly, we point out the similarities between $\mathcal{DS}$ and $\mathcal{ID}$ and make a comparison between the algorithms associated with them. The setup algorithm $\mathcal{S}$ performs similar operations as the key generation algorithm Gen. Indeed, both of them take as input 1^k and generate:

- params or *pk,* respectively the system parameters or public key.
- master-key or *sk,* that will be used by the PKG in the extract algorithm or as a signing key by the user.

Thus, we can view that params = *pk* and master-key = *sk.*

The extract algorithm $\mathcal{E}$ is similar to the signing algorithm Sign. They take ID and m, respectively, as input and produce the corresponding private key d and signature σ, respectively. In other words, we can set that $\mathsf{ID} = m$ and $d = \sigma$.

Now in $\mathcal{ID}$, the prover $\mathcal{P}$ holds a secret key $d = \sigma$ corresponding to his public identity ID. Then $\mathcal{P}$ and $\mathcal{V}$ runs the Δ-challenge ZK protocol of $\mathcal{DS}$. We give the detail description as follows:

Setup. On input 1^k, $\mathcal{S}$ generates params = *pk* and master-key = *sk* using Gen.
Extract. For a given public identity $\mathsf{ID} = m$, $\mathcal{E}$ uses Sign to generate the corresponding private key $d = \sigma$, by using the master-key = *sk.*
Identification Protocol. The prover and verifier perform the Δ-challenge ZK protocol of $\mathcal{DS}$ and obtain the protocol as depicted in Fig. 1.

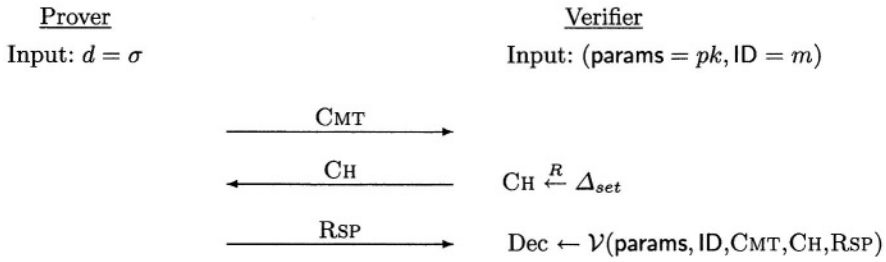


Fig. 1. A canonical ID-based identification protocol

4.3 Security Analysis

Theorem 1. *Let* $\mathcal{DS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *be a digital signature scheme which has a* Δ*-challenge ZK protocol. Let* $\mathcal{ID} = (\mathcal{S}, \mathcal{E}, \mathcal{P}, \mathcal{V})$ *be the associated canonical ID-based identification scheme as per the transformation shown above. Then* $\mathcal{ID}$ *is* (t, q_I, ϵ)*-secure against impersonation under passive attacks if* $\mathcal{DS}$ *is* (t', q_S, ϵ')*-secure against existential forgery on adaptive chosen message attacks, where*

$$t \geq (t'/2) - poly(k),\ q_I = q_S,\ \epsilon \leq \sqrt{\epsilon'} + (1/\Delta).$$

Proof. (Sketch) Let I be an impersonator who (t, q_I, ϵ)-breaks the ID-based identification scheme $\mathcal{ID}$. Then we will show that $\mathcal{DS}$ is not (t', q_S, ϵ')-secure. That is, we will present a forger F who (t', q_S, ϵ')-breaks the signature scheme $\mathcal{DS}$.

The forger F receives pk as its input. It then gives pk to the impersonator I. In Phase 1, the impersonator I starts the extract queries. If I issues an extract query ID_i, then the forger F queries ID_i to its signing oracle. F forwards the answer $d_i = \sigma_i$ of the signing oracle to I. These queries may be asked adaptively. I also issues some transcript queries on ID_i. Since $\mathcal{DS}$ has a Δ-challenge ZK protocol, there exists a simulator S whose output follows the same distribution as *View*. If I issues a request ID_i, F runs the simulator S on input (pk, ID_i). Suppose that S outputs $(\mathrm{CMT}_i, \mathrm{CH}_i, \mathrm{RSP}_i)$. Then F gives it to I.

Some time later, I decides that Phase 1 is over and it outputs a public identity ID on which it wishes to be challenged. I plays the role as the cheating prover, trying to convince the verifier $\mathcal{V}$ that she is the holder of public identity ID . F plays the role as $\mathcal{V}$. Immediately after the first run, F resets the prover I to after the step whereby I has sent the message CMT_1. F then runs the protocol again. Let the conversation transcripts for the first run and second run be (CMT,CH, RSP) and (CMT,CH', RSP'), respectively. Based on the Reset Lemma proposed by Bellare and Palacio in [3], we can extract the private key $d = \sigma$ from the two conversation transcripts with probability more than $(\epsilon - 1/\Delta)^2$.

Finally, when the impersonator I outputs σ, the forger F returns the message-signature pair (ID, σ) as its forgery. Thus it is clear that

$$t' \leq 2t + poly(k),\ q_S = q_I,\ \epsilon' \geq (\epsilon - \frac{1}{\Delta})^2.$$

□

5 Proposed ID-based Identification Scheme

In this section, we show the first provably secure ID-based identification scheme by applying our transformation to the GDH signature scheme.

5.1 q-Challenge ZK Protocol

We first show that the GDH signature scheme as described in Section 2.2 satisfies the requirement in Section 4.1.

Theorem 2. *The GDH signature scheme has a q-challenge ZK protocol.*

Proof. For the GDH signature scheme, we show a three-move canonical protocol *(P, V)* which satisfies the requirement in Section 4.1.

1. P chooses $r \in Z_q$ randomly and sends $x = rH(m)$ to V.
2. V chooses $c \in Z_q$ randomly and sends c to P.
3. P computes $y = (r + c)\sigma$ and sends y to V.
4. V accepts if and only if $(P, Q, x + cH(m), y)$ is a DH-tuple.

It is clear that the above protocol satisfies the completeness. The soundness is proved as follows. Suppose that (x, c_1, y_1) and (x, c_2, y_2) are two acceptable conversations. Then it holds that

$$x + c_1 H(m) = l_1 P,\ y_1 = l_1 Q$$

$$x + c_2 H(m) = l_2 P,\ y_2 = l_2 Q$$

for some l_1 and l_2. From the above equations, we obtain

$$(c_2 - c_1)H(m) = (l_2 - l_1)P \text{ and } y_2 - y_1 = (l_2 - l_1)Q.$$

This shows that $\sigma = (c_2 - c_1)^{-1}(y_2 - y_1)$ is a signature on m. (Recall that $Q = sP$ and $\sigma = sH(m)$.)

Finally, we show a simulator S. The purpose of S is to output $(\tilde{x}, \tilde{c}, \tilde{y})$ such that $(P, Q, \tilde{x} + \tilde{c}H(m), \tilde{y})$ is a DH-tuple. That is, $(P, Q, \tilde{x} + \tilde{c}H(m), \tilde{y}) = (P, Q, lP, lQ)$ for some l. Hence S chooses $l \in Z_q$ and $\tilde{c} \in Z_q$ randomly. S then outputs $(lP - \tilde{c}H(m), \tilde{c}, lQ)$. Thus we have shown that (P, V) is perfect ZK for the honest verifier. □

5.2 ID-based Identification Scheme Based on GDH

We can then obtain an ID-based identification scheme immediately from Section 4.2. Let $\mathcal{ID} = (\mathcal{S}, \mathcal{E}, \mathcal{P}, \mathcal{V})$ be four PPT algorithms as follows.

Setup. On input 1^k, generate an additive group G with prime order q. Choose an arbitrary generator $P \in G$. Pick a random $s \in Z_q$ and set $P_{pub} = sP$. Choose a hash function $H : \{0,1\}^* \to G$. Let the system parameters $\mathsf{params} = (P, P_{pub}, H)$ and the $\mathsf{master\text{-}key}$ is s which is known to the PKG only.

Extract. Given a public identity ID, compute the corresponding private key $d_{\mathsf{ID}} = sQ_{\mathsf{ID}}$ where $Q_{\mathsf{ID}} = H(\mathsf{ID})$.

Identification Protocol.

1. $\mathcal{P}$ chooses $r \in Z_q$ randomly, computes $U = rQ_{\mathsf{ID}}$ and sends U to $\mathcal{V}$.
2. $\mathcal{V}$ chooses $c \in Z_q$ randomly and sends c to $\mathcal{P}$.
3. $\mathcal{P}$ computes $V = (r + c)d_{\mathsf{ID}}$ and sends V to $\mathcal{V}$.
4. $\mathcal{V}$ verifies whether $(P, P_{pub}, U + cQ_{\mathsf{ID}}, V)$ is a DH-tuple.

Remark. Note that params is the public key of the GDH signature scheme and s is the secret key. d_{ID} is the signature on a message ID.

5.3 Security Against Passive Attacks

From Theorem 1 and Theorem 2, it is clear that the above ID-based identification scheme is secure against passive attacks if the GDH signature scheme is secure against existential forgery on adaptive chosen message attacks. The latter is indeed the case as shown in Proposition 1. Therefore, the above ID-based identification scheme is secure against passive attacks.

By combining these results quantitatively, we can obtain the concrete security. The security definition is generalized to the random oracle model as follows. We say that an ID-based identification scheme $\mathcal{ID}$ is (t, q_I, q_H, ϵ)-secure under passive (active) attacks if the condition of Definition 6 is satisfied, where the impersonator I can make at most q_H random oracle queries. (We can prove the random oracle version of Theorem 1 easily, where both F and I use the same random oracle H. If I makes a random oracle query, then F makes the same query to H and sends the obtained answer to I.)

Theorem 3. *If G is a (τ, t', ϵ')-GDH group, then the above ID-based identification scheme is (t, q_I, q_H, ϵ)-secure under passive attacks, where*

$$t \geq (t'/2) - c_A \log_2 q(q_H + q_I) - poly(k),$$
$$\epsilon \leq \sqrt{2eq_I\epsilon'} + (1/q),$$

and c_A is a small constant. Here e is the base of the natural logarithm.

6 ID-based Signature Scheme Based on GDH

We can further transform our proposed ID-based identification scheme to an ID-based signature scheme. This transformation is direct as in other Fiat-Shamir transformations except that it involves ID-based transformation.

The resulting signature scheme coincides with Cha and Cheon's scheme [7]. However, we can give a much tighter security reduction due to our transformation: GDH signature scheme → ID-based identification scheme → ID-based signature scheme. This in turn improves the efficiency of the scheme since smaller modulus can be used for the same level of security. (In [7], the security proof relies on the forking lemma. Hence the reduction is not tight and the proof is very complicated.)

6.1 Scheme

Setup. On input 1^k, generate an additive group G with prime order q. Choose an arbitrary generator $P \in G$. Pick $s \in Z_q^*$ randomly and set $P_{pub} = sP$. Choose two cryptographic hash functions: $H : \{0,1\}^* \to G$, $H_1 : \{0,1\}^* \times G \to Z_q$. Let the system parameters, $\mathsf{params} = (P, P_{pub}, H, H_1)$ and the $\mathsf{master\text{-}key}$ is s which is known to the PKG only.

Extract. Given a public identity ID, compute the corresponding private key $d_{\mathsf{ID}} = sQ_{\mathsf{ID}}$ where $Q_{\mathsf{ID}} = H(\mathsf{ID})$.

Signing. Given the private key d_{ID} and a message m, pick a random number $r \in Z_q$. Return the signature $\sigma = (U, V)$ where $U = rQ_{\mathsf{ID}}, c = H_1(m, U)$ and $V = (r + c)d_{\mathsf{ID}}$.

Verification. Given the system parameters $\mathsf{params} = (P, P_{pub}, H, H_1)$, a message m and a signature $\sigma = (U, V)$ for an identity ID, compute $c = H_1(m, U)$ and verify that $(P, P_{pub}, U + cQ_{\mathsf{ID}}, V)$ is a valid DH-tuple.

6.2 Security

The security definition of ID-based digital signature schemes is given in [7]. We say that an ID-based digital signature scheme is $(t, q_I, q_S, q_H, q_{H_1}, \epsilon)$-secure if for any forger F who runs in time t,

$$\Pr(F \text{ can output a valid forgery}) < \epsilon,$$

where F can make at most q_I extract queries, at most q_S signing queries and at most q_H and q_{H_1} queries to the random oracle H and H_1, respectively.

Then from Theorem 3 and Lemma 1 of [1], we can obtain the following theorem.

Theorem 4. *If G is a (τ, t', ϵ')-GDH group, then the ID-based GDH signature scheme is $(t, q_I, q_S, q_H, q_{H_1}, \epsilon)$-secure, where*

$$t \geq (t'/2) - c_A \log_2 q(q_H + q_I) - poly(k),$$
$$\epsilon \leq \frac{(1 + q_{H_1})(q\sqrt{2eq_I\epsilon'} + 1) + (1 + q_{H_1} + q_S)q_S}{q},$$

and c_A is a small constant. Here e is the base of the natural logarithm.

7 Security Against Active Attacks of the Proposed ID-based Identification Scheme

In this section, We show that our proposed ID-based identification scheme as described in Section 5.2 is secure against active attacks if the one-more DH problem is hard, where the one-more DH assumption is a natural analogue of the one-more RSA inversion assumption which was first introduced in [2]. The same assumption and the discrete-log related assumption were later used in [3] to prove the security against impersonation under active and concurrent attacks for GQ and Schnorr identification schemes, respectively.

7.1 One-More DH Assumption

We briefly describe the one-more DH adversary. An one-more DH adversary is a randomized, polynomial-time algorithm M that gets input $(P, P_{pub} = sP)$ and has access to two oracles, namely the DH-oracle that given $Q \in G$ returns $sQ \in G$ and a challenge oracle that each time it is invoked (it takes no input), returns a random challenge point $W \in G$.

First, run $M(P, P_{pub})$ with its oracles. Let $W_1, \ldots, W_n$ denote the challenges returned by M's challenge oracle. M can ask at most $n - 1$ DH-oracle queries. We say that M wins if its output is a sequence of points $sW_1, \ldots, sW_n \in G$, meaning M solves the DH problem of all the challenge points. In other words, the one-more DH assumption states that it is computationally infeasible for the adversary to solve the DH problem of all the challenge points if its DH-oracle

queries are strictly less than its challenge oracle queries. (When the adversary makes one challenge query and no DH-oracle queries, this is the standard DH assumption.)

We say that the one-more DH problem is (t, ϵ)-hard if $\Pr(M \text{ wins}) < \epsilon$ for any M which runs in time t.

7.2 Security Proof

Theorem 5. *Let H be a random oracle from $\{0,1\}^*$ to G. If the one-more DH problem is (t', ϵ')-hard, then the ID-based identification scheme is (t, q_I, q_H, ϵ)-secure against active attacks, where*

$$t \geq (t'/2) - poly(k),\ \epsilon \leq \sqrt{e(1+q_I)\epsilon'} + (1/q).$$

The proof will be given in the full version of the paper.

8 ID-based Variants of GQ Schemes

GQ identification scheme is not ID-based as mentioned in Section 1.3. However, we can easily obtain an ID-based variant of GQ identification scheme by combining the FDH-RSA signature scheme with our transformation. Further, Coron showed a very tight security proof for the FDH-RSA signature scheme [9]. Hence we can obtain a tight security proof for the ID-based variant of GQ identification scheme directly from Theorem 1. Similarly, we can obtain an ID-based variant of GQ signature scheme. In particular, they are obtained by our transformation: RSA signature $\rightarrow$ ID-based GQ identification scheme $\rightarrow$ ID-based GQ signature. The details will be given in the full version of the paper.

9 Conclusion

We have formalized the concept of ID-based identification scheme. We have also presented a transformation from any digital signature scheme having a Δ-challenge ZK protocol to an ID-based identification scheme. A concrete example is given based on Boneh et al.'s GDH signature scheme. Eventually, by using Fiat-Shamir transformation, we reached at an ID-based signature scheme which is coincided with Cha and Cheon's scheme. However, we can achieve a tighter security reduction due to our transformation.

References

1. M. Abdalla, J. An, M. Bellare and C. Namprempre. From identification to signatures via the Fiat-Shamir transform: minimizing assumptions for security and forward-security. *Advances in Cryptology — EUROCRYPT '02, LNCS 2332,* pp. 418–433, Springer-Verlag, 2002.

2. M. Bellare, C. Namprempre, D. Pointcheval and M. Semanko. The power of RSA inversion oracles and the security of Chaum's RSA-based blind signature scheme. *Financial Cryptography 2001, LNCS 2339,* pp. 319–338, Springer-Verlag, 2002.
3. M. Bellare and A. Palacio. GQ and Schnorr identification schemes: proofs of security against impersonation under attive and concurrent attacks. *Advances in Cryptology — CRYPTO '02, LNCS 2442,* pp. 162–177, Springer-Verlag, 2002.
4. D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. *Advances in Cryptology — CRYPTO '01, LNCS 2139,* pp. 213–229, Springer-Verlag, 2001.
5. D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. *Siam Journal of Computing, Vol. 32,* pp. 586–615, 2003. Updated version of [4].
6. D. Boneh, B. Lynn and H. Shacham. Short signatures from the the weil pairing. *Advances in Cryptology — ASIACRYPT '01, LNCS 2248,* pp. 514–532, Springer-Verlag, 2001.
7. J. C. Cha and J. H. Cheon. An identity-based signature from gap Diffie-Hellman groups. *Public Key Cryptography — PKC '03, LNCS 2567,* pp. 18–30, Springer-Verlag, 2003.
8. C. Cocks. An identity based encryption scheme based on quadratic residues. *Cryptography and Coding, LNCS 2260,* pp. 360–363, Springer-Verlag, 2001.
9. J. Coron. On the exact security of full domain hash. *Advances in Cryptology — CRYPTO '00, LNCS 1880,* pp. 229–235, Springer-Verlag, 2000.
10. U. Feige, A. Fiat and A. Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology, Vol. 1,* pp. 77–94, Springer-Verlag, 1988.
11. A. Fiat and A. Shamir. How to prove yourself: practical solutions to identification and signature problems. *Advances in Cryptology — CRYPTO '86, LNCS 263,* pp. 186–194, Springer-Verlag, 1987.
12. L. Guillou and J. Quisquater. A practical zero-knowledge protocol fitted to security microprocessors minimizing both transmission and memory. *Advances in Cryptology — EUROCRYPT '88, LNCS 330,* pp. 123–128, Springer-Verlag, 1989.
13. F. Hess. Efficient identity based signature schemes based on pairings. *Selected Areas in Cryptography — SAC '02, LNCS 2595,* pp. 310–324, Springer-Verlag, 2002.
14. A. Kiayias and M. Yung. Extracting group signatures from traitor tracing schemes. *Advances in Cryptology — EUROCRYPT '03, LNCS 2656,* pp. 630–648, Springer-Verlag, 2003.
15. T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. *Advances in Cryptology — CRYPTO '92, LNCS 740,* pp. 31–53, Springer-Verlag, 1993.
16. K. G. Paterson. ID-based signatures from pairings on elliptic curves. *Electronic Letters, Vol. 38, No. 18,* pp. 1025–1026, 2002.
17. C. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology, Vol. 4,* pp. 161–174, Springer-Verlag, 1991.
18. R. Sakai, K. Ohgishi and M. Kasahara. Cryptosystems based on pairing. *2000 Symposium on Cryptography and Information Security — SCIS '00,* Okinawa, Japan, Jan. 26–28, 2000.
19. Z.-F. Zhang, J. Xu and D.-G. Feng. Attack on an identification scheme based on gap Diffie-Hellman problem. *IACR Cryptology ePrint Archive, Report 2003/153.* Available from `http://eprint.iacr.org/2003/153/`.

Identity-Based Threshold Decryption

Joonsang Baek[1] and Yuliang Zheng[2]

[1] School of Network Computing, Monash University,
McMahons Road, Frankston, VIC 3199, Australia
joonsang.baek@infotech.monash.edu.au
[2] Dept. Software and Info. Systems, UNC Charlotte, NC 28223, USA
yzheng@uncc.edu

Abstract. In this paper, we examine issues related to the construction of identity-based threshold decryption schemes and argue that it is important in practice to design an identity-based threshold decryption scheme in which a private key associated with an identity is shared. A major contribution of this paper is to construct the first identity-based threshold decryption scheme secure against chosen-ciphertext attack. A formal proof of security of the scheme is provided in the random oracle model, assuming the Bilinear Diffie-Hellman problem is computationally hard. Another contribution of this paper is, by extending the proposed identity-based threshold decryption scheme, to construct a mediated identity-based encryption scheme secure against more powerful attacks than those considered previously.

1 Introduction

Threshold decryption is particularly useful where the centralization of the power to decrypt is a concern. And the motivation of identity (ID)-based encryption originally proposed by Shamir [17] is to provide confidentiality without the need of exchanging public keys or keeping public key directories. A major advantage of ID-based encryption is that it allows one to encrypt a message by using a recipient's identifiers such as an email address.

A combination of these two concepts will allow one to build an "ID-based threshold decryption" scheme. One possible application of such a scheme can be considered in a situation where an identity denotes the name of the group sharing a decryption key. As an example, suppose that Alice wishes to send a confidential message to a committee in an organization. Alice can first encrypt the message using the identity (name) of the committee and then send over the ciphertext. Let us assume that Bob who is the committee's president has created the identity and hence has obtained a matching private decryption key from the Private Key Generator (PKG). Preparing for the time when Bob is away, he can share his private key out among a number of decryption servers in such a way that any committee member can successfully decrypt the ciphertext if, and only if, the committee member obtains a certain number of decryption shares from the decryption servers.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 262–276, 2004.

Another application of the ID-based threshold decryption scheme is to use it as a building block to construct a mediated ID-based encryption scheme [7]. The idea is to split a private key associated with the receiver Bob's ID into two parts, and give one share to Bob and the other to the Security Mediator (SEM). Accordingly, Bob can decrypt a ciphertext only with the help of the SEM. As a result, instantaneous revocation of Bob's privilege to perform decryption is possible by instructing the SEM not to help him any more.

In this paper, we deal with the problem of constructing an ID-based threshold decryption scheme which is efficient and practical while meets a strong security requirement. We also treat the problem of applying the ID-based threshold decryption scheme to design a mediated ID-based encryption scheme secure against more powerful attacks than those considered previously in the literature.

2 Preliminaries

We first review the "admissible bilinear map", which is the mathematical primitive that plays on central role in Boneh and Franklin's ID-based encryption scheme [5].

Bilinear Map. The admissible bilinear map $\hat{e}$ [5] is defined over two groups of the same prime-order q denoted by $\mathcal{G}$ and $\mathcal{F}$ in which the Computational Diffie-Hellman problem is hard. (By $\mathcal{G}^*$ and $\mathbb{Z}_q^*$, we denote $\mathcal{G} \setminus \{O\}$ where O is the identity element of $\mathcal{G}$, and $\mathbb{Z}_q \setminus \{0\}$ respectively.) We will use an additive notation to describe the operation in $\mathcal{G}$ while we will use a multiplicative notation for the operation in $\mathcal{F}$. In practice, the group $\mathcal{G}$ is implemented using a group of points on certain elliptic curves, each of which has a small MOV exponent [15], and the group $\mathcal{F}$ will be implemented using a subgroup of the multiplicative group of a finite field. The admissible bilinear map, denoted by $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$, has the following properties.

- Bilinear: $\hat{e}(aR_1, bR_2) = \hat{e}(R_1, R_2)^{ab}$, where $R_1, R_2 \in \mathcal{G}$ and $a, b \in \mathbb{Z}_q^*$.
- Non-degenerate: $\hat{e}$ does not send all pairs of points in $\mathcal{G} \times \mathcal{G}$ to the identity in $\mathcal{F}$. (Hence, if R is a generator of $\mathcal{G}$ then $\hat{e}(R, R)$ *is a* generator of $\mathcal{F}$.)
- Computable: For all $R_1, R_2 \in \mathcal{G}$, the map $\hat{e}(R_1, R_2)$ is efficiently computable.

Throughout this paper, we will simply use the term "Bilinear map" to refer to the admissible bilinear map defined above.

The "BasicIdent" *Scheme.* We now describe Boneh and Franklin's basic version of ID-based encryption scheme called "BasicIdent" which only gives semantic security (that is, indistinguishability under chosen plaintext attack).

In the setup stage, the PKG specifies a group $\mathcal{G}$ generated by $P \in \mathcal{G}^*$ and the Bilinear map $\hat{e} : \mathcal{G} \times \mathcal{G} \to \mathcal{F}$. It also specifies two hash functions $\mathsf{H}_1 : \{0,1\}^* \to \mathcal{G}^*$ and $\mathsf{H}_2 : \mathcal{F} \to \{0,1\}^l$, where l denotes the length of a plaintext.The PKG then picks a master key x uniformly at random from $\mathbb{Z}_q^*$ and computes a public key $Y_{\mathrm{PKG}} = xP$. The PKG publishes descriptions of the group $\mathcal{G}$ and $\mathcal{F}$ and the hash functions H_1 and H_2. Bob, the receiver, then contacts the PKG to get his private key $D_{\mathtt{ID}} = xQ_{\mathtt{ID}}$ where $Q_{\mathtt{ID}} = \mathsf{H}_1(\mathtt{ID})$. Alice, the sender, can now

encrypt her message $M \in \{0,1\}^l$ using Bob's identity $\mathtt{ID}$ by computing $U = rP$ and $V = \mathsf{H}_2(\hat{e}(Q_{\mathtt{ID}}, Y_{\mathrm{PKG}})^r) \oplus M$, where r is chosen at random from $\mathbb{Z}_q^*$ and $Q_{\mathtt{ID}} = \mathsf{H}_1(\mathtt{ID})$. The resulting ciphertext $C = (U, V)$ is sent to Bob. Bob decrypts C by computing $M = V \oplus \mathsf{H}_2(\hat{e}(D_{\mathtt{ID}}, U))$.

3 Related Work and Discussion

Boneh and Franklin's "Distributed PKG". In order to prevent a single PKG from full possession of the master key in ID-based encryption, Boneh and Franklin [5] suggested that the PKG's master key should be shared among a number of PKGs using the techniques of threshold cryptography, which they call "Distributed PKG". More precisely, the PKG's master key x is distributed into a number of PKGs in such a way that each of the PKG holds a share $x_i \in \mathbb{Z}_q^*$ of a Shamir's (t, n)-secret-sharing [16] of $x \in \mathbb{Z}_q^*$ and responds to a user's private key extraction request with $D_{\mathtt{ID}}^i = x_i Q_{\mathtt{ID}}$, where $Q_{\mathtt{ID}} = \mathsf{H}_1(\mathtt{ID})$. If the technique of [11] is used, one can ensure that the master key is jointly generated by PKGs so that the master key is not stored or computed in any single location.

As an extension of the above technique, Boneh and Franklin suggested that the distributed PKGs should function as decryption servers for threshold decryption. That is, each PKG responds to a decryption query $C = (U, V)$ in BasicIdent with $\hat{e}(x_i Q_{\mathtt{ID}}, U)$. However, we argue that this method is not quite practical in practice since it requires each PKG to be involved *at all times* (that is, *on-line*) in the generation of decryption shares because the value *"U"* changes whenever a new ciphertext is created. Obviously, this creates a bottleneck on the PKGs and also violates one of the basic requirements of an ID-based encryption scheme, "the PKG can be closed after key generation", which was envisioned by Shamir in his original proposal of ID-based cryptography [17]. Moreover, there is a scalability problem when the number of available distributed PKGs is not matched against the number of decryption servers required, say, there are only 3 available PKGs while a certain application requires 5 decryption servers.

Therefore, a better approach would be *sharing a private key associated with an identity* rather than sharing a master key of the PKG. In addition to its easy adaptability to the situation where an identity denotes a group sharing a decryption key as described in Section 1, an advantage of this approach is that one can fully utilize Boneh and Franklin's Distributed PKG method without the above-mentioned scalability problem, dividing the role of "distributed PKGs" from that of "decryption servers". That is, an authorized dealer (a representative of group, such as "Bob" described in Section 1, or a single PKG) may ask an identity to each of the "distributed PKGs" for a *partial* private key associated the identity. Having obtained enough partial private keys, the dealer can construct the whole private key and distribute it into the "decryption servers" in his domain at will while the master key remains secret from any parties.

Other Related Work on ID-Based Threshold Decryption. To our knowledge, other papers that have treated "threshold decryption" in the context of ID-based cryptography are [8] and [13]. Dodis and Yung [8] observed how threshold de-

cryption can be realized in Gentry and Silverberg [12]'s "hierarchical ID-based encryption" setting. Interestingly, their approach is to share a private key (not the master key of the PKG) obtained from a user at a higher level. Although this was inevitable in the hierarchical ID-based encryption setting and its advantage in general ID-based cryptography was not mentioned in [8], it is more sound approach than sharing the master key of the PKG as we discussed above. However, their threshold decryption scheme is very-sketched and chosen-ciphertext security for the scheme was not considered in [8]. More recently, Libert and Quisquater [13] also constructed an ID-based threshold decryption scheme. However, their approach was to share a master key of the PKG, which is different from ours. Moreover, our scheme gives chosen ciphertext security while Libert and Quisquater's scheme does not.

4 Security Notion for ID-based Threshold Decryption

4.1 Description of Generic ID-based Threshold Decryption

A generic ID-based threshold decryption scheme, which we denote by "$\mathcal{IDTHD}$", consists of algorithms GK, EX, DK, E, D, SV, and SC. Below, we describe each of the algorithms.

Like other ID-based cryptographic schemes, we assume the existence of a trusted PKG. The PKG runs the key/common parameter generation algorithm GK to generate its master/public key pair and all the necessary common parameters. The PKG's public key and the common parameters are given to every interested party.

On receiving a user's private key extraction request which consists of an identity, the PKG then runs the private key extraction algorithm EX to generate the private key associated with the requested identity.

An authorized dealer who possesses the private key associated with an identity can run the private key distribution algorithm DK to distribute the private key into n decryption servers. DK makes use of an appropriate secret-sharing technique to generate shares of the private key as well as verification keys that will be used for checking the validity of decryption shares. Each share of the private key and its corresponding verification key are sent to an appropriate decryption server. The decryption servers then keep their private key shares secret but publish the verification keys. It is important to note here that the entity that runs DK can vary flexibly depending on the cryptographic services that the PKG can offer. For example, if the PKG has an only functionality of issuing private keys, the authorized dealer that runs DK would be a normal user (such as Bob in the example given in Section 1) other than the PKG. However, if the PKG has other functionalities, for example, organizing threshold decryption, the PKG can run DK.

Given a user's identity, any user that wants to encrypt a plaintext can run the encryption algorithm E to obtain a ciphertext. A *legitimate* user that wants to decrypt a ciphertext gives it to the decryption servers requesting decryption

shares. The decryption servers then run the decryption share generation algorithm D taking the ciphertext as input and send the resulting decryption shares to the user. Note that the validity of the shares can be checked by running the decryption share verification algorithm SV. When the user collects valid decryption shares from at least t servers, the plaintext can be reconstructed by running the share combining algorithm SC.

4.2 Chosen Ciphertext Security for ID-based Threshold Decryption

We now define a security notion for the $\mathcal{IDTHD}$ scheme against chosen-ciphertext attack, which we call "IND-IDTHD-CCA".

Definition 1 (IND-IDTHD-CCA). Let $\mathsf{A}^{\mathsf{CCA}}$ be an attacker assumed to be a probabilistic Turing machine. Suppose that a security parameter k is given to $\mathsf{A}^{\mathsf{CCA}}$ as input. Now, consider the following game in which the attacker $\mathsf{A}^{\mathsf{CCA}}$ interacts with the "Challenger".

Phase 1: The Challenger runs the PKG's key/common parameter generation algorithm taking a security parameter k as input. The Challenger gives $\mathsf{A}^{\mathsf{CCA}}$ the resulting common parameter cp which includes the PKG's public key pk_{PKG}. However, the Challenger keeps the master key sk_{PKG} secret from $\mathsf{A}^{\mathsf{CCA}}$.

Phase 2: $\mathsf{A}^{\mathsf{CCA}}$ issues a number of private key extraction queries. We denote each of these queries by $\mathtt{ID}$. On receiving the identity query $\mathtt{ID}$, the Challenger runs the private key extraction algorithm on input $\mathtt{ID}$ and obtains a corresponding private key $sk_{\mathtt{ID}}$. Then, the Challenger returns $sk_{\mathtt{ID}}$ to $\mathsf{A}^{\mathsf{CCA}}$.

Phase 3: $\mathsf{A}^{\mathsf{CCA}}$ corrupts $t-1$ out of n decryption servers.

Phase 4: $\mathsf{A}^{\mathsf{CCA}}$ issues a target identity query $\mathtt{ID}^*$. On receiving $\mathtt{ID}^*$, the Challenger runs the private key extraction algorithm to obtain a private key $sk_{\mathtt{ID}^*}$ associated with the target identity. The Challenger then runs the private key distribution algorithm on input $sk_{\mathtt{ID}^*}$ with parameter (t, n) and obtains a set of private/verification key pairs $\{(sk_{\mathtt{ID}^*{}_i}, vk_{\mathtt{ID}^*{}_i})\}$, where $1 \leq i \leq n$. Next, the Challenger gives $\mathsf{A}^{\mathsf{CCA}}$ the private keys of corrupted decryption servers and the verifications keys of all the decryption servers. However, the private keys of uncorrupted servers are kept secret from $\mathsf{A}^{\mathsf{CCA}}$.

Phase 5: $\mathsf{A}^{\mathsf{CCA}}$ issues arbitrary private key extraction queries and arbitrary decryption share generation queries to the uncorrupted decryption servers. We denote each of these queries by $\mathtt{ID}$ and C respectively. On receiving $\mathtt{ID}$, the Challenger runs the private key extraction algorithm to obtain a private key associated with $\mathtt{ID}$ and returns it to $\mathsf{A}^{\mathsf{CCA}}$. The only restriction here is that $\mathsf{A}^{\mathsf{CCA}}$ is not allowed to query the target identity $\mathtt{ID}^*$ to the private key extraction algorithm. On receiving C, the Challenger runs the decryption share generation algorithm taking C and the target identity ID^* as input to obtain a corresponding decryption share and returns it to $\mathsf{A}^{\mathsf{CCA}}$.

Phase 6: $\mathsf{A}^{\mathsf{CCA}}$ outputs two equal-length plaintexts (M_0, M_1). Then the Challenger chooses a bit β uniformly at random and runs the encryption algorithm on input cp, M_β and $\mathtt{ID}^*$ to obtain a target ciphertext $C^* = \mathsf{E}(cp, \mathtt{ID}^*, M_\beta)$. Finally, the Challenger gives $(C^*, \mathtt{ID}^*)$ to $\mathsf{A}^{\mathsf{CCA}}$.

Phase 7: $\mathsf{A}^{\mathsf{CCA}}$ issues arbitrary private key extraction queries and arbitrary decryption share generation queries. We denote each of these queries by $\mathtt{ID}$ and C

respectively. On receiving ID, the Challenger runs the private key extraction algorithm to obtain a private key associated with ID and returns it to $\mathsf{A}^{\mathsf{CCA}}$. As Phase 5, the only restriction here is that $\mathsf{A}^{\mathsf{CCA}}$ is not allowed to query the target identity ID* to the private key extraction algorithm. On receiving C, the Challenger runs the decryption share generation algorithm on input C to obtain a corresponding decryption share and returns it to $\mathsf{A}^{\mathsf{CCA}}$. Differently from Phase 5, the target ciphertext C^* is not allowed to query in this phase.

Phase 8: $\mathsf{A}^{\mathsf{CCA}}$ outputs a guess $\beta \in \{0,1\}$.

We define $\mathsf{A}^{\mathsf{CCA}}$'s success as a function $\mathbf{Succ}^{\text{IND-IDTHD-CCA}}_{\mathcal{IDTHD},\mathsf{A}^{\mathsf{CCA}}}(k) = 2 \cdot \Pr[\tilde{\beta} = \beta] - 1$. The ID-based threshold decryption scheme $\mathcal{IDTHD}$ is said to be IND-IDTHD-CCA secure if, for any attacker $\mathsf{A}^{\mathsf{CCA}}$ whose running time is polynomially bounded, $\mathbf{Succ}^{\text{IND-IDTHD-CCA}}_{\mathcal{IDTHD},\mathsf{A}^{\mathsf{CCA}}}(k)$ is negligible in k.

5 Our ID-based Threshold Decryption Scheme

5.1 Building Blocks

First, we present necessary building blocks that will be used to construct our ID-based threshold decryption scheme. We remark that since our ID-based threshold decryption scheme is also of the Diffie-Hellman (DH)-type, it follows Shoup and Gennaro [18]'s framework for the design of DH-based threshold decryption schemes to some extent. However, our scheme has a number of features that distinguishes itself from the schemes in [18] due to the special property of the underlying group $\mathcal{G}$.

Publicly Checkable Encryption. Publicly checkable encryption is a particularly important tool for building threshold decryption schemes secure against chosen-ciphertext attack as discussed by Lim and Lee [14]. The main reason is that in the threshold decryption, the attacker has decryption shares as additional information as well as a ciphertext, hence there is a big chance for the attacker to get enough decryption shares to recover the plaintext before the validity of the ciphertext is checked. (Readers are referred to [14] and [18] for more detailed discussions on this issue.)

The public checkability of ciphertexts in threshold decryption schemes is usually given by non-interactive zero-knowledge (NIZK) proofs, e.g., [18,10]. However, we emphasize that in our scheme, this can be done *without* a NIZK proof, by simply creating a tag on the ElGamal [9] ciphertext as follows.

Let $M \in \{0,1\}^l$ be a message. Then, encrypt M by creating a ciphertext $C = (U, V, W) = (rP, \mathsf{H}_2(\kappa) \oplus M, r\mathsf{H}_3(U,V))$ where $\kappa = \hat{e}(\mathsf{H}_1(\mathtt{ID}), Y_{\mathrm{PKG}})^r$ for hash functions $\mathsf{H}_1 : \{0,1\}^* \to \mathcal{G}^*$, $\mathsf{H}_2 : \mathcal{F} \to \{0,1\}^l$, and $\mathsf{H}_3 : \mathcal{G}^* \times \{0,1\}^l \to \mathcal{G}^*$. Without recovering M during the decryption process (that is, leaving the ciphertext C intact), the validity of C can be checked by testing if $\hat{e}(P,W) = \hat{e}(U, H_3)$, where $H_3 = \mathsf{H}_3(U,V) \in \mathcal{G}^*$. Note that this validity test exploits the fact that the Decisional Diffie-Hellman (DDH) problem can be solved in polynomial time in the group $\mathcal{G}$, and passing the test implies that (P, U, H_3, W) is a Diffie-Hellman tuple since $(P, U, H_3, W) = (P, rP, sP, rsP)$ assuming that $H_3 = sP \in_R \mathcal{G}^*$ for some $s \in \mathbb{Z}_q^*$.

Sharing a Point on $\mathcal{G}$. In order to share a private key $D_{\mathrm{ID}} \in \mathcal{G}$, we need some trick. In what follows, we present a Shamir's (t, n)-secret-sharing over $\mathcal{G}$.

Let q be a prime order of a group $\mathcal{G}$ (of points on elliptic curve). Let $S \in \mathcal{G}^*$ be a point to share. Suppose that we have chosen integers t (a threshold) and n satisfying $1 \leq t \leq n < q$. First, we pick $R_1, R_2, \ldots, R_{t-1}$ at random from $\mathcal{G}^*$. Then, we define a function $F : \mathbb{N} \cup \{0\} \rightarrow \mathcal{G}$ such that $F(u) = S + \sum_{l=1}^{t-1} u^l R_l$. (Note that in practice, "picking R_l at random from $\mathcal{G}^*$" can be implemented by computing $r_l P$ for randomly chosen $r_l \in \mathbb{Z}_q^*$, where $P \in \mathcal{G}^*$ is a generator of $\mathcal{G}$.) We then compute $S_i = F(i) \in \mathcal{G}$ for $1 \leq i \leq n$ and send (i, S_i) to the i-th member of the group of cardinality n. When the number of shares reaches the threshold t, the function $F(u)$ can be reconstructed by computing $F(u) = \sum_{j \in \Phi} c_{uj}^{\Phi} S_j$ where $c_{uj}^{\Phi} = \prod_{\iota \in \Phi, \iota \neq j} \frac{u-\iota}{j-\iota} \in \mathbb{Z}_q$ is the Lagrange coefficient for a set $\Phi \subset \{1, \ldots, n\}$ such that $|\Phi| \geq t$.

Zero Knowledge Proof for the Equality of Two Discrete Logarithms Based on the Bilinear Map. To ensure that all decryption shares are consistent, that is, to give robustness to threshold decryption, we need a certain checking procedure. In contrast to the ciphertext validity checking mechanism of in our publicly checkable encryption presented above, we need a non-interactive zero-knowledge proof system since the share of the key κ is the element of the group $\mathcal{F}$, where the DDH problem is believed to be hard.

Motivated by [6] and [18], we construct a zero-knowledge proof of membership system for the language $L_{\mathsf{EDLog}_{P,\tilde{P}}^{\mathcal{F}}} \stackrel{\text{def}}{=} \{(\mu, \tilde{\mu}) \in \mathcal{F} \times \mathcal{F} | \log_g \mu = \log_{\tilde{g}} \tilde{\mu}\}$ where $g = \hat{e}(P, P)$ and $\tilde{g} = \hat{e}(P, \tilde{P})$ for generators P and $\tilde{P}$ of $\mathcal{G}$ (the groups $\mathcal{G}$ and $\mathcal{F}$ and the Bilinear map $\hat{e}$ are as defined in Section 2) as follows.

Suppose that $(P, \tilde{P}, g, \tilde{g})$ and $(\kappa, \tilde{\kappa}) \in L_{\mathsf{EDLog}_{P,\tilde{P}}^{\mathcal{F}}}$ are given to the Prover and the Verifier, and the Prover knows a secret $S \in \mathcal{G}^*$. The proof system which we call "ZKBm" works as follows.

- The Prover chooses a non-identity element T uniformly at random from $\mathcal{G}$ and computes $\gamma = \hat{e}(T, P)$ and $\tilde{\gamma} = \hat{e}(T, \tilde{P})$. The Prover sends γ and $\tilde{\gamma}$ to the Verifier.
- The Verifier chooses h uniformly at random from $\mathbb{Z}_q^*$ and sends it to the Prover.
- On receiving h, the Prover computes $L = T + hS \in \mathcal{G}$ and sends it to the Verifier. The Verifier checks if $\hat{e}(L, P) = \gamma \kappa^h$ and $\hat{e}(L, \tilde{P}) = \tilde{\gamma} \tilde{\kappa}^h$. If the equality holds then the Verifier returns *"Accept"*, otherwise, returns *"Reject"*.

The above protocol actually satisfies completeness, soundness and zero-knowledge against the honest Verifier (The proof is given in the full version of this paper [1].) Note that ZKBm can easily be converted to a NIZK proof, making the random challenge an output of a random oracle [2]. Note also that the above protocol can be viewed as a proof that $(g, \tilde{g}, \kappa, \tilde{\kappa})$ is a Diffie-Hellman tuple since if $(\kappa, \tilde{\kappa}) \in L_{\mathsf{EDLog}_{P,\tilde{P}}^{\mathcal{F}}}$ then $\kappa = g^x$ and $\tilde{\kappa} = \tilde{g}^x$ for some $x \in \mathbb{Z}_q^*$ and hence $(g, \tilde{g}, \kappa, \tilde{\kappa}) = (g, \tilde{g}, g^x, \tilde{g}^x) = (g, g^y, g^x, g^{xy})$ for some $y \in \mathbb{Z}_q^*$.

5.2 Description of Our Scheme – IdThdBm

We now describe our ID-based threshold decryption scheme. We call our scheme "IdThdBm", meaning "ID-based threshold decryption scheme from the bilinear map". IdThdBm consists of the following algorithms.

- $\mathsf{GK}(k)$: Given a security parameter k, this algorithm generates two groups $\mathcal{G}$ and $\mathcal{F}$ of the same prime order $q \geq 2^k$ and chooses a generator P of $\mathcal{G}$. Then, it specifies the Bilinear map $\hat{e} : \mathcal{G} \times \mathcal{G} \rightarrow \mathcal{F}$ and the hashfunctions H_1, H_2, H_3 and H_4 such that $\mathsf{H}_1 : \{0,1\}^* \rightarrow \mathcal{G}^*$; $\mathsf{H}_2 : \mathcal{F} \rightarrow \{0,1\}^l$; $\mathsf{H}_3 : \mathcal{G}^* \times \{0,1\}^l \rightarrow \mathcal{G}^*$; $\mathsf{H}_4 : \mathcal{F} \times \mathcal{F} \times \mathcal{F} \rightarrow \mathbb{Z}_q^*$, where l denotes the length of a plaintext. Next, it chooses the PKG's master key x uniformly at random from $\mathbb{Z}_q^*$ and computes the PKG's public key $Y_{\mathrm{PKG}} = xP$. Finally, it returns a common parameter $cp = (\mathcal{G}, q, P, \hat{e}, \mathsf{H}_1, \mathsf{H}_2, \mathsf{H}_3, \mathsf{H}_4, Y_{\mathrm{PKG}})$ while keeping the master key x secret.
- $\mathsf{EX}(cp, \mathtt{ID})$: Given an identity ID, this algorithm computes $Q_{\mathtt{ID}} = \mathsf{H}_1(\mathtt{ID})$ and $D_{\mathtt{ID}} = xQ_{\mathtt{ID}}$. Then, it returns the private key $D_{\mathtt{ID}}$ associated with ID.
- $\mathsf{DK}(cp, \mathtt{ID}, D_{\mathtt{ID}}, t, n)$ where $1 \leq t \leq n < q$: Given a private key $D_{\mathtt{ID}}$, the number of decryption servers n and a threshold parameter t, this algorithm first picks $R_1, R_2, \ldots, R_{t-1}$ at random from $\mathcal{G}^*$ and constructs $F(u) = D_{\mathtt{ID}} + \sum_{j=1}^{t-1} u^j R_j$ for $u \in \{0\} \cup \mathbb{N}$. It then computes each server Γ_i's private key $S_i = F(i)$ and verification key $y_i = \hat{e}(S_i, P)$ for $1 \leq i \leq n$. Subsequently, it secretly sends the distributed private key S_i and the verification key y_i to server Γ_i for $1 \leq i \leq n$. Γ_i then keeps S_i as secret while making y_i public.
- $\mathsf{E}(cp, \mathtt{ID}, m)$: Given a plaintext $M \in \{0,1\}^l$ and an identity ID, this algorithm chooses r uniformly at random from $\mathbb{Z}_q^*$, and subsequently computes $Q_{\mathtt{ID}} = \mathsf{H}_1(\mathtt{ID})$ and $\kappa = \hat{e}(Q_{\mathtt{ID}}, Y_{\mathrm{PKG}})^r$. It then computes

$$U = rP; V = \mathsf{H}_2(\kappa) \oplus M; W = r\mathsf{H}_3(U, V)$$

  and returns a ciphertext $C = (U, V, W)$.
- $\mathsf{D}(cp, S_i, C)$: Given a private key S_i of each decryption server and a ciphertext $C = (U, V, W)$, this algorithm computes $H_3 = \mathsf{H}_3(U, V)$ and checks if $\hat{e}(P, W) = \hat{e}(U, H_3)$.
  If C has passed the above test, this algorithm computes $\kappa_i = \hat{e}(S_i, U)$, $\tilde{\kappa_i} = \hat{e}(T_i, U)$, $\tilde{y_i} = \hat{e}(T_i, P)$, $\lambda_i = \mathsf{H}_4(\kappa_i, \tilde{\kappa_i}, \tilde{y_i})$, and $L_i = T_i + \lambda_i S_i$ for random $T_i \in \mathcal{G}$, and outputs $\delta_{i,C} = (i, \kappa_i, \tilde{\kappa_i}, \tilde{y_i}, \lambda_i, L_i)$. Otherwise, it returns $\delta_{i,C} = (i,$ *"Invalid Ciphertext"*$)$.
- $\mathsf{SV}(cp, \{y_i\}_{1 \leq i \leq n}, C, \delta_{i,C})$: Given a ciphertext $C = (U, V, W)$, a set of verification keys $\{y_1, \ldots, y_n\}$, and a decryption share $\delta_{i,C}$, this algorithm computes $H_3 = \mathsf{H}_3(U, V)$ and checks if $\hat{e}(P, W) = \hat{e}(U, H_3)$.
  If C has passed the above test then this algorithm does the following:
  - If $\delta_{i,C}$ is of the form $(i,$ *"Invalid Ciphertext"*$)$ then return *"Invalid Share"*.
  - Else parse $\delta_{i,C}$ as $(i, \kappa_i, \tilde{\kappa_i}, \tilde{y_i}, \lambda_i, L_i)$ and compute $\lambda_i' = \mathsf{H}_4(\kappa_i, \tilde{\kappa_i}, \tilde{y_i})$.
    - Check if $\lambda_i' = \lambda_i$, $\hat{e}(L_i, U)/\kappa_i^{\lambda_i'} = \tilde{\kappa_i}$ and $\hat{e}(L_i, P)/y_i^{\lambda_i'} = \tilde{y_i}$.
    - If the test above holds, return *"Valid Share"*, else output *"Invalid Share"*.

  Otherwise, does the following:
  - If $\delta_{i,C}$ is of the form $(i,$ *"Invalid Ciphertext"*$)$, return *"Valid Share"*, else output *"Invalid Share"*.

– $\mathsf{SC}(cp, C, \{\delta_{j,C}\}_{j\in\Phi})$: Given a ciphertext C and a set of valid decryption shares $\{\delta_{j,C}\}_{j\in\Phi}$ where $|\Phi| \geq t$, this algorithm computes $H_3 = \mathsf{H}_3(U, V)$ and checks if $\hat{e}(P, W) = \hat{e}(U, H_3)$.
If C has not passed the above test, this algorithm returns *"Invalid Ciphertext"*. (In this case, all the decryption shares are of the form $(i,$ *"Invalid Ciphertext"*$)$.)
Otherwise, it computes $\kappa = \prod_{j\in\Phi} \kappa_j^{c_{0,j}^{\Phi}}$ and $M = \mathsf{H}_2(\kappa) \oplus V$, and returns M.

5.3 Security Analysis – IdThdBm

Bilinear Diffie-Hellman Problem. First, we review the Bilinear Diffie-Hellman (BDH) problem, which was introduced by Boneh and Franklin [5].

Definition 2 (BDH). Let $\mathcal{G}$ and $\mathcal{F}$ be two groups of order q where q is prime, as defined in Section 2. Let $P \in \mathcal{G}^*$ be a generator of $\mathcal{G}$. Suppose that there exists a Bilinear map $\hat{e} : \mathcal{G} \times \mathcal{G} \rightarrow \mathcal{F}$. Let $\mathsf{A}^{\mathsf{BDH}}$ be an attacker modelled as a probabilistic Turing machine.

The BDH problem refers to the computational problem in which $\mathsf{A}^{\mathsf{BDH}}$ is to compute the BDH key $\hat{e}(P, P)^{abc}$ given $(\mathcal{G}, q, P, aP, bP, cP)$ and a security parameter k. We define $\mathsf{A}^{\mathsf{BDH}}$'s success as a function $\mathbf{Succ}^{\mathrm{BDH}}_{\mathcal{G},\mathsf{A}^{\mathsf{BDH}}}(k)$=Pr[$\mathsf{A}^{\mathsf{BDH}}$ outputs $\hat{e}(P, P)^{abc}$]. The BDH problem is said to be computationally intractable if, for any attacker $\mathsf{A}^{\mathsf{BDH}}$ whose running time is polynomially bounded, $\mathbf{Succ}^{\mathrm{BDH}}_{\mathcal{G},\mathsf{A}^{\mathsf{BDH}}}(k)$ is negligible in k.

Proof of Security. Regarding the security of the IdThdBm scheme, we obtain the following theorem. (For a more detailed proof, we refer readers to the full version of this paper [1].)

Theorem 1. *In the random oracle model, the* IdThdBm *scheme is IND-IDTHD-CCA secure if the BDH problem is computationally intractable.*

Proof. (Sketch) To prove the above theorem, we derive a non-ID-based threshold decryption scheme, which we call "ThdBm", from the IdThdBm scheme. Actually, ThdBm is the same as IdThdBm except that it does not have a private key extraction algorithm and hence the hash function $\mathsf{H}_1 : \{0,1\}^* \rightarrow \mathcal{G}^*$ is not used. The private key D of this scheme is generated by choosing Q and x uniformly at random from $\mathcal{G}^*$ and $\mathbb{Z}_q^*$ respectively, and computing $D = xQ$. The public key of this scheme consists of (Q, Y), where $Y = xP$. Note that the private key D is shared among n decryption servers. The encryption of a plaintext message $m \in \{0,1\}^l$ can be done by choosing r uniformly at random from $\mathbb{Z}_q^*$ and computing $U = rP$, $V = \mathsf{H}_2(\kappa) \oplus m$, and $W = r\mathsf{H}_3(U, V)$, where $d = \hat{e}(Q, Y)$ and $\kappa = d^r$.

As a first step, we show how to use the IND-IDTHD-CCA attacker $\mathsf{A}^{\mathsf{CCA}}$ for IdThdBm to construct an IND-THD-CCA attacker $\mathsf{B}^{\mathsf{CCA}}$ for ThdBm.(IND-THD-CCA denotes the chosen-ciphertext security notion for non-ID-based threshold decryption defined in [18].) First, $\mathsf{B}^{\mathsf{CCA}}$ gives Y as the PKG's public key $\mathsf{A}^{\mathsf{CCA}}$. $\mathsf{B}^{\mathsf{CCA}}$ then randomly chooses an index μ from the range $[1, q_{\mathsf{H}_1}]$ where q_{H_1} denotes the maximum number of queries made by $\mathsf{A}^{\mathsf{CCA}}$ to the random oracle H_1. By

$\mathtt{ID}_\mu$, we denote the μ-th query to the random oracle H_1. $\mathsf{B}^{\mathsf{CCA}}$ hopes $\mathtt{ID}_\mu$ to be a target identity $\mathtt{ID}^*$ that $\mathsf{A}^{\mathsf{CCA}}$ outputs at some stage. Now, if $\mathsf{A}^{\mathsf{CCA}}$ queries H_1 at $\mathtt{ID} \neq \mathtt{ID}_\mu$, $\mathsf{B}^{\mathsf{CCA}}$ responds with τP, where τ is randomly chosen from $\mathbb{Z}_q^*$. Otherwise, $\mathsf{B}^{\mathsf{CCA}}$ responds with Q. Similarly, if $\mathsf{A}^{\mathsf{CCA}}$ issues $\mathtt{ID} \neq \mathtt{ID}_\mu$ as a private key extraction query, $\mathsf{B}^{\mathsf{CCA}}$ responds to the query with τY, where τ is randomly chosen from $\mathbb{Z}_q^*$, and stops the simulation otherwise. (However, if $\mathtt{ID}_\mu = \mathtt{ID}^*$, this query is not allowed.) If $\mathsf{A}^{\mathsf{CCA}}$ issues decryption share generation queries after it submits the target identity, $\mathsf{B}^{\mathsf{CCA}}$ uses its decryption servers to answer those queries. Notice that if $\mathtt{ID}_\mu = \mathtt{ID}^*$, which happens with probability $1/q_{\mathsf{H}_1}$, the simulation is perfect.

The next step is to show how to use the IND-THD-CCA attacker $\mathsf{B}^{\mathsf{CCA}}$ for ThdBm to construct an attacker $\mathsf{A}^{\mathsf{BDH}}$ for solving the BDH problem. Suppose that $(\mathcal{G}, q, \hat{e}, P, aP, bP, cP)$ for random $a, b, c \in \mathbb{Z}_q^*$ are given to $\mathsf{A}^{\mathsf{BDH}}$. Assume that $\mathsf{B}^{\mathsf{CCA}}$ has access to the common parameter ($\mathcal{G}$, q, P, $\hat{e}$, H_2, H_3, H_4, Y, Q). First, $\mathsf{A}^{\mathsf{BDH}}$ replaces Y by bP and Q by cP. If $\mathsf{B}^{\mathsf{CCA}}$ corrupts a subset of $t-1$ servers, where t is a threshold parameter, $\mathsf{A}^{\mathsf{BDH}}$ assumes that the servers $\Gamma_1, \Gamma_2, \ldots, \Gamma_{t-1}$ have been corrupted without loss of generality. $\mathsf{A}^{\mathsf{BDH}}$ then chooses $S_1, S_2, \ldots, S_{t-1}$ uniformly at random from $\mathcal{G}$ and computes $y_i = \hat{e}(Q, Y)^{c_{i0}^{\Phi}} \prod_{j=1}^{t-1} \hat{e}(S_j, P)^{c_{ij}^{\Phi}}$, where $t \leq i \leq n$ and c_{ij}^{Φ} denotes the Lagrange coefficient for a set $\Phi = \{0, 1, \ldots, t-1\}$. $\mathsf{A}^{\mathsf{BDH}}$ sends y_i to each of the uncorrupted decryption servers, that is, $\mathsf{A}^{\mathsf{BDH}}$ replaces the verification keys with the new y_i computed above.

Whenever the random oracle H_3 is queried at some point by $\mathsf{B}^{\mathsf{CCA}}$, $\mathsf{A}^{\mathsf{BDH}}$ picks s uniformly at random from $\mathbb{Z}_q^*$, computes $H_3 = sY$, and responds $\mathsf{B}^{\mathsf{CCA}}$ with it. On receiving queries to other random oracles, $\mathsf{A}^{\mathsf{BDH}}$ picks values at random from the ranges of the random oracles, and responds with them.

When $\mathsf{B}^{\mathsf{CCA}}$ submits two plaintexts (M_0, M_1) to the encryption oracle, $\mathsf{A}^{\mathsf{BDH}}$ creates a target ciphertext $C^* = (U^*, V^*, W^*)$ as follows. First, $\mathsf{A}^{\mathsf{BDH}}$ sets $U^* = aP$. $\mathsf{A}^{\mathsf{BDH}}$ then picks a string V^* at random from $\{0,1\}^l$, computes $H_3^* = s^*P$ for random $s^* \in \mathbb{Z}_q^*$, and sets $H_3^* = \mathsf{H}_3(U^*, V^*)$. $\mathsf{A}^{\mathsf{BDH}}$ also computes $W^* = s^*U^*$. Having created C^*, $\mathsf{A}^{\mathsf{BDH}}$ returns it to $\mathsf{B}^{\mathsf{CCA}}$ as a target ciphertext. Note here that $\hat{e}(P, W^*) = \hat{e}(U^*, H_3^*)$ since $(P, U^*, H_3^*, W^*) = (P, aP, s^*P, s^*aP)$ and hence is a legitimate Diffie-Hellman tuple. Therefore, as long as $\mathsf{B}^{\mathsf{CCA}}$ does not query the random oracle H_2 at the point $\hat{e}(P, P)^{abc}$, the simulation is perfect. However, happening such an event means that $\mathsf{A}^{\mathsf{BDH}}$ is able to solve the BDH problem. So $\mathsf{A}^{\mathsf{BDH}}$ simulates $\mathsf{B}^{\mathsf{CCA}}$'s view up to this event.

Now, suppose that $\mathsf{A}^{\mathsf{CCA}}$ has already made a query (U, V) to the random oracle H_3. By the construction of the simulator for H_3, we have $H_3 = \mathsf{H}(U, V) = sY$ for random $s \in \mathbb{Z}_q^*$. Since $\mathsf{A}^{\mathsf{BDH}}$ knows the value s, $\mathsf{A}^{\mathsf{BDH}}$ can compute $K = (1/s)W$ and hence $\kappa = \hat{e}(Q, K)$. Note here that $(1/s)W = (1/s)rsY = rY = rxP$ and $Q = cP$. Then, $\mathsf{A}^{\mathsf{BDH}}$ computes $\kappa_i = \kappa^{c_{i0}^{\Phi}} \prod_{j=1}^{t-1} \hat{e}(S_j, U)^{c_{ij}^{\Phi}}$ for $t \leq i \leq n$. It is easy to check κ_i is a correct i-th share of the BDH key $\kappa = \hat{e}(Q, Y)^r$.

The rest is a simulation of a full decryption share $\delta_{i,C} = (i, \kappa_i, \tilde{\kappa_i}, \tilde{y_i}, L_i)$. This can easily be done by the zero-knowledge simulation technique, responding to queries to the random oracle H_4 with an element randomly chosen from $\mathbb{Z}_q^*$. □

6 Application to Mediated ID-based Encryption

6.1 Security Issues in Mediated ID-based Encryption

The main motivation of mediated cryptography [4] is to revoke a user's privilege to perform cryptographic operations such as decrypting ciphertexts or signing messages *instantaneously*. In [4], Boneh et al. constructed the first mediated encryption and signature schemes using the RSA primitive. Their idea is to split a user's private key into two parts and give one piece to the on-line Security Mediator (SEM) and the other to the user. To decrypt or sign, the user must acquire a message-specific token which is associated with the SEM part of private key from the SEM. As a result, revocation is achieved by instructing the SEM not to issue tokens for the user.

Recently, the problem of realizing mediated encryption in the ID-based setting was considered by Ding and Tsudik [7]. They proposed an ID-based mediated encryption scheme based on RSA-OAEP [3]. Although their scheme offers good performance and practicality, it has a drawback which stems from the fact that a common RSA modulus is used for all the users within the system and hence, to guarantee the security of Ding and Tsudik's scheme, one should assume that the SEM's private key must be protected throughout the life of the system.

As an alternative to Ding and Tsudik's solution, Libert and Quisquater [13] proposed a new mediated ID-based encryption scheme based on Boneh and Franklin's ID-based encryption scheme. In term of security, it has an advantage over Ding and Tsudik's scheme in a sense that a compromise of the SEM's private key does not lead to a break of the whole system. In contrast to this positive result, Libert and Quisquater observed that *even though the SEM's private key is protected,* their scheme as well as Ding and Tsudik's scheme are not secure against "inside attack" in which the attacker who possesses the user part of private key conducts chosen-ciphertext attack. As a result, it should be strictly assumed in those schemes that users' private keys must be protected to ensure chosen-ciphertext security. In practice, this assumption is fairly strong in that there may be more chance for users to compromise their private keys than the SEM does since the SEM is usually assumed to be a trusted entity configured by a system administrator.

However, in the following section, we present a new mediated ID-based encryption scheme based on our `IdThdBm` scheme, which is secure against ciphertext attack in a *strong* sense, that is, secure against chosen-ciphertext attack conducted by the attacker that obtains the user part of private key.

6.2 Description of Our Scheme – `mIdeBm`

We describe our mediated ID-based encryption scheme "`mIdeBm`" based on the `IdThdBm` scheme with $(t, n) = (2, 2)$ as follows.

- **Setup:** Given a security parameter k, the PKG runs the key generation algorithm of IdThdBm. The output of this algorithm $cp = (\mathcal{G}, q, P, \hat{e}, \mathsf{H}_1, \mathsf{H}_2, \mathsf{H}_3, \mathsf{H}_4, Y_{\mathrm{PKG}})$ is as defined in the description of IdThdBm. Note that cp is given to all interested parties while the master key x is kept secret within the PKG.
- **Keygen:** Given a user's identity ID, the PKG computes $Q_{\mathrm{ID}} = \mathsf{H}_1(\mathrm{ID})$ and $D_{\mathrm{ID}} = xQ_{\mathrm{ID}}$. It then splits D_{ID} using the (2,2)-secret-sharing technique as follows[3].
  - Pick R at random from $\mathcal{G}^*$ and construct $F(u) = D_{\mathrm{ID}} + uR$ for $u \in \{0\} \cup \mathbb{N}$.
  - Compute $D_{\mathrm{ID},sem} = F(1)$ and $D_{\mathrm{ID},user} = F(2)$.

  The PKG gives $D_{\mathrm{ID},sem}$ to the SEM and $D_{\mathrm{ID},user}$ to the user.
- **Encrypt:** Given a plaintext $M \in \{0,1\}^l$ and a user's identity ID, a sender creates a ciphertext $C = (U, V, W)$ such that

$$U = rP; V = \mathsf{H}_2(\kappa) \oplus M; W = r\mathsf{H}_3(U, V),$$

  where $\kappa = \hat{e}(\mathsf{H}_1(\mathrm{ID}), Y_{\mathrm{PKG}})^r$ for random $r \in \mathbb{Z}_q^*$.
- **Decrypt:** When receiving $C = (U, V, W)$, a user forwards it to the SEM. The SEM and the user perform the following in parallel.
  - SEM (We call this procedure "SEM oracle"):
    1. Check if the user's identity ID is revoked. If it is, return "ID *Revoked*".
    2. Otherwise, do the following:
       * Compute $H_3 = \mathsf{H}_3(U, V)$ and check if $\hat{e}(P, W) = \hat{e}(U, H_3)$. If C has passed this test, compute $\kappa_{sem} = \hat{e}(D_{\mathrm{ID},sem}, U)$ and send $\delta_{\mathrm{ID},sem,C} = (sem, \kappa_{sem})$ to the user. Otherwise, send $\delta_{\mathrm{ID},sem,C} =$ (*sem, "Invalid Ciphertext"*) to the user.
  - User (We call this procedure "User oracle"):
    1. Compute $H_3 = \mathsf{H}_3(U, V)$ and check if $\hat{e}(P, W) = \hat{e}(U, H_3)$. If C has passed this test, compute $\kappa_{user} = \hat{e}(D_{\mathrm{ID},user}, U)$. Otherwise, return *"Reject"* and terminate.
    2. Get $\delta_{\mathrm{ID},sem,C}$ from the SEM and do the following:
       * If $\delta_{\mathrm{ID},sem,C}$ is of the form (*sem, "Invalid Ciphertext"*), return *"Reject"* and terminate. Otherwise, compute $\kappa = \kappa_{sem}^{c_{01}^{\Phi}} \kappa_{user}^{c_{02}^{\Phi}}$ where c_{01}^{Φ} and c_{02}^{Φ} denote the Lagrange coefficients for the set $\Phi = \{1, 2\}$ and $M = \mathsf{H}_2(\kappa) \oplus V$, and return M.

Notice that in the SEM oracle of the above scheme, the validity of a ciphertext is checked before generating a token in the same way as the decryption share generation algorithm of IdThdBm does.

6.3 Security Analysis - mIdeBm

In this section, we show that the chosen-ciphertext security of the above scheme against the strong attacker that obtains the user part of private key is relative to the IND-IDTHD-CCA (Definition 1) security of the (2, 2)-IdThdBm scheme.

To begin with, we define IND-mID-sCCA (indistinguishability of mediated ID-based encryption against strong chosen-ciphertext attack), which is similar to IND-mID-wCCA ("w" stands for "weak") defined in [13] but assumes the stronger attacker that can corrupt users to get their private keys.

[3] In this particular case of (2, 2)-secret-sharing, one may share D_{ID} by taking a random $D_{\mathrm{ID},sem}$ and computing $D_{\mathrm{ID},user} = D_{\mathrm{ID}} - D_{\mathrm{ID},sem}$ for efficiency.

Definition 3 (IND-mID-sCCA). Let $\mathsf{A}^{\mathsf{CCA}'}$ be an attacker that defeats the IND-mID-sCCA security of an mediated ID-based encryption scheme $\mathcal{MIDE}$ which consists of Setup, Keygen, Encrypt and Decrypt algorithms. (For details of these algorithms, readers are referred to mIdeBm given in Section 6.2.) We assume that $\mathsf{A}^{\mathsf{CCA}'}$ is a probabilistic Turing machine taking a security parameter k as input. Consider the following game in which the attacker $\mathsf{A}^{\mathsf{CCA}'}$ interacts with the "Challenger".

Phase 1: The Challenger runs the Setup algorithm taking a security parameter k. The Challenger then gives the common parameter to $\mathsf{A}^{\mathsf{CCA}'}$.
Phase 2: Having obtained the common parameter, $\mathsf{A}^{\mathsf{CCA}'}$ issues the following queries.

- "User key extraction" query ID: On receiving this query, the Challenger runs the Keygen algorithm to obtain the user part of private key and sends it to $\mathsf{A}^{\mathsf{CCA}'}$.
- "SEM key extraction" query ID: On receiving this query, the Challenger runs the Keygen algorithm to obtain the SEM part of private key and sends it to $\mathsf{A}^{\mathsf{CCA}'}$.
- "SEM oracle" query (ID, C): On receiving this query, the Challenger runs the Keygen algorithm to obtain a SEM part of private key. Taking the resulting private key as input, the Challenger runs the SEM oracle in the Decrypt algorithm to obtain a decryption token for C and sends it to $\mathsf{A}^{\mathsf{CCA}'}$.
- "User oracle" query (ID, C): On receiving this query, the Challenger runs the Keygen algorithm to obtain a User part of private key. Taking the resulting private key as input, the Challenger runs the User oracle in the Decrypt algorithm to obtain a decryption token for C and sends it to $\mathsf{A}^{\mathsf{CCA}'}$.

Phase 3: $\mathsf{A}^{\mathsf{CCA}'}$ selects two equal-length plaintexts (M_0, M_1) and a target identity ID * which was not queried before. On receiving (M_0, M_1) and ID*, the Challenger runs the Keygen algorithm to obtain User and SEM parts of the private key associated with ID*. The Challenger then chooses $\beta \in \{0, 1\}$ at random and creates a target ciphertext C^* by encrypting M_β under the target identity ID*. The Challenger gives the target ciphertext and *the User part of the private key* to $\mathsf{A}^{\mathsf{CCA}'}$.
Phase 4: $\mathsf{A}^{\mathsf{CCA}'}$ continues to issue "User key extraction" query ID $\neq$ ID*, "SEM key extraction" query ID $\neq$ ID*, "SEM oracle" query (ID, C) $\neq$ (ID*, C*), and "User oracle" query (ID, C) $\neq$ (ID*, C^*). The details of these queries are as described in Phase 2.
Phase 5: $\mathsf{A}^{\mathsf{CCA}'}$ outputs a guess $\tilde{\beta} \in \{0, 1\}$.

We define $\mathsf{A}^{\mathsf{CCA}'}$'s success as a function $\mathbf{Succ}^{\mathrm{IND-mID-sCCA}}_{\mathcal{MIDE},\mathsf{A}^{\mathsf{CCA}'}}(k) = 2 \cdot \Pr[\tilde{\beta} = \beta] - 1$. The mediated ID-based encryption scheme $\mathcal{MIDE}$ is said to be IND-mID-sCCA secure if, for any attacker $\mathsf{A}^{\mathsf{CCA}}$ whose running time is polynomially bounded, $\mathbf{Succ}^{\mathrm{IND-mID-sCCA}}_{\mathcal{MIDE},\mathsf{A}^{\mathsf{CCA}'}}(k)$ is negligible in k.

We now state and prove the following theorem. (Readers are referred to [1] for a more detailed proof.)

Theorem 2. *If the* (2, 2)-IdThdBm *scheme is IND-IDTHD-CCA secure then the* mIdeBm *scheme is IND-mID-sCCA secure.*

Proof. (Sketch) We show how to use the IND-mID-sCCA attacker $\mathsf{A}^{CCA'}$ for mIdeBm to construct an IND-IDTHD-CCA attacker A^{CCA} for IdThdBm.

When $\mathsf{A}^{CCA'}$ issues a new "User key extraction" or "SEM key extraction" query, which is an ID, A^{CCA} forwards ID to its Challenger as a private key extraction query, obtains a private key D_{ID} associated with ID, and gives D_{ID} to $\mathsf{A}^{CCA'}$. Having done this, A^{CCA} splits D_{ID} into $D_{\mathrm{ID},sem}$ and $D_{\mathrm{ID},user}$ using the (2, 2)-secret-sharing technique. A^{CCA} then adds $\langle \mathrm{ID}, D_{\mathrm{ID},user} \rangle$ and $\langle \mathrm{ID}, D_{\mathrm{ID},sem} \rangle$ to User KeyList and SEM KeyList respectively. Using these lists, A^{CCA} answers $\mathsf{A}^{CCA'}$'s "SEM oracle" and "User oracle" queries, each of which consists of (ID, C). If necessary, A^{CCA} forwards the ID in those queries to its Challenger to get a private key associated with it. It should be emphasized here that A^{CCA} always checks the validity of the ciphertext $C = (U, V, W)$ by testing whether $\hat{e}(P, W)$ equals to $\hat{e}(U, \mathsf{H}_3(U, V))$. If C does not pass this test, A^{CCA} rejects it.

Once $\mathsf{A}^{CCA'}$ issues two equal-length plaintexts (M_0, M_1) and a target identity ID*, A^{CCA} forwards $(M_0, M_1, \mathrm{ID}^*)$ to its Challenger. On receiving $(M_0, M_1, \mathrm{ID}^*)$, the Challenger runs the private key extraction algorithm of IdThdBm to get a private key D_{ID^*} associated with ID* and runs the private key distribution algorithm of IdThdBm to split D_{ID^*} into $D_{\mathrm{ID}^*,sem}$ and $D_{\mathrm{ID}^*,user}$. The Challenger gives $D_{\mathrm{ID}^*,user}$ to A^{CCA} as a corrupted party's private key. A^{CCA} then sends this back to $\mathsf{A}^{CCA'}$. In doing so, the *strong* attacker $\mathsf{A}^{CCA'}$ possesses the user part of private key. Now, the Challenger chooses $\beta \in \{0, 1\}$ at random and runs the encryption algorithm E of IdThdBm taking (M_β, ID^*) as input and gets a target ciphertext C^*. The Challenger gives it to A^{CCA}. Then, A^{CCA} sends C^* back to $\mathsf{A}^{CCA'}$.

A^{CCA} answers "User key extraction", "SEM key extraction", "SEM oracle", and "User oracle" queries in the same way it did before. Note, however, that the cases when (ID, C^*) and (ID*,C) are asked as "SEM oracle" and "User oracle" queries should be handled at this stage. Especially, A^{CCA} uses its decryption servers to handle the query (ID*,C).

Finally, if $\mathsf{A}^{CCA'}$ outputs a guess $\beta' \in \{0, 1\}$, A^{CCA} returns it as its guess. □

7 Concluding Remarks

In this paper, we discussed the issues related to the realization of ID-based threshold decryption and proposed the first threshold ID-based decryption scheme provably secure against chosen-ciphertext attack. We also showed how our ID-based threshold decryption scheme can result in a mediated ID-based encryption scheme secure against "inside attack", whereby an attacker who possesses a user part of private key conducts chosen-ciphertext attack.

Interesting future research would be finding more security applications where "ID-based threshold decryption" is particularly useful.

Acknowledgement

The authors are grateful to anonymous referees for their helpful comments. The first author also thanks Ron Steinfeld and John Malone-Lee for their valuable comments on the earlier version of this paper.

References

1. J. Baek and, Y. Zheng, *Identity-Based Threshold Decryption,* IACR ePrint Archive Report 2003/164.
2. M. Bellare and P. Rogaway, *Random Oracles are Practical: A Paradigm for Designing Efficient Protocols,* Proceedings of the First ACM Conference on Computer and Communications Security 1993, pages 62–73.
3. M. Bellare and P. Rogaway, *Optimal Asymmetric Encryption,* Advances in Cryptology -Proceedings of Eurocrypt '94, LNCS 950, Springer-Verlag 1994, pages 92–111.
4. D. Boneh, X. Ding, G. Tsudik and C. Wong, *A Method for Fast Revocation of Public Key Certificates and Security Capabilities,* Proceedings of the 10th USENIX Security Symposium, USENIX, 2001.
5. D. Boneh and M. Franklin, *Identity-Based Encryption from the Weil Pairing,* Proceedings of CRYPTO 2001, LNCS 2139, Springer-Verlag 2001, pages 213–229.
6. D. Chaum and T. Perderson, *Wallet Databases with Observers,* Proceedings of CRYPTO '92, LNCS 740, Springer-Verlag 1992, pages 89–105.
7. X. Ding and G. Tsudik, *Simple Identity-Based Cryptography with Mediated RSA,* Proceedings CT-RSA 2003, LNCS 2612, Springer-Verlag 2003, pages 192–209.
8. Y. Dodis and M Yung, *Exposure-Resilience for Free: The Hierarchical ID-based Encryption Case,* Proceedings of IEEE Security in Storage Workshop 2002, pages 45–52.
9. T. ElGamal, *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms,* IEEE Trans. Info. Theory, 31, 1985, pages 469–472.
10. P. Fouque and D. Pointcheval, *Threshold Cryptosystems Secure Chosen-Ciphertext Attacks,* Proceedings of ASIACRYPT 2001, LNCS 2248, Springer-Verlag 2001, pages 351–368.
11. R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, *Secure Distributed Key Generation for Discrete-Log Based Cryptosystem,* Proceedings of EUROCRYPT '99, LNCS 1592, Springer-Verlag 1999, pages 295–310.
12. C. Gentry and A. Silverberg, *Hierarchical ID-Based Cryptography,* Proceedings of ASIACRYPT 2002, LNCS 2501, Springer-Verlag 2002, pages 548–566.
13. B. Libert and J. Quisquater, *Efficient Revocation and Threshold Pairing Based Cryptosystems,* Principles of Distributed Computing (PODC) 2003.
14. C. Lim and P. Lee, *Another Method for Attaining Security Against Adaptively Chosen Ciphertext Attack,* Proceedings of CRYPTO '93, LNCS 773, Springer-Verlag 1993, pages 410–434.
15. A. J. Menezes, T. Okamoto, and S. A. Vanstone: *Reducing Elliptic Curve Logarithms to a Finite Field,* IEEE Tran. on Info. Theory, Vol. 31, pages 1639–1646, IEEE, 1993.
16. A. Shamir, *How to Share a Secret,* Communications of the ACM, Vol. 22, 1979, pages 612–613.
17. A. Shamir, *Identity-based Cryptosystems and Signature Schemes,* Proceedings of CRYPTO '84, LNCS 196, Springer-Verlag 1984, pages 47–53.
18. V. Shoup and R. Gennaro, *Securing Threshold Cryptosystems against Chosen Ciphertext Attack,* Journal of Cryptology, Vol. 15, Springer-Verlag 2002, pages 75–96.

An Efficient Signature Scheme from Bilinear Pairings and Its Applications

Fangguo Zhang, Reihaneh Safavi-Naini, and Willy Susilo

School of Information Technology and Computer Science
University of Wollongong, NSW 2522 Australia
{fangguo, rei, wsusilo}@uow.edu.au

Abstract. In Asiacrypt2001, Boneh, Lynn, and Shacham [8] proposed a short signature scheme (BLS scheme) using bilinear pairing on certain elliptic and hyperelliptic curves. Subsequently numerous cryptographic schemes based on BLS signature scheme were proposed. BLS short signature needs a special hash function [6, 1, 8]. This hash function is probabilistic and generally inefficient. In this paper, we propose a new short signature scheme from the bilinear pairings that unlike BLS, uses general cryptographic hash functions such as SHA-1 or MD5, and does not require special hash functions. Furthermore, the scheme requires less pairing operations than BLS scheme and so is more efficient than BLS scheme. We use this signature scheme to construct a ring signature scheme and a new method for delegation. We give the security proofs for the new signature scheme and the ring signature scheme in the random oracle model.

Keywords: *Short signature, Bilinear pairings, ID-based cryptography, Ring signature, Proxy signature*

1 Introduction

In recent years, bilinear pairings have found various applications in cryptography and have allowed us to construct some new cryptographic schemes [5, 6, 7, 8, 11, 20, 23, 27]. BLS scheme is a signature scheme that uses bilinear pairings and has the shortest length among signature schemes in classical cryptography. The scheme is based on Weil pairing and can be obtained from the private key extraction process of Boneh-Franklin's [6] ID-based encryption scheme. BLS short signature needs a special hash function, i.e., an admissible encoding function called MapToPoint that is also used by most conventional cryptographic schemes from pairings. Although there has been much discussions on the construction of such hash algorithm [1, 8], to our knowledge, all these algorithms are still probabilistic and there is no deterministic polynomial time algorithm for them.

The Computational Diffie-Hellman Problem (CDHP) is a well-studied problem and its hardness is widely believed to be closely related to the hardness of the

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 277–290, 2004.

Discrete Logarithm Problem (DLP). There are two variations of CDHP: Inverse Computational Diffie-Hellman Problem (Inv-CDHP) and Square Computational Diffie-Hellman Problem (Squ-CDHP).

In this paper, we propose a new short signature scheme that is constructed from Inv-CDHP based on bilinear pairing and does not require any special hash function. We note that in pairing based cryptosystems, the computation of the pairing is the most time-consuming. Although numerous papers discuss the complexity of pairings and how to speed up the pairing computation [2, 11], the computation of the pairing still remains time-consuming. Our new scheme uses less pairing operations than BLS scheme, and hence, is more efficient than BLS scheme. Based on the new signature scheme, we propose a ring signature scheme and a new method for delegation and some proxy signature schemes. We prove the security of the new signature scheme and the corresponding ring signature scheme in the random oracle model (the cryptographic hashing function (such as MD5 or SHA-1) is seen as an oracle which produces a random value for each new query).

The rest of the paper is organized as follows: The next section briefly explains the bilinear pairing and some problems related to pairings. Section 3 gives the new basic signature scheme and its security analysis. Based on this basic signature scheme, we give a ring signature scheme and some proxy signature schemes in Section 5 and 6, respectively. Section 7 concludes this paper.

2 Bilinear Pairing and Some Problems

Let $\mathbb{G}_1$ be a cyclic additive group generated by P, whose order is a prime q, and $\mathbb{G}_2$ be a cyclic multiplicative group with the same order q. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a map with the following properties:

1. **Bilinearity:** $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1, a, b \in \mathbb{Z}_q$
2. **Non-degeneracy:** There exists $P, Q \in \mathbb{G}_1$ such that $e(P, Q) \neq 1$, in other words, the map does not send all pairs in $\mathbb{G}_1 \times \mathbb{G}_1$ to the identity in $\mathbb{G}_2$;
3. **Computability:** There is an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in \mathbb{G}_1$.

In our setting of prime order groups, the **Non-degeneracy** is equivalent to $e(P, Q) \neq 1$ for all $P, Q \in \mathbb{G}_1$. So, when P is a generator of $\mathbb{G}_1$, $e(P, P)$ is a generator of $\mathbb{G}_2$. Such a bilinear map is called a bilinear pairing (more precisely, called an admissible bilinear pairing).

We consider the following problems in the additive group $(\mathbb{G}_1; +)$.

- **Discrete Logarithm Problem (DLP):** Given two group elements P and Q, find an integer $n \in \mathbb{Z}_q^*$, such that $Q = nP$ whenever such an integer exists.
- **Decision Diffie-Hellman Problem (DDHP):** For $a, b, c \in \mathbb{Z}_q^*$, given P, aP, bP, cP decide whether $c \equiv ab \bmod q$.
- **Computational Diffie-Hellman Problem (CDHP):** For $a, b \in \mathbb{Z}_q^*$, given P, aP, bP, compute abP.

There are two variations of CDHP:

- **Inverse Computational Diffie-Hellman Problem (Inv-CDHP):** For $a \in \mathbb{Z}_q^*$, given P, aP, compute $a^{-1}P$.
- **Square Computational Diffie-Hellman Problem (Squ-CDHP):** For $a \in \mathbb{Z}_q^*$, given P, aP, compute a^2P.

The following theorem relates these problemes [17, 22].

Theorem 1. *CDHP, Inv-CDHP and Squ-CDHP are polynomial time equivalent.*

Assumptions: We assume that DLP, CDHP Inv-CDHP and Squ-CDHP are hard, which means there is no polynomial time algorithm to solve any of them with non-negligible probability.

A *Gap Diffie-Hellman (GDH) group* is a group which the DDHP is easy but the CDHP is hard in it. From bilinear pairing, we can obtain the GDH group. Such groups can be found on supersingular elliptic curves or hyperelliptic curves over finite field, and the bilinear parings can be derived from the Weil or Tate pairing. For more details, we refer the readers to [6, 9, 13].

All schemes in this paper can work on any GDH group. Throughout this paper, we define the system parameters in all schemes as follows. Let P be a generator of $\mathbb{G}_1$ with order q, the bilinear pairing is given by $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. These system parameters can be obtained using a **GDH Parameter Generator** $\mathcal{IG}$ [6]. Define a cryptographic hash function $H : \{0,1\}^* \rightarrow \{0,1\}^\lambda$, where $|q| \geq \lambda \geq 160$.

3 New Short Signature Scheme from Bilinear Pairings

3.1 The Basic Signature Scheme

A signature scheme consists of the following four algorithms : a parameter generation algorithm ParamGen, a key generation algorithm KeyGen, a signature generation algorithm Sign and a signature verification algorithm Ver.

We describe the new signature scheme as follows:

1. ParamGen. The system parameters are $\{\mathbb{G}_1, \mathbb{G}_2, e, q, P, H\}$.
2. KeyGen. Randomly selects $x \in_R \mathbb{Z}_q^*$, and computes $P_{pub} = xP$. The public key is P_{pub}. The secret key is x.
3. Sign. Given a secret key x, and a message m, computes $S = \frac{1}{H(m)+x}P$. The signature is S.
4. Ver. Given a public key P_{pub}, a message m, and a signature S, verify if

$$e(H(m)P + P_{pub},\ S) = e(P,\ P).$$

The verification works because of the following equations:

$$\begin{aligned} e(H(m)P + P_{pub},\ S) &= e((H(m)+x)P,\ (H(m)+x)^{-1}P) \\ &= e(P,\ P)^{(H(m)+x)\cdot(H(m)+x)^{-1}} \\ &= e(P,\ P) \end{aligned}$$

3.2 Security Discussions

The strongest notion of security for signature schemes was defined by Goldwasser, Micali and Rivest [12] as follows:

Definition 1 (Secure signatures [12]). *A signature scheme* $\mathcal{S} = <$ *ParamGen, KeyGen, Sign, Ver* $>$ *is existentially unforgeable under an adaptive chosen message attack if it is infeasible for a forger who only knows the public key to produce a valid message-signature pair after obtaining polynomially many signatures on messages of its choice from the signer.*

Formally, for every probabilistic polynomial time forger algorithm $\mathcal{F}$ *there does not exist a non-negligible probability* ϵ *such that*

$$Adv(\mathcal{F}) = \Pr\left[\begin{array}{l} \langle pk, sk\rangle \leftarrow \langle \mathit{ParamGen}, \mathit{KeyGen}\rangle(1^l); \\ for\, i = 1, 2, \ldots, k; \\ m_i \leftarrow \mathcal{F}(pk, m_1, \sigma_1, \ldots, m_{i-1}, \sigma_{i-1}), \sigma_i \leftarrow \mathit{Sign}(sk, m_i); \\ \langle m, \sigma\rangle \leftarrow \mathcal{F}(pk, m_1, \sigma_1, \ldots, m_k, \sigma_k); \\ m \notin \{m_1, \ldots, m_k\} \text{ and } \mathit{Ver}(pk, m, \sigma) = accept \end{array}\right] \geq \epsilon.$$

Here we use the definition of [4] which takes into account the existence of an ideal hash function, and gives a concrete security analysis of digital signatures.

Definition 2 (Exact security of signatures [4]). *A forger* $\mathcal{F}$ *is said to* (t, q_H, q_S, ϵ)*-break the signature scheme* $\mathcal{S} = <$ *ParamGen, KeyGen, Sign, Ver* $>$ *via an adaptive chosen message attack if after at most* q_H *queries to the hash oracle,* q_S *signatures queries and* t *processing time, it outputs a valid forgery with probability at least* ϵ*.*

A signature scheme $\mathcal{S}$ *is* (t, q_H, q_S, ϵ)*-secure if there is no forger who* (t, q_H, q_S, ϵ)*-breaks the scheme.*

To give the security proof of the new signature scheme, we recall a problem proposed by S. Mitsunari *et. al* [18], called k-CAA (collusion attack algorithm with k traitors), and used as the security basis in Mitsunari *et. al*'s traitor tracing scheme.

Definition 3 (k-CAA). *For an integer* k*, and* $x \in_R \mathbb{Z}_q$*,* $P \in \mathbb{G}_1$*, given*

$$\{P, Q = xP, h_1, \ldots, h_k \in \mathbb{Z}_q, \frac{1}{h_1 + x}P, \ldots, \frac{1}{h_k + x}P\},$$

to compute $\frac{1}{h+x}P$ *for some* $h \notin \{h_1, \ldots, h_k\}$*.*

We say that the k-CAA is (t, ϵ)-hard if for all t-time adversaries $\mathcal{A}$, we have

$$Adv\text{k-CAA}_{\mathcal{A}} = \Pr\left[\begin{array}{l} \mathcal{A}(P, Q = xP, \frac{1}{h_1+x}P, \ldots, \frac{1}{h_k+x}P) = \frac{1}{h+x}P \\ |x \in_R \mathbb{Z}_q, P \in \mathbb{G}_1, h_1, \ldots, h_k \in \mathbb{Z}_q, h \notin \{h_1, \ldots, h_k\} \end{array}\right] < \epsilon.$$

On the security of proposed signature scheme against an adaptive chosen message attack, we have the following theorem:

Theorem 2. *If there exists a (t, q_H, q_S, ϵ)-forger $\mathcal{F}$ using adaptive chosen message attack for the proposed signature scheme, then there exists a (t', ϵ')-algorithm $\mathcal{A}$ solving q_S-CAA, where $t' = t$, $\epsilon' \geq (\frac{q_S}{q_H})^{q_S} \cdot \epsilon$* [1].

Proof. In the proposed signature scheme, before signing a message m, we need to make a query $H(m)$. Our proof is in random oracle model (the hash function is seen as a random oracle, i.e., the output of the hash function is uniformly distributed).

Suppose that a forger $\mathcal{F}$ (t, q_H, q_S, ϵ)-break the signature scheme using an adaptive chosen message attack. We will use $\mathcal{F}$ to construct an algorithm $\mathcal{A}$ to solve q_S-CAA. Suppose $\mathcal{A}$ is given a challenge: Given $P \in \mathbb{G}_1$, $Q = xP$, $h_1, h_2, \ldots, h_{q_S} \in \mathbb{Z}_q$, and $\frac{1}{h_1+x}P, \frac{1}{h_2+x}P, \ldots, \frac{1}{h_{q_S}+x}P$, to compute $\frac{1}{h+x}P$ for some $h \notin \{h_1, \ldots, h_{q_S}\}$.

Now $\mathcal{A}$ plays the role of the signer and sets $P_{pub} = Q$. $\mathcal{A}$ will answer hash oracle queries and signing queries itself. We assume that $\mathcal{F}$ never repeats a hash query or a signature query.

S1 $\mathcal{A}$ prepares q_H responses $\{w_1, w_2, \ldots, w_{q_H}\}$ of the hash oracle queries, $h_1, \ldots, h_{q_S}$ are distributed randomly in this response set.

S2 $\mathcal{F}$ makes a hash oracle query on m_i for $1 \leq i \leq q_H$. $\mathcal{A}$ sends w_i to $\mathcal{F}$ as the response of the hash oracle query on m_i.

S3 $\mathcal{F}$ makes a signature oracle query for w_i. If $w_i = h_j$, $\mathcal{A}$ returns $\frac{1}{h_j+x}P$ to $\mathcal{F}$ as the response. Otherwise the process stops and $\mathcal{A}$ has failed.

S4 Finally $\mathcal{F}$ halts and outputs a message-signature pair (m, S). Here the hash value of m is some w_l and $w_l \notin \{h_1, \ldots, h_{q_S}\}$. Since (m, S) is a valid forgery and $H(m) = w_l$, it satisfies:

$$e(H(m)P + Q,\ S) = e(P,\ P).$$

So, $S = \frac{1}{w_l+x}P$. $\mathcal{A}$ outputs (w_l, S) as a solution to $\mathcal{A}$'s challenge.

Algorithm $\mathcal{F}$ cannot distinguish between $\mathcal{A}$'s simulation and real life because the hash function behaves as a random oracle. The running time of $\mathcal{A}$ is equal to the running time of $\mathcal{F}$ $t' = t$. In step S3, the success probability of $\mathcal{A}$ is $\frac{q_S}{q_H}$, so, for all signature oracle queries, $\mathcal{A}$ will not fail with probability $\rho \geq (\frac{q_S}{q_H})^{q_S}$ (if $\mathcal{F}$ only makes $s(\leq q_S)$ signature oracle queries, the success probability of $\mathcal{A}$ is $(\frac{q_S}{q_H})^s$). Hence, after the algorithm $\mathcal{A}$ finished step S4, the success probability of $\mathcal{A}$ is: $\epsilon' \geq (\frac{q_S}{q_H})^{q_S} \cdot \epsilon$. □

In [18], S. Mitsunari *et. al* introduced another new problem, k-weak Computational Diffie-Hellman Problem (k-wCDHP), and gave the following theorem.

Definition 4 (k-wCDHP). *Given $k+1$ values $< P, yP, y^2P, \ldots, y^kP >$, compute $\frac{1}{y}P$.*

[1] To obtain a good bound for ϵ', we should assume that q_S and q_H are very closed.

Theorem 3 ([18]). *There exists a polynomial time algorithm to solve* (k-1)-*wCDHP if and only if there exists a polynomial time algorithm for k-CAA.*

So, in our signature scheme, the security against the existential forgery under an adaptive chosen message attack at least depends on k-wCDHP.

To give a more specific evaluation of the security of our signature scheme, we introduce a new problem-

Definition 5 (k+1 Exponent Problem). *Given* $k+1$ *values* $< P, yP, y^2P, \ldots, y^kP >$, *compute* $y^{k+1}P$.

We have the following theorem. The proof is given in the full version of this paper.

Theorem 4. *k-wCDHP and k+1EP are polynomial time equivalent.*

We note that $k+1EP$ and k-wCDHP are no harder than the CDHP. There exists a special case where k-wCDHP or $k+1EP$ can be easily solved. This case gives an attack on the new signature scheme. Given $P_0 = P, P_1 = yP$, $P_2 = y^2P, \ldots, P_k = y^kP$, if there are at least two same elements in them, *e.g.*, $P_i = P_j$ $(i \neq j)$, that means $y^i \bmod q \equiv y^j \bmod q$, and so, the order of y in $\mathbb{Z}_q$ is $j-i$. Then

$$y^{-1}P = P_{j-i-1} \quad or \quad y^{k+1}P = P_{k+1 \bmod (j-i)}.$$

However, because y can be regarded as a random element in $\mathbb{Z}_q^*$, we can show that the success probability of this attack is negligible.

Let $q-1 = \prod_{i=1}^{s} p_i^{e_i}$. For any $a \in \mathbb{Z}_q^*$, the order of a is a divisor of $q-1$. Given k, suppose that the number of element a in $\mathbb{Z}_q^*$ such that $ord(a) \leq k$ is given by N. Obviously, $N < k^2$ (the maximum of the number of the divisors less than k is k). Let ρ be the probability that a randomly chosen element in $\mathbb{Z}_q^*$ has order less than k, then

$$\rho = \frac{N}{q} < \frac{k^2}{q}.$$

So, if $q \approx 2^{160}$, we limit $k \leq 2^{40}$, which means the attacker has at most 2^{40} message-signature pairs. Then using the above attack, the success probability is at most

$$\frac{(2^{40})^2}{2^{160}} = 2^{-80} \approx 0.82718 \times 10^{-24}.$$

Summarizing the above discussions, we have the following result.

Corollary 1 *Assuming that k+1EP is hard, i.e., there is no polynomial time algorithm to solve k+1EP with non-negligible probability, then the proposed signature scheme is secure under the random oracle model.*

3.3 Efficiency

Short signatures are important in low-bandwidth communication environments. A number of short signature schemes, such as: Quartz [19], McEliece-based signature [10], have been proposed. BLS scheme is the shortest signature scheme known in classical cryptography (Quartz and McEliece-based signature belong to the multivariate cryptography). Our signature only consists of one element of $\mathbb{G}_1$. In practice, the size of the element in $\mathbb{G}_1$ (elliptic curve group or hyperelliptic curve Jacobians) can be reduced by a factor of 2 using compression techniques. So, like BLS signature scheme, our signature scheme is a short signature scheme.

We compare our signature scheme with the BLS scheme from computation overhead view point. We denote Pa the pairing operation, Pm the point scalar multiplication on $\mathbb{G}_1$, Ad the point addition on $\mathbb{G}_1$, Inv the inversion in $\mathbb{Z}_q$ and MTP the MapToPoint hash operation in BLS scheme. We summarize the result in Table 1(we ignore the general hash operation).

Schemes	*Setup*	*Signing*	*Verification*
Proposed	*Same*	1Inv + 1Pm	2(*or* 1)Pa + 1Pm + 1Ad
BLS scheme	*Same*	1MTP + 1Pm	2Pa + 1MTP

Table 1. Comparison of our scheme and the BLS scheme

We assume that BLS scheme and our scheme are all using the GDH group derived from the curve $E/F_{3^{163}}$ defined by the equation $y^2 = x^3 - x + 1$. The group provides 1551-bit discrete-log security. The MapToPoint hash operation requires at least one quadratic or cubic equation over $F_{3^{163}}$ to be solved. So the cost of one MapToPoint hash operation is bigger than one inversion in $\mathbb{Z}_q$. Despite a number of attempts [2, 3, 11] to reduce the complexity of pairing, still the operation is very costly. For example, according to the best result in [3], one pairing operation is about 11110 multiplications in $F_{3^{163}}$, while a point scalar multiplication of $E/F_{3^{163}}$ is a few hundred multiplications in $F_{3^{163}}$. In our scheme, $e(P, P)$ can be precomputed and published as part of the signer's public key and so there is only one pairing operation in verification. This compare to two pairing operations in BLS scheme, gives a more efficient scheme.

4 Relation to ID-based Public Key Setting

The concept of ID-based encryption and signature were first introduced by Shamir [26]. The basic idea of ID-based cryptosystems is to ues the identity information of a user functions as his public key. ID-based public key setting involves a Private Key Generator (PKG) and users. The basic operations consist of **setup** and **private key extraction.** Informally, an ID-based encryption scheme (IBE) consists of four algorithms: (1) **Setup** generates the system parameters and a *master-key,* (2) **Extract** uses the *master-key* to generate the private key corresponding to an arbitrary string ID, (3) **Encrypt** encrypts a plaintext using a public key ID and (4) **Decrypt** decrypts the ciphertexts using the corresponding private key.

Recently, bilinear pairings have been used to construct ID-based cryptosystem. As noted by Moni Naor in [6], any ID-based encryption scheme immediately gives a public key signature scheme. Therefore, there is a relationship between the short signature schemes and the ID-based public key setting from bilinear pairing, that is the signing process in the short signature scheme can be regarded as the private key extract process in the ID-based public key setting. From this viewpoint, our new signature scheme can be regarded as being derived from Sakai-Kasahara's new ID-based encryption scheme with pairing [24, 25].

5 A Ring Signature Scheme

Ring signature schemes were proposed by Rivest, Shamir, and Tauman [21]. In a ring signature, a user selects a set of possible signers including himself that is called a ring. A possible signer is anyone with a public key for a standard signature scheme. The user can then sign a message using his private key and the public keys of all of the members of the ring. The signed message then has the property that it can be verified to be signed by a user in the ring, but the identity of the actual signer will not be revealed, hence the signature provides anonymity for the signer and the anonymity cannot be revoked.

Ring signature schemes should satisfy the following properties: **Correctness, Unconditional ambiguity** or **Anonymity** and **Unforgeability.**

A number of ring signature schemes based on the pairings are proposed. Zhang et.al [28] proposed an ID-based ring signature scheme. In [7], Boneh et.al gave a ring signature scheme from BLS signature scheme. In this section, we give a new ring signature scheme based on the signature scheme in Section 3.

The system parameters are $params = \{\mathbb{G}_1, \mathbb{G}_2, e, q, P, H\}$. Let Alice be a signer with public key $P_{pubk} = s_k P$ and private key s_k, and $L = \{P_{pubi}\}$ be the set of public keys and $|L| = n$.

Ring Signing:

For message m, Alice chooses $a_i \in_R \mathbb{Z}_q$ for all $i \neq k$ and obtains

$$S_k = -\frac{1}{H(m) + s_k} \sum_{i \neq k} (a_i (H(m)P + P_{pubi})) + \frac{1}{H(m) + s_k} P.$$

Let $S_i = a_i P$, for all $i \neq k$. The ring signature is $\sigma = \langle S_1, S_2, \ldots, S_n \rangle$.

Ring Verification:

$$\prod_{i=1}^{n} e(H(m)P + P_{pubi},\ S_i) = e(P,\ P).$$

The following is a brief analysis of the scheme.

Correctness. The verification of the signature is correct because of the following.

$$\prod_{i=1}^{n} e(H(m)P + P_{pubi},\ S_i)$$

$$= \prod_{i \neq k} e(H(m)P + P_{pubi},\ a_i P) \cdot e(H(m)P + P_{pubk},\ \frac{1}{H(m)+s_k}(P - \sum_{i \neq k}(a_i(H(m)P + P_{pubi})))$$

$$= e(\sum_{i \neq k}(a_i(H(m)P + P_{pubi})),\ P) \cdot e(P,\ -\sum_{i \neq k}(a_i(H(m)P + P_{pubi}))) \cdot e(P,\ P)$$

$$= e(P,\ P)$$

Unconditional ambiguity. The scheme has unconditionally signer-ambiguity. Assume that $\sigma = \langle S_1, S_2, \ldots, S_n \rangle$ is a ring signature on the set of users L generated with private key s_k. All S_i except S_k are taken randomly from $\mathbb{G}_1$ due to $S_i = a_i P$ and $a_i \in_R \mathbb{Z}_q$. S_k is computed by these $a_i, H(m)$ and s_k. Therefore, for fixed L and m, $\langle S_1, S_2, \ldots, S_n \rangle$ has $\mid \mathbb{G}_1 \mid^{n-1}$ possible values, all of which can be chosen by the signature generation procedure with equal probability and regardless of the signer. At the same time, the distribution $\{S_1, S_2, \ldots, S_n\}$ is identical to the distribution $\{a_1 P, a_2 P, \ldots, a_n P : \sum_{i=1}^{n} a_i P = C\}$, here C is element of $\mathbb{G}_1$ depend on L and m. So, for any algorithm $\mathcal{A}$, any set of users L, and a random $k \in L$, the probability $\Pr[\mathcal{A}(\sigma) = k]$ is at most $1/ \mid L \mid$.

Unforgeability. For the unforgeability, we have the following theorem:

Theorem 5. *If there exists a (t, q_H, q_S, ϵ)-forger $\mathcal{F}$ algorithm that can produce a forgery of a ring signature on a set of users of size n, then there exists a (t', ϵ')-algorithm $\mathcal{A}$ that can solve q_S-CAA, where*

$$t' \leq t + (3 + q_S) n t_{sm} + 2(n-1) t_{add} + (n-1) t_{mu} + (n-1) t_{inv},$$

$$\epsilon' \geq (\frac{q_S}{q_H})^{q_S} \cdot \frac{1}{q_H - q_S} \cdot \epsilon.$$

Here, t_{sm} is the time of one point scalar multiplication in $\mathbb{G}_1$, t_{add} is the time of one addition in $\mathbb{G}_1$, t_{inv} is the time of one inversion in $\mathbb{Z}_q$ and t_{mu} is the time for one multiplications in $\mathbb{Z}_q$.

Proof. We adopt the security model of Rivest, Shamir and Tauman. Consider the following game played between an adversary and a challenger. The adversary is given the public keys $P_1, \ldots, P_n$ of a set of users U , and is given oracle access to H and a ring-signing oracle. The goal of the adversary is to output a valid ring signature for U of a message m subject to the condition that m has never been presented to the ring-signing oracle.

Suppose that there exists a (t, q_H, q_S, ϵ)-forger $\mathcal{F}$ algorithm that can produce a forgery of a ring signature on a set of users of size n. We will use $\mathcal{F}$ to construct an algorithm $\mathcal{A}$ to solve q_S-CAA. Suppose that $\mathcal{A}$ is given a challenge: Given $P \in \mathbb{G}_1$, $Q = xP$, $h_1, h_2, \ldots, h_{q_S} \in \mathbb{Z}_q$, and $\frac{1}{h_1+x}P, \frac{1}{h_2+x}P, \ldots, \frac{1}{h_{q_S}+x}P$, compute $\frac{1}{h+x}P$ for some $h \notin \{h_1, \ldots, h_{q_S}\}$.

- **Setup:** $\mathcal{A}$ plays the role of the real signer and picks $a_1 = 1, a_2, \ldots, a_n$ at random from $\mathbb{Z}_q$ and sets

$$P_1 = Q,\ P_2 = a_2 Q + h(a_2 - 1)P,\ \ldots,\ P_n = a_n Q + h(a_n - 1)P.$$

  Here, we assume that the number of users n is an odd number. $\mathcal{A}$ prepares q_H respondences $\{w_1, w_2, \ldots, w_{q_H}\}$ of hash oracle queries. $h_1, \ldots, h_{q_S}$ and h are distributed randomly in this respondences set.
- **Hash queries:** $\mathcal{F}$ is given the public keys $P_1, P_2, \ldots, P_n$. $\mathcal{F}$ makes a hash oracle query on m_i for $1 \le i \le q_H$. $\mathcal{A}$ sends w_i to $\mathcal{F}$ as the respondence of hash oracle query on m_i.
- **Signing queries:** $\mathcal{F}$ makes a ring signature oracle query for w_i. If $w_i = h_j$, $\mathcal{A}$ returns

$$\sigma_i = \{S_{i1},\ S_{i2},\ \ldots,\ S_{in}\}$$

  to $\mathcal{F}$ as the signing result. Here

$$\begin{aligned}
S_{i1} &= (1 - \sum_{l=2}^{n}(-1)^l(a_l - 1)^{-1}) \cdot \frac{1}{h_j + x}P = (1 - a) \cdot \frac{1}{h_j + x}P \\
S_{i2} &= (a_2 - 1)^{-1} \cdot \frac{1}{h_j + x}P \\
\ldots &= \ldots \\
S_{il} &= (-1)^l(a_l - 1)^{-1} \cdot \frac{1}{h_j + x}P \\
\ldots &= \ldots \\
S_{in} &= (a_n - 1)^{-1} \cdot \frac{1}{h_j + x}P
\end{aligned}$$

  From the construction of S_{il}, we can verify that σ_i can pass the ring verification:

$$\begin{aligned}
&\prod_{l=1}^{n} e(H(m_i)P + P_l, S_{il}) \\
&= e(h_j P + Q, \frac{1 - a}{h_j + x}P) \prod_{l=2}^{n} e(h_j P + a_l Q + h(a_l - 1)P, \frac{(-1)^t(a_l - 1)^{-1}}{h_j + x}P) \\
&= e(P, P)^{1-a} \prod_{l=2}^{n} e(h_j P + Q + (a_l - 1)Q + h(a_l - 1)P, \frac{(-1)^t(a_l - 1)^{-1}}{h_j + x}P) \\
&= e(P, P)^{1-a} \prod_{l=2}^{n} e(h_j P + Q, \frac{(-1)^t(a_l - 1)^{-1}}{h_j + x}P) e((a_l - 1)(Q + hP), \\
&\quad \frac{(-1)^t(a_l - 1)^{-1}}{h_j + x}P) \\
&= e(P, P)^{1-a} \prod_{l=2}^{n} e(P, P)^{(-1)^t(a_l - 1)^{-1}} \ (\textit{Due to } n \textit{ be an odd number}) \\
&= e(P, P)
\end{aligned}$$

Otherwise, the process stops and $\mathcal{A}$ reports failure.

- **Output:** Eventually $\mathcal{F}$ outputs a message-signature pair $(m, \sigma = \{S_1, S_2, \ldots, S_n\})$ for ring public keys $P_1, P_2, \ldots, P_n$, here the hash value of m is some w_l such that no signature query was issued for m. If $w_l \neq h$, then $\mathcal{A}$ reports failure and terminates. Otherwise,

$$\prod_{i=1}^{n} e(H(m)P + P_i, S_i) = \prod_{i=1}^{n} e(hP + a_iQ + h(a_i - 1)P, S_i) = e(P, P).$$

Hence

$$\prod_{i=1}^{n} e(a_ihP + a_iQ, S_i) = \prod_{i=1}^{n} e(hP + Q, a_iS_i) = e(hP + Q, \sum_{i=1}^{n} a_iS_i) = e(P, P).$$

Then $\mathcal{A}$ outputs the required $\frac{1}{h+x}P$ as $\sum_{i=1}^{n} a_iS_i$.

$\mathcal{A}$ will not fail with probability $(\frac{q_S}{q_H})^{q_S} \cdot \frac{1}{q_H - q_S} \cdot \epsilon$ (For all signature oracle queries, $\mathcal{A}$ will not fail with probability $\rho \geq (\frac{q_S}{q_H})^{q_S}$. In **Output,** the probability of $w_l = h$ is $\frac{1}{q_H - q_S}$).

In **Setup,** there are $n-1$ multiplications in $\mathbb{Z}_q$, $n-1$ additions and $2n$ scalar multiplications of $\mathbb{G}_1$. There are nq_S scalar multiplications of $\mathbb{G}_1$ and $n-1$ inversions over $\mathbb{Z}_q$ in $\mathcal{A}$'s signature queries, and n scalar multiplications $n-1$ additions of $\mathbb{G}_1$ in **Output.** We denote t_{sm} the time of one scalar multiplication in $\mathbb{G}_1$, t_{add} the time of one addition in $\mathbb{G}_1$, t_{inv} the time of one inversion in $\mathbb{Z}_q$ and t_{mu} the time of one multiplications in $\mathbb{Z}_q$. So $\mathcal{A}$'s running time t' is $\mathcal{F}$'s running time plus $(2n + nq_S + n)t_{sm} + 2(n-1)t_{add} + (n-1)t_{mu} + (n-1)t_{inv}$, i.e., $t' \leq t + (3 + q_S)nt_{sm} + 2(n-1)t_{add} + (n-1)t_{mu} + (n-1)t_{inv}$. □

Note that when $n = 1$, this ring signature scheme is the basic signature scheme.

6 Delegation of Right and Proxy Signatures

Assume that there are two participants, one called original signer with public key PK_o and secret key s_o, the other called proxy signer with public key PK_p and secret key s_p, they have the common system parameters: $\{\mathbb{G}_1, \mathbb{G}_2, e, q, P, H\}$. We describe the delegation in detail as follows:

- The original signer makes a warrant w. There is an explicit description of the delegation relation in the warrant w.
- The original signer computes $So_w = (s_o + H(w))^{-1}PK_p$, and sends w and So_w to proxy signer.
- The proxy signer checks if $e(H(w)P + PK_o, So_w) = e(P, PK_p)$, if it is right, then computes $S_w = s_pSo_w$.

S_w satisfies: $e(H(w)P + PK_o, S_w) = e(PK_p, PK_p)$.

No one can forge an $S_{w'}$ of a warrant w', since there are two signatures on a warrant: First, the original signer uses the signature scheme in Section 3 to sign the warrant, and then, the proxy signer will use BLS short signature scheme to sign it, these two signature schemes are secure. On the other hand, the above delegation does not require the secure channel for the delivery of the signed warrant by the original signer, *i.e.,* the original signer can publish w and So_w. More precisely, any adversary can get the original signer's signature on warrant w. Even this, the adversary cannot get the S_w of the proxy signer, because S_w should satisfy $e(H(w)P+PK_o, S_w) = e(PK_p, PK_p)$, and $e(H(w)P+PK_o, So_w) = e(P, PK_p)$. From P, So_w and PK_p to get S_w, this is CDHP.

The above delegation is a partial delegation with warrant [15]. It is can be regarded as the generation of the proxy key in proxy signature. The proxy secret key is S_w, and the proxy public key is $PK_o + PK_p$. Then the proxy signer can uses any ID-based signature schemes and ID-based blind signature schemes from pairings (takes the ID public key as $H_2(w)$) and secret key as S_w, the public key of PKG as $PK_o + PK_p$) to get proxy signature and proxy blind signature schemes.

Next, we give two applications of above delegation method in proxy signature: designing proxy signature scheme and a proxy blind signature scheme. We only describe the schemes without security analysis.

A Proxy Signature Scheme

Proxy signatures are very useful tools when one needs to delegate his/her signing capability to other party[15, 16]. Using above delegation, we give a new proxy signature scheme.

Setup: Define another cryptographic hash function: $H_1 : \{0,1\}^* \times \mathbb{G}_1 \rightarrow \mathbb{Z}_q^*$. The system parameters $params = \{\mathbb{G}_1, \mathbb{G}_2, e, q, P, H, H_1\}$, the original signer has public-secret key pair (PK_o, s_o), the proxy signer has public-secret key pair (PK_p, s_p).

Generation of the proxy key: The proxy signer receives a proxy key S_w using above delegation protocol.

Signing: For a message m, choose a random number $r \in \mathbb{Z}_q^*$, compute $U = r \cdot (H(w)P + PK_o)$. Compute $h = H_1(m||U)$ and $V = (h + r)^{-1} S_w$. The proxy signature on m is (U, V, w).

Verification: Verify that

$$e(U + H_1(m||U)(H(w)P + PK_o), V) = e(PK_p, PK_p).$$

A Proxy Blind Signature Scheme

Proxy blind signature is considered to be the combination of proxy signature and blind signature, so, it satisfies the security properties of both the blind signature and the proxy signature. Such signature is suitable for many applications where the users' privacy and proxy signature are required. Now, we give a new proxy blind signature scheme.

Setup: Same as above proxy signature scheme.

Generation of the proxy key: The proxy signer receives a proxy key S_w.

Proxy blind signature generation: Suppose that m is the message to be signed.

- The proxy signer randomly chooses a number $r \in_R \mathbb{Z}_q^*$, computes $U = r \cdot (H(w)P + PK_o)$, and sends U and the warrant w to the user.
- (Blinding) The user randomly chooses $\alpha, \beta \in_R \mathbb{Z}_q^*$ as blinding factors. He/She computes $U' = \alpha U + \alpha\beta(H(w)P + PK_o)$ and $h = \alpha^{-1}H_1(m||U') + \beta$, sends h to the signer.
- (Signing) The signer sends back V, where $V = (r + h)^{-1}S_w$.
- (Unblinding) The user computes $V' = \alpha^{-1}V$ and outputs (m, U', V').

Then (U', V', w) is the proxy blind signature of the message m.

Verification: A verifier accepts this proxy blind signature if and only if

$$e(U' + H_1(m||U')(H(w)P + PK_o), V') = e(PK_p, PK_p).$$

7 Conclusion

In this paper, we proposed a new short signature scheme that is more efficient than BLS scheme. The security of this signature scheme depends on a new problem, namely k-CAA or $k + 1EP$. It is shown that $k + 1EP$ is no harder than the CDHP. Based on this basic signature scheme, a ring signature scheme and a new method for delegation are proposed.

Acknowledgements

The authors thank Ben Lynn, Yi Mu, Xiaofeng Chen for helpful discussions about this work and anonymous referees for their helpful comments.

References

[1] P.S.L.M. Barreto and H.Y. Kim, *Fast hashing onto elliptic curves over fields of characteristic 3,* Cryptology ePrint Archive, Report 2001/098.

[2] P.S.L.M. Barreto, H.Y. Kim, B.Lynn, and M.Scott, *Efficient algorithms for pairing-based cryptosystems,* Crypto 2002, LNCS 2442, pp.354-368, Springer-Verlag, 2002.

[3] P.S.L.M. Barreto, B.Lynn, and M.Scott, *On the selection of pairing-friendly groups,* SAC 2003. LNCS, Springer- Verlag, 2003.

[4] M. Bellare and P. Rogaway, *The exact security of digital signatures - How to sign with RSA and Rabin.* Eurocrypt'96, LNCS 1070, Springer- Verlag, 1996, pp. 399-416.

[5] A. Boldyreva, *Efficient threshold signature, multisignature and blind signature schemes based on the Gap-Diffie-Hellman -group signature scheme,* PKC 2003, LNCS 2139, pp.31-46, Springer-Verlag, 2003.

[6] D. Boneh and M. Franklin, *Identity-based encryption from the Weil pairing,* Crypto 2001, LNCS 2139, pp.213-229, Springer-Verlag, 2001.

[7] D. Boneh, C. Gentry, B. Lynn and H. Shacham, *Aggregate and verifiably encrypted signatures from bilinear maps,* Eurocrypt 2003, LNCS 2656, pp.272-293, Springer-Verlag, 2003.
[8] D. Boneh, B. Lynn, and H. Shacham, *Short signatures from the Weil pairing,* Asiacrypt 2001, LNCS 2248, pp.514-532, Springer-Verlag, 2001.
[9] J.C. Cha and J.H. Cheon, *An identity-based signature from gap Diffie-Hellman groups,* PKC 2003, LNCS 2139, pp. 18-30, Springer-Verlag, 2003.
[10] N.T. Courtois, M. Finiasz and N. Sendrier, *How to achieve a McEliece-based Digital Signature Schem,* Asiacrypt 2001, LNCS 2248, pp.157-174, Springer-Verlag, 2001.
[11] S. D. Galbraith, K. Harrison, and D. Soldera, *Implementing the Tate pairing,* ANTS 2002, LNCS 2369, pp.324-337, Springer-Verlag, 2002.
[12] S. Goldwasser, S. Micali and R. Rivest, *A digital signature scheme secure against adaptive chosen-message attacks,* SIAM Journal of computing, 17(2), pp. 281-308, April 1988.
[13] F. Hess, *Efficient identity based signature schemes based on pairings,* SAC 2002, LNCS 2595, pp.310-324, Springer-Verlag, 2002.
[14] A. Joux, *A one round protocol for tripartite Diffie-Hellman,* ANTS IV, LNCS 1838, pp.385-394, Springer-Verlag, 2000.
[15] S. Kim, S. Park, and D. Won, *Proxy signatures, revisited,* ICICS'97, LNCS 1334, Springer-Verlag, pp. 223-232, 1997.
[16] M. Mambo, K. Usuda, and E. Okamoto, *Proxy signature: Delegation of the power to sign messages,* In IEICE Trans. Fundamentals, Vol. E79-A, No. 9, Sep., pp. 1338-1353, 1996.
[17] U. Maurer, *Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms,* Crypto 94, LNCS 839, pp.271-281, Springer-Verlag, 1994.
[18] S. Mitsunari, R. Sakai and M. Kasahara, *A new traitor tracing,* IEICE Trans. Vol. E85-A, No.2, pp.481-484, 2002.
[19] J. Patarin, N. Courtois and L. Goubin, *QUARTZ, 128-bit long digital signatures,* CT-RSA 2001, LNCS 2020, pp. 282-297, Springer-Verlag, 2001.
[20] K.G. Paterson, *ID-based signatures from pairings on elliptic curves,* Electron. Lett., Vol.38, No.18, pp.1025-1026, 2002.
[21] R.L. Rivest, A. Shamir and Y. Tauman, *How to leak a secret,* Asiacrypt 2001, LNCS 2248, pp.552-565, Springer-Verlag, 2001.
[22] A.R. Sadeghi and M. *Steiner,Assumptions related to discrete logarithms: why subtleties make a real difference,* Eurocrypt 2001, LNCS 2045, pp.243-260, Springer-Verlag, 2001.
[23] R. Sakai, K. Ohgishi and M. Kasahara, *Cryptosystems based on pairing,* SCIS 2000-C20, Jan. 2000. Okinawa, Japan.
[24] R. Sakai and M. Kasahara, *Cryptosystems based on pairing over elliptic curve,* SCIS 2003, 8C-1, Jan. 2003. Japan.
[25] R. Sakai and M. Kasahara, *ID based Cryptosystems with pairing on elliptic curve,* Cryptology ePrint Archive, Report 2003/054.
[26] A. Shamir, *Identity-based Cryptosystems and signature schemes,* Advances in Cryptology-Crypto 84, LNCS 196, pp.47-53, Springer-Verlag, 1984.
[27] N.P. Smart, *An identity based authenticated key agreement protocol based on the Weil pairing,* Electron. Lett., Vol.38, No.13, pp.630-632, 2002.
[28] F. Zhang and K. Kim, *ID-based blind signature and ring signature from pairings,* Asiacrpt2002, LNCS 2501, pp. 533-547, Springer-Verlag, 2002.

An RSA Family of Trap-Door Permutations with a Common Domain and Its Applications

Ryotaro Hayashi[1], Tatsuaki Okamoto[2], and Keisuke Tanaka[1]*

[1] Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, 2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552, Japan
{hayashi9, keisuke}@is.titech.ac.jp
[2] NTT Labs, 1-1 Hikarino-oka, Yokosuka-shi, Kanagawa 239-0847, Japan
okamoto@isl.ntt.co.jp

Abstract. Bellare, Boldyreva, Desai, and Pointcheval [1] recently proposed a new security requirement of the encryption schemes called "key-privacy." It asks that the encryption provide (in addition to privacy of the data being encrypted) privacy of the key under which the encryption was performed. Incidentally, Rivest, Shamir, and Tauman [2] recently proposed the notion of ring signature, which allows a member of an ad hoc collection of users S to prove that a message is authenticated by a member of S without revealing which member actually produced the signature.

We are concerned with an underlying primitive element common to the key-privacy encryption and the ring signature schemes, that is, families of trap-door permutations with a common domain. For a standard RSA family of trap-door permutations, even if all of the functions in a family use RSA moduli of the same size (the same number of bits), it will have domains with different sizes. In this paper, we construct an RSA family of trap-door permutations with a common domain, and propose the applications of our construction to the key-privacy encryption and ring signature schemes, which have some advantage to the previous schemes.

Keywords: RSA, trap-door permutations, key-privacy, anonymity, encryption, ring signature

1 Introduction

Bellare, Boldyreva, Desai, and Pointcheval [1] recently proposed a new security requirement of the encryption schemes called "key-privacy." It asks that the encryption provide (in addition to privacy of the data being encrypted) privacy of the key under which the encryption was performed. The standard RSA encryption does not provide key-privacy. Since even if two public keys N_0 and N_1 $(N_0 < N_1)$ are the same bits, $N_1 - N_0$ may be large. In [1], they provided the

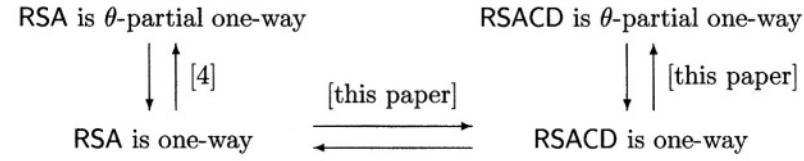
* Supported in part by NTT Information Sharing Platform Laboratories and Grant-in-Aid for Scientific Research, Ministry of Education, Culture, Sports, Science, and Technology, 14780190.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 291–304, 2004.

key-privacy encryption scheme, RSA-RAEP, which is a variant of RSA-OAEP (Bellare and Rogaway [3], Fujisaki, Okamoto, Pointcheval, and Stern [4]), and solved this problem by repeating the evaluation of the RSA-OAEP permutation $f(x, r)$ with plaintext x and random r, each time using different r until the value is in the safe range (See section 3.2.). For deriving a value in the safe range, the number of the repetition would be very large (the value of the security parameter).

Incidentally, Rivest, Shamir, and Tauman [2] recently proposed the notion of ring signature, which allows a member of an ad hoc collection of users S to prove that a message is authenticated by a member of S without revealing which member actually produced the signature. Unlike group signature, ring signature has no group managers, no setup procedures, no revocation procedures, and no coordination. The signer does not need the knowledge, consent, or assistance of the other ring members to put them in the ring. All the signer needs is knowledge of their regular public keys. They also proposed the efficient schemes based on RSA and Rabin. In their RSA-based scheme, the trap-door RSA permutations of the various ring members will have domains of different sizes. This makes it awkward to combine the individual signatures, so one should construct some trap-door one-way permutation which has a common domain for each user. Intuitively, in the ring signature scheme, Rivest, Shamir, and Tauman solved this by encoding the message to an N_i-ary representation and applying a standard permutation f to the low-order digits (See section 4.2.). As mentioned in [2], for deriving a secure permutation g with a common domain, the domain of g would be 160 bits larger than that of f.

In this paper, we will take a different approach. We use neither the repetition of evaluation of a permutation nor an N_i-ary representation. We are concerned with an underlying primitive element common to the key-privacy encryption and the ring signature schemes, that is, families of trap-door permutations with a common domain. For a standard RSA family of trap-door permutations denoted by RSA, even if all of the functions in a family use RSA moduli of the same size (the same number of bits), it will have domains with different sizes. We construct an RSA family of trap-door permutations with a common domain denoted by RSACD, and prove that the θ-partial one-wayness of RSACD is equivalent to the one-wayness of RSACD for $\theta > 0.5$, and that the one-wayness of RSACD is equivalent to the one-wayness of RSA. Fujisaki, Okamoto, Pointcheval, and Stern [4] showed that the θ-partial one-wayness of RSA is equivalent to the one-wayness of RSA for $\theta > 0.5$. Thus, the following relations are satisfied for $\theta > 0.5$.



We then propose the application to the key-privacy encryption scheme. Our proposed scheme is more efficient than the previous scheme with respect to the

number of exponentiations for encryption in the worst case. When we use the RSA moduli which is uniformly distributed in $(2^{k-1}, 2^k)$, the expected number of our scheme is the same as that of RSA-RAEP. In our scheme, the number of exponentiations for encryption is at most two, while in RSA-RAEP, the upper bound of this number is k_1 ($\gg 2$, security parameter).

We also propose the application to the ring signature scheme. We consider the case that the members of the same group use the RSA moduli of the same length. In our scheme, the domain of trap-door one-way permutation to sign and verify a ring signature is $\{0,1\}^k$, while that of the previous scheme is $\{0,1\}^{k+160}$, where k is the length of the RSA moduli. Thus, we can reduce the size of signature in this situation.

The organization of this paper is as follows. In Section 2, after reviewing the definitions of families of functions and the standard RSA family, we propose the RSA family of trap-door permutations with a common domain. We also prove that the θ-partial one-wayness of RSACD is equivalent to the one-wayness of RSACD for $\theta > 0.5$, and that the one-wayness of RSACD is equivalent to the one-wayness of RSA. In Section 3, we propose the application of our new family to the key-privacy encryption scheme. In Section 4, we propose the application of our new family to the ring signature scheme. We conclude in Section 5.

2 An RSA Family of Trap-Door Permutations with a Common Domain

2.1 Preliminaries

In this section, we briefly review the definitions of families of functions, and the standard RSA family of trap-door permutations denoted by RSA.

Definition 1 (families of functions [1]). *A family of functions $F = (K, S, E)$ is specified by three algorithms.*

- *The randomized key-generation algorithm K takes as input a security parameter $k \in \mathbb{N}$ and returns a pair (pk, sk) where pk is a public key and sk is an associated secret key. (In cases where the family is not trap-door, the secret key is simply the empty string.)*
- *The randomized sampling algorithm S takes input pk and returns a random point in a set that we call the domain of pk and denote by $\mathrm{Dom}_F(pk)$.*
- *The deterministic evaluation algorithm E takes input pk and a point $x \in \mathrm{Dom}_F(pk)$ and returns an output we denote by $E_{pk}(x)$. We let $\mathrm{Rng}_F(pk) = \{E_{pk}(x) \mid x \in \mathrm{Dom}_F(pk)\}$ denote the range of the function $E_{pk}(\cdot)$.*

Definition 2 (families of trap-door permutations [1]). *We say that F is a family of trap-door functions if there exists a deterministic inversion algorithm I that takes input sk and a point $y \in \mathrm{Rng}_F(pk)$ and returns a point $x \in \mathrm{Dom}_F(pk)$ such that $E_{pk}(x) = y$. We say that F is a family of trap-door permutations if F is a family of trap-door functions, $\mathrm{Dom}_F(pk) = \mathrm{Rng}_F(pk)$, and E_{pk} is a permutation on this set.*

We describe the definition of θ-partial one-way.

Definition 3 (θ-partial one-way [1]). *Let $F = (K, S, E)$ be a family of functions. Let $b \in \{0,1\}$ and $k \in \mathbb{N}$ be a security parameter. Let $0 < \theta \leq 1$ be a constant. Let A be an adversary. Now, we consider the following experiments:*

$$
\begin{array}{l}
\texttt{Experiment } \mathbf{Exp}_{F,A}^{\theta-\mathrm{pow}-\mathrm{fnc}}(k) \\
\quad (pk, sk) \stackrel{R}{\leftarrow} K(k) \\
\quad x_1||x_2 \stackrel{R}{\leftarrow} \mathrm{Dom}_F(pk) \texttt{ where } |x_1| = \lceil \theta \cdot |(x_1||x_2)| \rceil \\
\quad y \leftarrow E_{pk}(x_1||x_2) \\
\quad x_1' \leftarrow A(pk, y) \texttt{ where } |x_1'| = |x_1| \\
\quad \texttt{for any } x_2' \texttt{ if } E_{pk}(x_1'||x_2') = y \texttt{ then return } 1 \\
\quad \texttt{else return } 0
\end{array}
$$

We define the advantages of the adversary via

$$\mathbf{Adv}_{F,A}^{\theta-\mathrm{pow}-\mathrm{fnc}}(k) = \Pr[\mathbf{Exp}_{F,A}^{\theta-\mathrm{pow}-\mathrm{fnc}}(k) = 1]$$

where the probability is taken over $(pk, sk) \stackrel{R}{\leftarrow} K(k)$, $x_1||x_2 \stackrel{R}{\leftarrow} \mathrm{Dom}_F(pk)$, and the coin tosses of A. We say that the family F is θ-partial one-way if the function $\mathbf{Adv}_{F,A}^{\theta-\mathrm{pow}-\mathrm{fnc}}(\cdot)$ is negligible for any adversary A whose time complexity is polynomial in k. In particular, we say that the family F is one-way when F is 1-partial one-way.

We now describe the standard RSA family of trap-door permutations.

Definition 4 (the standard RSA family of trap-door permutations [1]). *The specifications of the standard RSA family of trap-door permutations $\mathsf{RSA} = (K, S, E)$ are as follows. The key generation algorithm takes as input a security parameter k and picks random, distinct primes p, q in the range $2^{k/2-1} < p, q < 2^{k/2}$. (If k is odd, increment it by 1 before picking the primes.) It sets $N = pq$. (i.e. $2^{k-2} < N < 2^k$.) It picks $e, d \in \mathbb{Z}_{\phi(N)}^*$ such that $ed = 1 \pmod{\phi(N)}$ where $\phi(N) = (p-1)(q-1)$. The public key is N, e, k and the secret key is N, d, k. The sets $\mathrm{Dom}_{\mathsf{RSA}}(N, e, k)$ and $\mathrm{Rng}_{\mathsf{RSA}}(N, e, k)$ are both equal to $\mathbb{Z}_N^*$. The evaluation algorithm $E_{N,e,k}(x) = x^e \bmod N$ and the inversion algorithm $I_{N,d,k}(y) = y^d \bmod N$. The sampling algorithm returns a random point in $\mathbb{Z}_N^*$.*

Fujisaki, Okamoto, Pointcheval, and Stern [4] showed that the θ-partial one-wayness of RSA is equivalent to the one-wayness of RSA for $\theta > 0.5$.

2.2 The Construction of RSACD

In this section, we propose the RSA family of trap-door permutations with a common domain denoted by RSACD.

Definition 5 (the RSA family of trap-door permutations with a common domain). *The specifications of the RSA family of trap-door permutations*

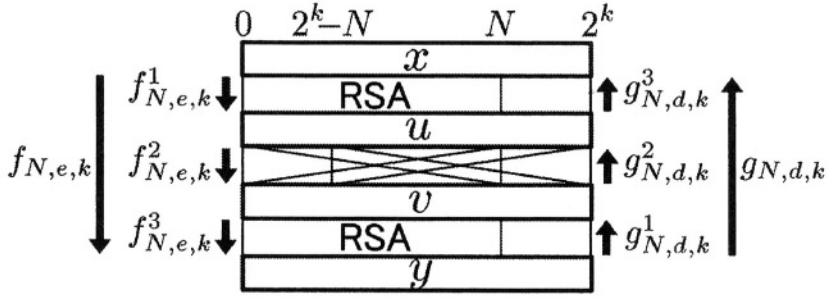


Fig. 1. Function $f_{N,e,k}$ and $g_{N,d,k}$

with a common domain RSACD= (K, S, E) *are as follows. The key generation algorithm is almost the same as that for* RSA *family. The difference is picking two distinct primes* p, q *such that* $2^{k/2-1} < p, q < 2^{k/2}$ *and* $2^{k-1} < pq < 2^k$. *The sets* $\mathrm{Dom}_{\mathsf{RSACD}}(N, e, k)$ *and* $\mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)$ *are both* $\{x \mid x \in [0, 2^k) \wedge x \bmod N \in \mathbb{Z}^*_N\}$. *The sampling algorithm returns a random point in* $\mathrm{Dom}_{\mathsf{RSACD}}(N, e, k)$. *The evaluation algorithm* $E_{N,e,k}(x) = f_{N,e,k}(x)$ *and the inversion algorithm* $I_{N,d,k}(y) = g_{N,d,k}(y)$ *are as follows (See Figure 1.).*

```
Function f_{N,e,k}(x)
  u ← f^1_{N,e,k}(x); v ← f^2_{N,e,k}(u)
  y ← f^3_{N,e,k}(v)
  return y

Function f^1_{N,e,k}(x)
  if (x < N) u ← x^e mod N
  else u ← x
  return u

Function f^2_{N,e,k}(u)
  if (u < 2^k − N) v ← u + N
  elseif (2^k − N ≤ u < N) v ← u
  else v ← u − N
  return v

Function f^3_{N,e,k}(v)
  if (v < N) y ← v^e mod N
  else y ← v
  return y
```

```
Function g_{N,d,k}(y)
  v ← g^1_{N,d,k}(y); u ← g^2_{N,d,k}(v)
  x ← g^3_{N,d,k}(u)
  return x

Function g^1_{N,d,k}(y)
  if (y < N) v ← y^d mod N
  else v ← y
  return v

Function g^2_{N,d,k}(v)
  if (v < 2^k − N) u ← v + N
  elseif (2^k − N ≤ v < N) u ← v
  else u ← v − N
  return u

Function g^3_{N,d,k}(u)
  if (u < N) x ← u^d mod N
  else x ← u
  return x
```

The choice of N from $(2^{k-1}, 2^k)$ ensures that all elements in $\mathrm{Dom}_{\mathsf{RSACD}}(N, e, k)$ are permuted by the RSA function at least once.

2.3 Properties of RSACD

In this section, we prove that the θ-partial one-wayness of RSACD is equivalent to the one-wayness of RSACD for $\theta > 0.5$, and that the one-wayness of RSACD is equivalent to the one-wayness of RSA.

Theorem 1. *Let A be an algorithm that outputs the $k - k_0$ most significant bits of the pre-image of its input $y \in \mathsf{Rng}_{\mathsf{RSACD}}(N, e, k)$ for $2^{k-1} < N < 2^k$ with $k > 2k_0$ (i.e. A is a $((k-k_0)/k)$-partial inverting algorithm for* RSACD *with $k > 2k_0$), with success probability $\epsilon = \mathbf{Adv}^{\theta-\mathrm{pow}-\mathrm{fnc}}_{\mathsf{RSACD},A}(k)$ where $\theta = (k - k_0)/k > 0.5$, within time bound t. There exists an algorithm B that outputs a pre-image of y (i.e. B is an inverting algorithm for* RSACD*) with success probability $\epsilon' = \mathbf{Adv}^{1-\mathrm{pow}-\mathrm{fnc}}_{\mathsf{RSACD},B}(k)$, within time bound t' where*

$$\epsilon' \geq \frac{\epsilon^2}{16} \cdot (1 - 2^{2k_0-k+7}), \quad t' \leq 2t + O(k^3).$$

To prove this theorem, we use the following lemma proved in [4].

Lemma 1 ([4]). *Consider an equation $\alpha t + u = c \pmod N$ which has solutions t and u smaller than 2^{k_0}. For all values of α, except a fraction $2^{2k_0+6}/N$ of them, (t, u) is unique and can be computed in time $O((\log N)^3)$. (We say "α is a good value" when we can solve the above equation.)*

Proof (Theorem 1). We construct the algorithm B to compute a pre-image of $y \in \mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)$, then we analyze this algorithm and evaluate the success probability and the running time of B.

Algorithm $B((N, e, k), y)$

$\alpha \xleftarrow{R} \mathbb{Z}_N$; $\mathrm{pow} \xleftarrow{R} \{1, 2\}$; $c \xleftarrow{R} \{0, 1\}$
$y'_{temp} \leftarrow y \cdot \alpha^{e^{\mathrm{pow}}} \bmod N$
if $(c = 0)$ $y' \leftarrow y'_{temp}$
elseif $(0 \leq y'_{temp} < 2^k - N)$ $y' \leftarrow y'_{temp} + N$
else return fail
} [step 1] set α, pow, y'

$z \leftarrow A(y)$; $z' \leftarrow A(y')$ } [step 2] run A

find (r, s) **s.t.** $\alpha r - s = (z' - z\alpha) \cdot 2^{k_0} \pmod N$
$x \leftarrow z \cdot 2^{k_0} + r$
} [step 3] compute $g_{N,d,k}(y)$

return x

Analysis

For $y \in \mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)$ and $x = g_{N,d,k}(y)$, (x, y) satisfies one of the following equations.

$$(1) \quad y = x^e \pmod N \qquad (2) \quad y = x^{e^2} \pmod N$$

We say $type(y) = 1$ (respectively $type(y) = 2$) if (x, y) satisfies equation 1 (resp. equation 2).

After step 1, if B does not output `fail`, then y' is uniformly distributed over $\mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)$, and for y' and $x' = g_{N,d,k}(y')$, (x', y') satisfies one of the following equations.

$$(1') \quad y' = (x')^e \pmod N \qquad (2') \quad y' = (x')^{e^2} \pmod N$$

We say $type(y') = 1$ (respectively $type(y') = 2$) if (x', y') satisfies equation $1'$ (resp. equation $2'$).

After step 2, if A outputs correctly, namely, z is the $k - k_0$ most significant bits of x and z' is the $k - k_0$ most significant bits of x', then $x = z \cdot 2^{k_0} + r$ and $x' = z' \cdot 2^{k_0} + s$ for some (r, s) where $0 \le r, s < 2^{k_0}$. Furthermore, if $type(y) = type(y') = \text{pow}$, then $y = x^{e^{\text{pow}}} \pmod N$ and $y' = (x')^{e^{\text{pow}}} \pmod N$. Since $y' = y \cdot \alpha^{e^{\text{pow}}} \pmod N$ and $\gcd(e^{\text{pow}}, N) = 1$, we have $x' = \alpha x \pmod N$. Thus,

$$z' \cdot 2^{k_0} + s = \alpha \cdot (z \cdot 2^{k_0} + r) \pmod N$$
$$\alpha r - s = (z' - z\alpha) \cdot 2^{k_0} \pmod N$$

where $0 \le r, s < 2^{k_0}$. If α is a good value, algorithm B can solve this equation in step 3 (Lemma 1), and outputs $x = z \cdot 2^{k_0} + r$.

Now, we analyze the success probability. We define the following events:

- Fail : B outputs `fail` in step 1,
- GV : α is a good value,
- Type1 : $type(y) = type(y') = 1$,
- Type2 : $type(y) = type(y') = 2$,
- SucA: $A(y)$ and $A(y')$ are correct.

We have $\epsilon = \Pr[A(y) \text{ is correct} \wedge type(y) = 1] + \Pr[A(y) \text{ is correct} \wedge type(y) = 2]$ where y is uniformly distributed over $\text{Rng}_{\text{RSACD}}(N, e, k)$. Thus,

$$\Pr[A(y) \text{ is correct} \wedge type(y) = 1] > \frac{\epsilon}{2} \text{ or } \Pr[A(y) \text{ is correct} \wedge type(y) = 2] > \frac{\epsilon}{2}.$$

If B does not output `fail` in step 1, then y' is uniformly distributed over $\text{Rng}_{\text{RSACD}}(N, e, k)$. Therefore,

$$\Pr[\mathsf{SucA} \wedge \mathsf{Type1} | \neg \mathsf{Fail}] > \frac{\epsilon^2}{4} \text{ or } \Pr[\mathsf{SucA} \wedge \mathsf{Type2} | \neg \mathsf{Fail}] > \frac{\epsilon^2}{4}.$$

If $A(y)$ and $A(y')$ are correct, $type(y) = type(y') = \text{pow}$, and α is a good value, then B outputs correctly. Since $\Pr[\neg \mathsf{Fail}] > \Pr[c = 1] = 1/2$, $\Pr[\text{pow} = 1] = \Pr[\text{pow} = 2] = 1/2$, and $\Pr[\mathsf{GV}] > 1 - 2^{2k_0 - 6}/N > 1 - 2^{2k_0 - k + 7}$, we have

$$\begin{aligned}
\epsilon' &\ge \Pr[\mathsf{SucA} \wedge type(y) = type(y') = \text{pow} \wedge \alpha \text{ is a good value}] \\
&\ge \Pr[\mathsf{GV}] \times \Pr[\neg \mathsf{Fail}] \times \Pr[\mathsf{SucA} \wedge type(y) = type(y') = \text{pow} | \neg \mathsf{Fail}] \\
&\ge \frac{1}{2} \cdot (1 - 2^{2k_0 - k + 7}) \times (\Pr[\mathsf{SucA} \wedge \mathsf{Type1} \wedge \text{pow} = 1 | \neg \mathsf{Fail}] \\
&\qquad + \Pr[\mathsf{SucA} \wedge \mathsf{Type2} \wedge \text{pow} = 2 | \neg \mathsf{Fail}]) \\
&= \frac{1}{2} \cdot (1 - 2^{2k_0 - k + 7}) \times (\Pr[\text{pow} = 1] \times \Pr[\mathsf{SucA} \wedge \mathsf{Type1} | \neg \mathsf{Fail}] \\
&\qquad + \Pr[\text{pow} = 2] \times \Pr[\mathsf{SucA} \wedge \mathsf{Type2} | \neg \mathsf{Fail}]) \\
&> \frac{\epsilon^2}{16} \cdot (1 - 2^{2k_0 - k + 7}).
\end{aligned}$$

We estimate the running time of B. B runs A twice. B can solve $\alpha r - s = (z' - z\alpha) \cdot 2^{k_0} \pmod N$ in time $O(k^3)$. Therefore, $t' \le 2t + O(k^3)$. □

Theorem 2. *If* RSA *is one-way, then* RSACD *is one-way.*

Proof. We prove that if there exists a polynomial-time inverting algorithm A for RSACD with non-negligible probability $\epsilon = \mathbf{Adv}^{1-\mathrm{pow}-\mathrm{fnc}}_{\mathsf{RSACD},A}(k)$, then there exists a polynomial-time inverting algorithm D for RSA with non-negligible probability $\epsilon' = \mathbf{Adv}^{1-\mathrm{pow}-\mathrm{fnc}}_{\mathsf{RSA},D}(k)$. We specify the algorithm D to compute a pre-image of $Y \in \mathrm{Rng}_{\mathsf{RSA}}(N,e,k)$.

Algorithm $D((N,e,k),Y)$
 if $(2^{k-2} < N \leq 2^{k-1})$ **return fail**
 else
  $c \xleftarrow{R} \{0,1\}$
  if $(c{=}0)$ $y \leftarrow Y$; $x \leftarrow A((N,e,k),y)$; $u \leftarrow f^1_{N,e,k}(x)$; $v \leftarrow f^2_{N,e,k}(u)$; $X \leftarrow v$
  else $u \leftarrow Y$; $v \leftarrow f^2_{N,e,k}(u)$; $y \leftarrow f^3_{N,e,k}(v)$; $x \leftarrow A((N,e,k),y)$; $X \leftarrow x$
 return X

Now, we analyze the advantage of D. Let Fail be the event that D outputs `fail` and $\lambda = \Pr[\neg\mathsf{Fail}]$. It is clear that λ is non-negligible. In the following, $\Pr_1[\cdot]$ denotes $\Pr[\cdot|\neg\mathsf{Fail}]$. If D does not output `fail` and A outputs correctly, then D outputs correctly (See Figure 1). Therefore,

$$\begin{aligned}\epsilon' &> \Pr[\neg\mathsf{Fail}] \cdot (\Pr_1[c=0 \ \wedge \ A((N,e,k),Y) \text{ is correct}] \\ &\qquad + \Pr_1[c=1 \ \wedge \ A((N,e,k),Z) \text{ is correct}]) \\ &\geq \frac{\lambda}{2} \cdot (\Pr_1[A((N,e,k),Y) \text{ is correct}] \\ &\qquad + \Pr_1[A((N,e,k),Z) \text{ is correct } \wedge \ N \leq Z < 2^k]).\end{aligned}$$

where $Z = f^3_{N,e,k}(f^2_{N,e,k}(Y))$. We have

$$\begin{aligned}\Pr_1[A((N,e,k),Y) \text{ is correct}] &= \Pr_1[A((N,e,k),y) \text{ is correct} \,|\, 0 \leq y < N] \\ &> \Pr_1[A((N,e,k),y) \text{ is correct } \wedge \ 0 \leq y < N].\end{aligned}$$

Furthermore, we have $\Pr_1[N \leq Z < 2^k] > \Pr_1[N \leq y < 2^k]$ where Y is uniformly distributed over $\mathbb{Z}^*_N$ and y is uniformly distributed over $\mathrm{Rng}_{\mathsf{RSACD}}(N,e,k)$, since $\Pr_1[N \leq Z < 2^k] = \Pr_1[0 \leq Y < 2^k - N]$ and $|\mathbb{Z}^*_N| < |\mathrm{Rng}_{\mathsf{RSACD}}(N,e,k)|$. Since $\Pr_1[A((N,e,k),Z) \text{ is correct} \,|\, N \leq Z < 2^k] = \Pr_1[A((N,e,k),y) \text{ is correct} \,|\, N \leq y < 2^k]$, we have

$$\begin{aligned}&\Pr_1[A((N,e,k),Z) \text{ is correct } \wedge \ N \leq Z < 2^k] \\ &\qquad > \Pr_1[A((N,e,k),y) \text{ is correct } \wedge \ N \leq y < 2^k].\end{aligned}$$

Therefore,

$$\begin{aligned}\epsilon' &> \frac{\lambda}{2} \cdot (\Pr_1[A((N,e,k),y) \text{ is correct } \wedge \ 0 \leq y < N] \\ &\qquad + \Pr_1[A((N,e,k),y) \text{ is correct } \wedge \ N \leq y < 2^k]) \\ &= \frac{\lambda}{2} \cdot \Pr_1[A((N,e,k),y) \text{ is correct}] = \frac{\lambda}{2} \cdot \epsilon\end{aligned}$$

□

It is clear that if RSACD is one-way then RSA is one-way. Thus, the one-wayness of RSACD is equivalent to the one-wayness of RSA.

3 Application to Key-Privacy Encryption

3.1 Definitions of Key-Privacy

The classical security requirements of an encryption scheme, for example indistinguishability or non-malleability under the chosen-ciphertext attack, provide privacy of the encryption data. In [1], Bellare, Boldyreva, Desai, and Pointcheval proposed a new security requirement of encryption schemes called "key-privacy." It asks that the encryption provide (in addition to privacy of the data being encrypted) privacy of the key under which the encryption was performed.

In a heterogeneous public-key environment, encryption will probably fail to be anonymous for trivial reasons. For example, different users might be using different cryptosystems, or, if the same cryptosystem, have keys of different lengths. In [1], a public-key encryption scheme with common-key generation is described as follows.

Definition 6. *A public-key encryption scheme with common-key generation* $\mathcal{PE} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ *consists of four algorithms.*

- *The common-key generation algorithm* $\mathcal{G}$ *takes as input some security parameter* k *and returns some common key* I.
- *The key generation algorithm* $\mathcal{K}$ *is a randomized algorithm that takes as input the common key* I *and returns a pair* (pk, sk) *of keys, the public key and a matching secret key.*
- *The encryption algorithm* $\mathcal{E}$ *is a randomized algorithm that takes the public key* pk *and a plaintext* x *to return a ciphertext* y.
- *The decryption algorithm* $\mathcal{D}$ *is a deterministic algorithm that takes the secret key* sk *and a ciphertext* y *to return the corresponding plaintext* x *or a special symbol* $\perp$ *to indicate that the ciphertext was invalid.*

In [1], they formalized the property of "key-privacy." This can be considered under either the chosen-plaintext attack or the chosen-ciphertext attack, yielding two notions of security, IK-CPA and IK-CCA. (IK means "indistinguishability of keys".)

Definition 7 (IK-CPA, IK-CCA[1]). *Let* $\mathcal{PE} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ *be an encryption scheme. Let* $b \in \{0, 1\}$ *and* $k \in \mathbb{N}$. *Let* A_{cpa}, A_{cca} *be adversaries that run in two stages and where* A_{cca} *has access to the oracles* $\mathcal{D}_{sk_0}(\cdot)$ *and* $\mathcal{D}_{sk_1}(\cdot)$. *Note that* si *is the state information. It contains* pk_0, pk_1, *and so on. Now, we consider the following experiments:*

$\mathtt{Experiment}\ \mathbf{Exp}^{\mathrm{ik-cpa}-b}_{\mathcal{PE},A_{\mathrm{cpa}}}(k)$
$I \xleftarrow{R} \mathcal{G}(k)$
$(pk_0, sk_0) \xleftarrow{R} \mathcal{K}(I);\ (pk_1, sk_1) \xleftarrow{R} \mathcal{K}(I)$
$(x, \mathsf{si}) \leftarrow A_{\mathrm{cpa}}(\mathsf{find}, pk_0, pk_1)$
$y \leftarrow \mathcal{E}_{pk_b}(x)$
$d \leftarrow A_{\mathrm{cpa}}(\mathsf{guess}, y, \mathsf{si})$
$\mathtt{return}\ d$

$\mathtt{Experiment}\ \mathbf{Exp}^{\mathrm{ik-cca}-b}_{\mathcal{PE},A_{\mathrm{cca}}}(k)$
$I \xleftarrow{R} \mathcal{G}(k)$
$(pk_0, sk_0) \xleftarrow{R} \mathcal{K}(I);\ (pk_1, sk_1) \xleftarrow{R} \mathcal{K}(I)$
$(x, \mathsf{si}) \leftarrow A_{\mathrm{cca}}^{\mathcal{D}_{sk_0}(\cdot), \mathcal{D}_{sk_1}(\cdot)}(\mathsf{find}, pk_0, pk_1)$
$y \leftarrow \mathcal{E}_{pk_b}(x)$
$d \leftarrow A_{\mathrm{cca}}^{\mathcal{D}_{sk_0}(\cdot), \mathcal{D}_{sk_1}(\cdot)}(\mathsf{guess}, y, \mathsf{si})$
$\mathtt{return}\ d$

Above it is mandated that A_{cca} never queries $\mathcal{D}_{sk_0}(\cdot)$ and $\mathcal{D}_{sk_1}(\cdot)$ on the challenge ciphertext y. For atk $\in \{\mathrm{cpa}, \mathrm{cca}\}$ *we define the advantages via*

$$\mathbf{Adv}^{\mathrm{ik-atk}}_{\mathcal{PE},A_{\mathrm{atk}}}(k) = \left|\Pr[\mathbf{Exp}^{\mathrm{ik-atk-1}}_{\mathcal{PE},A_{\mathrm{atk}}}(k) = 1] - \Pr[\mathbf{Exp}^{\mathrm{ik-atk-0}}_{\mathcal{PE},A_{\mathrm{atk}}}(k) = 1]\right|.$$

The scheme $\mathcal{PE}$ is said to be IK-CPA secure (respectively IK-CCA secure) if the function $\mathbf{Adv}^{\mathrm{ik-cpa}}_{\mathcal{PE},A_{\mathrm{cpa}}}(\cdot)$ (resp. $\mathbf{Adv}^{\mathrm{ik-cca}}_{\mathcal{PE},A_{\mathrm{cca}}}(\cdot)$) is negligible for any adversary A whose time complexity is polynomial in k.

The "time-complexity" is the worst-case execution time of the experiment plus the size of the code of the adversary, in some fixed RAM model of computation.

3.2 RSA-RAEP by Bellare, Boldyreva, Desai, and Pointcheval

A simple observation that seems to be folklore is that standard RSA encryption does not provide key-privacy, even when all moduli in the system have the same length. Suppose an adversary knows that the ciphertext y is created under one of two keys (N_0, e_0) or (N_1, e_1), and suppose $N_0 \leq N_1$. If $y \geq N_0$ then the adversary bets it was created under (N_1, e_1), else it bets it was created under (N_0, e_0). It is not hard to see that this attack has non-negligible advantage.

In [1], they proposed an RSA-based encryption scheme which is secure in the sense of IK-CCA. It is RSA-RAEP which is a variant of RSA-OAEP. Since their variant chooses N from $(2^{k-2}, 2^k)$, it simply repeats the ciphertext computation, each time using new coins, until the ciphertext y satisfies $y < 2^{k-2}$.

Definition 8 (RSA-RAEP [1]). *RSA-RAEP $= (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ is as follows. The common-key generation algorithm $\mathcal{G}$ takes a security parameter k and returns parameters k, k_0 and k_1 such that $k_0(k)+k_1(k) < k$ for all $k > 1$. This defines an associated plaintext-length function $n(k) = k - k_0(k) - k_1(k)$. The key generation algorithm $\mathcal{K}$ takes k, k_0, k_1, runs the key-generation algorithm of* RSA, *and gets (N, e) and (N, d). The public key pk is $(N, e), k, k_0, k_1$ and the secret key sk is $(N, d), k, k_0, k_1$. The other algorithms are depicted below. Let $G : \{0,1\}^{k_0} \to \{0,1\}^{n+k_1}$ and $H : \{0,1\}^{n+k_1} \to \{0,1\}^{k_0}$. Note that $[x]^n$ denotes the n most significant bits of x and $[x]_m$ denotes the m least significant bits of x.*

Algorithm $\mathcal{E}^{G,H}_{pk}(x)$
- $ctr = -1$
- **repeat**
  - $ctr \leftarrow ctr + 1$
  - $r \xleftarrow{R} \{0,1\}^{k_0}$
  - $s \leftarrow (x \,||\, 0^{k_1}) \oplus G(r)$; $t \leftarrow r \oplus H(s)$
  - $v \leftarrow (s||t)^e \bmod N$
- **until** $((v < 2^{k-2}) \vee (ctr = k_1))$
- **if** $(ctr = k_1)$ $y \leftarrow 1||0^{k_0+k_1}||x$
- **else** $y \leftarrow 0||v$
- **return** y

Algorithm $\mathcal{D}^{G,H}_{sk}(y)$
- $b \leftarrow [y]^1$; $v \leftarrow [y]_{k_0+k_1+n}$
- **if** $(b = 1)$
  - $w \leftarrow [v]^{k_0+k_1}$; $x \leftarrow [v]_n$
  - **if** $(w = 0^{k_0+k_1})$ $z \leftarrow x$ **else** $z \leftarrow \perp$
- **else**
  - $s \leftarrow [v^d]^{n+k_1}$; $t \leftarrow [v^d]_{k_0}$
  - $r \leftarrow t \oplus H(s)$
  - $x \leftarrow [s \oplus G(r)]^n$; $p \leftarrow [s \oplus G(r)]_{k_1}$
  - **if** $(p = 0^{k_1})$ $z \leftarrow x$ **else** $z \leftarrow \perp$
- **return** z

They proved RSA-RAEP is secure in the sense of IND-CCA2 and IK-CCA in the random oracle model assuming RSA is one-way. In RSA-RAEP, the expected number of exponentiations for encryption is

$$\sum_{i=1}^{k_1} i \left(1 - \frac{2^{k-2}}{N}\right)^{i-1} \frac{2^{k-2}}{N} = \frac{1-(1-p)^{k_1}}{p} - k_1(1-p)^{k_1}$$

where $p = 2^{k-2}/N$. Suppose that N is uniformly distributed in $(2^{k-2}, 2^k)$, the expected number of this scheme is two. However, the upper bound of the number of exponentiations for encryption is $k_1 (\gg 2$, security parameter).

3.3 Our Proposed Encryption Scheme

In this section, we propose our encryption scheme, which uses RSACD instead of RSA.

Definition 9. *The common-key generation algorithm $\mathcal{G}$, and the oracles G and H are the same as RSA-RAEP. The key generation algorithm $\mathcal{K}$ is almost the same as RSA-RAEP. The difference is running the key-generation algorithm of* RSACD *instead of* RSA. *The other algorithms are described as follows. Note that the valid ciphertext y satisfies $y \in [0, 2^k)$ and $y \bmod N \in \mathbb{Z}_N^*$.*

$\mathtt{Algorithm}\ \mathcal{E}_{pk}^{G,H}(x)$	$\mathtt{Algorithm}\ \mathcal{D}_{sk}^{G,H}(y)$
$r \xleftarrow{R} \{0,1\}^{k_0}$	$s \leftarrow [g_{N,d,k}(y)]^{n+k_1};\ t \leftarrow [g_{N,d,k}(y)]_{k_0}$
$s \leftarrow (x \mid\mid 0^{k_1}) \oplus G(r)$	$r \leftarrow t \oplus H(s)$
$t \leftarrow r \oplus H(s)$	$x \leftarrow [s \oplus G(r)]^n;\ p \leftarrow [s \oplus G(r)]_{k_1}$
$v \leftarrow f_{N,e,k}(s\|\|t)$	$\mathtt{if}\ (p = 0^{k_1})\ z \leftarrow x\ \mathtt{else}\ z \leftarrow \perp$
$\mathtt{return}\ y$	$\mathtt{return}\ z$

Using Theorem 1 and 2, we can prove the following theorem.

Theorem 3. *Our scheme is secure in the sense of IND-CCA2 and IK-CCA in the random oracle model assuming* RSA *is one-way.*

Proof (Idea). Fujisaki, Okamoto, Pointcheval, and Stern [4] proved OAEP with partial one-way permutation is secure in the sense of IND-CCA2. Thus, OAEP with $f_{N,e,k}$ is secure in the sense of IND-CCA2 assuming RSACD is partial one-way.

Bellare, Boldyreva, Desai, and Pointcheval [1] proved RSA-RAEP is secure in the sense of IK-CCA in the random oracle model assuming RSA is partial one-way. Noticing that the function $f_{N,e,k}$ and $g_{N,d,k}$, and the domain of valid signature change, we can prove in a similar way that our scheme is secure in the sense of IK-CCA in the random oracle model assuming RSACD is partial one-way.

Therefore, by Theorem 1 and 2, we can prove that our scheme is secure in the sense of IND-CCA2 and IK-CCA under the assumption that RSA is one-way. □

In this scheme, the expected number of exponentiations in encryption is

$$1 \times \frac{2(2^k - N)}{2^k} + 2 \times \left(1 - \frac{2(2^k - N)}{2^k}\right) = \frac{N}{2^{k-1}}.$$

Suppose that N is uniformly distributed in $(2^{k-1}, 2^k)$, the expected number of our scheme is two, the same as RSA-RAEP. In our scheme, the number of exponentiations for encryption is at most two, while in RSA-RAEP, the upper bound of this number is k_1 ($\gg 2$, security parameter). Notice that we use the randomness only for an RSA-OAEP.

4 Application to Ring Signature

4.1 Definitions of Ring Signature

In [2], Rivest, Shamir, and Tauman proposed the notion of ring signature, which allows a member of an ad hoc collection of users S to prove that a message is authenticated by a member of S without revealing which member actually produced the signature. Unlike group signature, ring signature has no group managers, no setup procedures, no revocation procedures, and no coordination.

Definition 10 (Ring Signature [2]). *One assumes that each user (called a ring member) has received (via a PKI or a certificate) a public key P_k, for which the corresponding secret key is denoted by S_k. A ring signature scheme consists of the following algorithms.*

- **ring-sign**$(m, P_1, P_2, \cdots, P_r, s, S_s)$ *which produces a ring signature σ for the message m, given the public keys $P_1, P_2, \cdots, P_r$ of the r ring members, together with the secret key S_s of the s-th member (who is the actual signer).*
- **ring-verify**(m, σ) *which accepts a message m and a signature σ (which includes the public key of all the possible signers), and outputs either "true" or "false".*

The signer does not need the knowledge, consent, or assistance of the other ring members to put them in the ring. All he needs is knowledge of their regular public keys. Verification must satisfy the usual soundness and completeness conditions, but in addition the signature scheme must satisfy "signer-ambiguous", which is the property that the verifier should be unable to determine the identity of the actual signer with probability greater than $1/r + \epsilon$, where r is the size of the ring, and ϵ is negligible.

4.2 RSA-based Ring Signature Scheme by Rivest, Shamir, and Tauman

In [2], they constructed ring signature schemes in which all the ring member use RSA as their individual signature schemes. We review their scheme.

Let ℓ, k, and b be security parameters. Let E be a symmetric encryption scheme over $\{0,1\}^b$ using ℓ-bit keys and h be a hash function which maps arbitrary strings to ℓ-bit strings. They use h to make a key for E. They assume that each user has an RSA public key $P_i = (N_i, e_i)$ which specifies the trap-door one-way permutation f_i on $\mathbb{Z}_{N_i}$: $f_i(x) = x^e \bmod N_i$.

To sign and verify a ring signature, they proposed a combining function $C_{k,v}$ based on a symmetric encryption scheme E modeled by a (keyed) random permutation

$$C_{k,v}(y_1, \cdots, y_r) = E_k(y_r \oplus E_k(y_{r-1} \oplus \cdots E_k(y_2 \oplus E_k(y_1 \oplus v)) \cdots))$$

where v is an initialization value. In their scheme, the inputs y_i to the combining function are computed as $g_i(x_i)$ for some $x_i \in \{0,1\}^b$. They defined the extended trap-door permutation g_i over $\{0,1\}^b$ which has a common domain for each user as follows: for any b-bit input x_i define nonnegative integers q_i and r_i so that $x_i = q_i N_i + r_i$ and $0 \le r_i < N_i$. Then

$$g_i(x_i) = \begin{cases} q_i N_i + f_i(r_i) & \text{if } (q_i+1)N_i \le 2^b \\ x_i & \text{otherwise.} \end{cases}$$

If b is sufficiently large (e.g. 160 bits larger than any of the N_i), g_i is a one-way trap-door permutation. (See also [5].)

A ring signature on a message m consists in a tuple $(v, x_1, \cdots, x_r)$ and the signature is valid iff $C_{h(m),v}(g_1(x_1), \cdots, g_r(x_r)) = v$. For any message m, any fixed values v and $\{x_i\}_{i \ne s}$, one can efficiently compute the value y_s such that the combining function outputs v by using the following equation:

$$y_s = E_k^{-1}(y_{s+1} \oplus \cdots E_k^{-1}(y_r \oplus E_k^{-1}(v)) \cdots) \oplus E_k(y_{s-1} \oplus \cdots E_k(y_1 \oplus v) \cdots).$$

Now using her knowledge of the trap-door for function f_s, s-th member (the actual signer) is able to compute x_s such that $g_s(x_s) = y_s$. Thus, the ring member can generate a valid signature. Rivest, Shamir, and Tauman proved this scheme is unconditionally signer-ambiguous and provably secure in the random oracle model assuming RSA is one-way.

4.3 Our Proposed Ring Signature Scheme

Unlike group signature, ring signature has no group managers, no setup procedures, no revocation procedures, and no coordination, and each user can use a public key whose length is different from other users.

In [2], Rivest, Shamir, and Tauman mentioned the case that a member of the cabinet of some country wished to leak her secret to a journalist. In this kind of situation, it was reasonable to consider that the members of the same group use the RSA moduli of the same length. In our scheme, we assume the situation that each user chooses her public key with the same size.

Our scheme is almost the same as the previous scheme. The difference is using $f_{N_i,e,k}(\cdot)$ in Section 2.2 instead of $g_i(\cdot)$ in Section 4.2. Then, the domain

of $f_{N,e,k}(\cdot)$ is $\{0,1\}^k$, while that of the previous scheme is $\{0,1\}^{k+160}$, where k is the length of the RSA moduli. Thus, we can reduce the size of signature in this situation. In particular, the size of signature of our scheme is 160 bits smaller than that of the previous scheme in order to archive security parameter $k = 1024$. In our scheme, the number of exponentiations is one or two, while that of the original scheme in [2] is one. Since $f_{N_i,e,k}(\cdot)$ is a trap-door one-way permutation as well as $g_i(\cdot)$, we can easily prove the following theorem in a similar way as for the previous scheme.

Theorem 4. *Our scheme is unconditionally signer-ambiguous and provably secure in the random oracle model assuming* RSA *is one-way.*

We can also apply this scheme to the Rabin-based ring signature scheme in [2] in a similar way.

5 Conclusion

In this paper, we have constructed the RSA family of trap-door permutations with a common domain and proposed the applications of our construction to the key-privacy encryption and ring signature schemes, which have some advantage to the previous schemes. It might be interesting to consider other applications of our RSA family, and different constructions of a family of trap-door permutations with a common domain.

References

1. Bellare, M., Boldyreva, A., Desai, A., Pointcheval, D.: Key-Privacy in Public-Key Encryption. [6] 566–582
2. Rivest, R.L., Shamir, A., Tauman, Y.: How to Leak a Secret. [6] 552–565
3. Bellare, M., Rogaway, P.: Optimal asymmetric encryption – How to encrypt with RSA. In Santis, A.D., ed.: Advances in Cryptology – EUROCRYPT '94. Volume 950 of Lecture Notes in Conputer Science., Perugia, Italy, Springer-Verlag (1994) 92–111
4. Fujisaki, E., Okamoto, T., Pointcheval, D., Stern, J.: RSA-OAEP is Secure under the RSA Assumption. In Kilian, J., ed.: Advances in Cryptology – CRYPTO 2001. Volume 2139 of Lecture Notes in Conputer Science., Santa Barbara, California, USA, Springer-Verlag (2001) 260–274
5. Desmedt, Y.: Securing traceability of ciphertexts: Towards a secure software escrow scheme. In Guillou, L.C., Quisquater, J.J., eds.: Advances in Cryptology – EUROCRYPT '95. Volume 921 of Lecture Notes in Conputer Science., Saint-Malo, France, Springer-Verlag (1995) 147–157
6. Boyd, C., ed.: Advances in Cryptology – ASIACRYPT 2001. In Boyd, C., ed.: Advances in Cryptology – ASIACRYPT 2001. Volume 2248 of Lecture Notes in Conputer Science., Gold Coast, Australia, Springer-Verlag (2001)

A New Variant of the Matsumoto-Imai Cryptosystem through Perturbation

Jintai Ding

Department of Mathematical Sciences,
University of Cincinnati,
Cincinnati, OH, 45220,
USA
ding@math.uc.edu

Abstract. Though the multivariable cryptosystems first suggested by Matsumoto and Imai was defeated by the linearization method of Patarin due to the special properties of the Matsumoto-Imai (MI) cryptosystem, many variants and extensions of the MI system were suggested mainly by Patarin and his collaborators. In this paper, we propose a new variant of the MI system, which was inspired by the idea of "perturbation". This method uses a set of r (a small number) linearly independent linear functions $z_i = \sum_{j=1}^{n} \alpha_{ij} x_j + \beta_i$, i=1,..,r, over the variables x_i, which are variables of the MI system. The perturbation is performed by adding random quadratic function of z_i to the MI systems. The difference between our idea and a very similar idea of the Hidden Field Equation and Oil-Vinegar system is that our perturbation is internal, where we do not introduce any new variables, while the Hidden Field Equation and Oil-Vinegar system is an "external" perturbation of the HFE system, where a few extra (external) new variables are introduced to perform the perturbation. A practical implementation example of 136 bits, its security analysis and efficiency analysis are presented. The attack complexity of this perturbed Matsumoto-Imai cryptosystem is estimated.
Keywords: open-key, multivariable, quadratic polynomials, perturbation

1 Introduction

Since the invention of the RSA scheme, there has been great interest to seek new public key cryptosystems, which may serve us better for different purposes. One direction to look for such systems is based on multivariable polynomials, in particular, quadratic polynomials. This method relies on the proven theorem that solving a set of multivariable polynomial equations over a finite field, in general, is an NP-hard problem, which, however, does not guarantee the security.

One of the basic ideas to design such a system was started by Matsumoto and Imai [MI], where they suggested to use a map F over a large field $\bar{K}$, a degree n extension of a finite field k. Through identifying $\bar{K}$ as k^n, first, one would identify this map F as a multivariable polynomial map from k^n to k^n, which we

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 305–318, 2004.

call $\tilde{F}$; then, one would "hide" this map $\tilde{F}$ by composing from both sides by two invertible affine linear maps L_1 and L_2 on k^n. This gives a quadratic map

$$\bar{F} = L_1 \circ \tilde{F} \circ L_2$$

from k^n to k^n (by $\circ$, we mean composition of two maps). The map F suggested by Matsumoto and Imai is the map

$$F : X \longmapsto X^{1+q^i},$$

where q is the number of elements in k, X is an element in $\bar{K}$ and k is of characteristic 2. However this scheme was proven insecure under an algebraic attack using the linearization equations by Patarin [P].

Since then, there has been intensive developments by Patarin and his collaborators to find all possible modifications and extensions of the Matsumoto-Imai systems, which are secure. Those ideas to directly extend the Matsumoto-Imai system can be divided into three groups in accordance with the method used.

1) Minus-Plus method [CGP1]: This is the simplest idea among all, namely one takes out (Minus method, which was first suggested in [S]) a few of the quadratic polynomial components of $\bar{F}$, and (or) add (Plus method) a few randomly chosen quadratic polynomials. The main reason to take the "Minus" action is due to security concerns. The Minus (only) method is very suitable for signature schemes. One of them is Sflash [ACDG, CGP], and it was recently accepted as one of the final selections in the New European Schemes for Signatures, Integrity, and Encryption: IST-1999-12324.

2) Hidden Field Equation Method (HFE) [P]: This method is suggested by Patarin to be the strongest. In this case, the difference from the original Matsumoto-Imai system is that F is replaced by the map (function)

$$F : X \longmapsto \sum_{i,j}^{A} a_{ij} X^{q^i+q^j} + \sum_{i}^{B} b_i X^{q^i} + c,$$

where the coefficients are randomly chosen and the total degree of F must be small, otherwise the decryption process will become too slow. However a new algebraic attack using both the Minrank method and the relinearization method by Kipnis and Shamir [KS] shows that the number A can not be too small, but if A is big, the system is too slow due to the process of solving the polynomial equation in the decryption process. This is further confirmed by [C, FJ].

3) Hidden Field Equation and Oil-Vinegar Method [CGP2]: After the Hidden Field Equation Method, it is suggested to combine the Hidden Field Equation Method with another new method, Oil-Vinegar method. The basic idea is, on top of the HFE method, to add a few new variables to make the system more complicated. This method is essentially to replace F with an even more complicated function:

$$F : (X, \bar{X}) \longmapsto$$

$$\sum_{i,j}^{A} a_{ij} X^{q^i+q^j} + \sum_{i,j}^{B,B'} b_{i,j} X^{q^i} \bar{X}^{q^j} + \sum_{i,j}^{A'} \alpha_{ij} \bar{X}^{q^i+q^j} + \sum_{i,j}^{B'} \beta_i' \bar{X}^{q^i} + \sum_{i,j}^{B} b_i X^{q^i} + c,$$

where the new Vinegar variables given by the variable $\bar{X}$ is of a small dimension. One can see that these new variables are mixed in a special way with the original variables (like Oil and Vinegar). The decryption process requires an exhaustive search on these added small number of variables. For the signature case the search becomes a random selection, which has a good probability to succeed each time, and it continues until a correct answer is found. We recently observed that the attack in [KS] can also be applied here to actually eliminate the small number of added variables and attack the system. The basic idea is to use the algebraic method to find a way to purge out the Vinegar variables.

After all the efforts mentioned above, it seems that all the possible extensions and generalizations of the Matsumoto-Imai system are exhausted, but our construction provides another alternative.

The motivation for our work is to develop new constructions that could be strongly resistant to the algebraic attack [P, KS] and its extensions like XL, but without much sacrifice to the efficiency of the system.

From a very general point of view, the third method above (the HFE and Oil-Vinegar method) can also be interpreted as an extension of a commonly used idea in mathematics and physics, perturbation. Namely a good way to deal with a continuous system often is to "perturb" the system at a minimum scale. The HFE and Oil-Vinegar method can be viewed as a perturbation of the HFE method by the newly added Vinegar variables. However, because of the "Oil-Vinegar" idea, this perturbation, in some sense, is more of an "external" perturbation, where a few extra (external) new variables (Vinegar) are introduced to do so.

For our construction, the idea is very similar, nevertheless, what we suggest is rather an idea of "internal" perturbation. Our perturbation is performed through a small set of variables "inside" the space k^n (therefore they are "internal" variables) and we do not introduce any new variables. Namely given a quadratic multivariable system $\bar{F}$ over k^n, we randomly find a surjective affine linear map Z from k^n to k^r with a small dimension r, then we try to "perturb" the system through the small number variables related to Z.

This idea of internal perturbation is a very general idea that can be applied to all existing multivariable cryptosystems.

A suitable example is the case of Matsumoto-Imai system. The perturbation is performed by two steps:

1) first, we randomly choose r (small) linearly independent functions:

$$z_i = \sum_{i}^{n} \alpha_{ij} x_j + \beta_i,$$

where x_i are the variables of $\tilde{F}$, which can be treated as components of a surjective affine linear map Z from k^n to k^r;

2) then, we add randomly quadratic polynomial of z_i to the components of $\tilde{F}$ to define a new map $\tilde{\tilde{F}}$ to replace $\tilde{F}$:

$$\tilde{\tilde{F}}(x_1, ..., x_n) = (\tilde{F}_1(x_1, ..., x_n) + f_1(z_1, .., z_r), \tilde{F}_2(z_1, ..., z_n) + f_2(z_1, ..., z_r), \cdots,$$
$$\tilde{F}_n(x_1, ..., x_n) + f_n(z_1, ..., z_r)).$$

The rest is the same as that of the Matsumoto-Imai system.

In this case, the third method above is not applicable here due to the fact that there are no linear terms to mix Oil and Vinegar.

We will call our method hidden perturbation equation method due to the hidden equations that define the perturbation.

The advantages of such new systems include the facts that they may be able to resist well existing algebraic attacks [KS, C], which may make the system very secure, and the internal perturbation makes the process of elimination of unnecessary candidates in the decryption process much faster.

In the first section of the paper, we will introduce in detail the application of our general method to the Matsumoto-Imai cryptosystem. Then we will present a practical implementation example with 136 bits for the perturbation of a Matsumoto-Imai system, where we choose r to be 6. We will show that it should have a very high security level against all the known attacking methods. We will analyze the security and efficiency of the system and compare them with other multivariable cryptosystems with similar parameters.

2 Perturbation of Matsumoto-Imai System

2.1 The Original Matsumoto-Imai Cipher

Let $\bar{K}$ be a degree n extension of a finite field k of characteristic 2 with q elements, and $\bar{K} \cong k[x]/g(x)$, where $g(x)$ is a degree n irreducible polynomial over k. In general, the condition of characteristic 2 is not necessary, then we should modify the system slightly due to the multiplicity concern of the final map.

Let ϕ be the standard k-linear map that identify $\bar{K}$ with k^n:

$$\phi : \bar{K} \longmapsto k^n,$$

such that

$$\phi(a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}) = (a_0, a_1, a_2, \cdots, a_{n-1}).$$

Let

$$F(X) = X^{1+q^i},$$

over $\bar{K}$ such that g.c.d.$(1 + q^i, q^n - 1) = 1$.

F is an invertible map and its inverse is given by

$$F^{-1}(X) = X^t,$$

where $t(1 + q^i) = 1$ modulo $(q^n - 1)$.

Let $\tilde{F}$ be a map over k^n and

$$\tilde{F}(x_1, ..., x_n) = \phi \circ F \circ \phi^{-1}(x_1, ..., x_n)$$

$$= (\tilde{F}_1(x_1, ..., x_n), \tilde{F}_2(x_1, ..., x_n), \cdots, \tilde{F}_n(x_1, ..., x_n)).$$

Here $\tilde{F}_i(x_1, ..., x_n)$ are quadratic polynomials of n variables.

Let L_1 and L_2 be two randomly chosen invertible affine linear maps over k^n.

$$\bar{F}(x_1, ..., x_n) = L_1 \circ \tilde{F} \circ L_2(x_1, ..., x_n)$$

$$= (\bar{F}_1(x_1, ..., x_n), \bar{F}_2(x_1, ..., x_n), \cdots, \bar{F}_n(x_1, ..., x_n))$$

is the cipher suggested by Matsumoto-Imai, which was defeated by the algebraic attack using linearization equations by Patarin.

2.2 The Perturbed Matsumoto-Imai Cipher

Let r be a small number and

$$z_1(x_1, ..., x_n) = \sum_1^n \alpha_{j1} x_j + \beta_1,$$

$$\cdots$$

$$z_r(x_1, .., x_n) = \sum_1^n \alpha_{jr} x_j + \beta_r,$$

be a set of randomly chosen linear functions of x_i over k^n such that the terms of degree one are linearly independent. Let

$$Z(x_1, ..., x_n) = (z_1, \cdots, z_r) = (\sum_i^n \alpha_{i1} x_j + \beta_1, \cdots, \sum_i^n \alpha_{ir} x_j + \beta_r),$$

which gives a map from k^n to k^r.

Let

$$\bar{\bar{F}}(x_1, ..., x_n) = (\bar{\bar{F}}_1(x_1, ..., x_n), \bar{\bar{F}}_2(x_1, ..., x_n), \cdots, \bar{\bar{F}}_n(x_1, ..., x_n))$$

$$= (\tilde{F}_1(x_1, ..., x_n) + f_1(z_1, .., z_r), \tilde{F}_2(z_1, ..., z_n) + f_2(z_1, ..., z_r), \cdots,$$

$$\tilde{F}_n(x_1, ..., x_n) + f_n(z_1, ..., z_r)),$$

where f_i are randomly chosen quadratic polynomials with r variables.

Let $f(z_1, .., z_r) = (f_1(z_1, .., z_r), f_2(z_1, ..., z_r), \cdots, f_r(z_1, ..., z_r))$ and f can be viewed as a map from k^r to k^n. Let P be the set consisting of the pairs (λ, μ), where λ is a point that belongs to the image of f, and μ is the set of pre-images of λ under f. We call P the perturbation set. Here, we know that P has q^r elements probabilistically, and it does not include any pair whose first component is the zero vector.

We call $\bar{\bar{F}}$ a perturbation of $\tilde{F}$ by Z.

$$\hat{F}(x_1, ..., x_n) = L_1 \circ \bar{\bar{F}} \circ L_2(x_1, ..., x_n) = (y_1(x_1, ..., x_n), ..., y_n(x_1, ..., x_n)),$$

where y_i are quadratic polynomial components of $\hat{F}$. We call $\hat{F}$ the perturbed Matsumoto-Imai cipher.

Let $\tilde{f}(x_1, .., x_n) = f(z_1(x_1, ..., x_n), .., z_r(x_1, ..., x_n))$, which is a map from k^n to k^n. We can see that

$$\hat{F}(x_1, ..., x_n) = L_1 \circ \tilde{F} \circ L_2 + L_1 \circ \tilde{f} \circ L_2(x_1, ..., x_n),$$

and the perturbation is performed by just adding $L_1 \circ \tilde{f} \circ L_2(x_1, ..., x_n)$ to the original Matsumoto-Imai cipher.

We can use it to establish a public key cryptosystem.

2.3 The Public Key

The public key include

1) the field k including its addition and multiplication structure;
2) the n quadratic polynomials $y_1(x_1, ..., x_n), ..., y_n(x_1, ..., x_n)$.

2.4 Encryption

Given a message vector $M = (x'_1, ..., x'_n)$ as the plaintext, the ciphertext is the vector

$$(y'_1, ..., y'_n) = (y_1(x'_1, ..., x'_n), ..., y_n(x'_1, ..., x'_n)).$$

2.5 The Private Key and the Decryption

The private key includes:

1) the map F,
2) the set of linear functions $z_1, ..., z_r$,
3) the set of points in P (or the set of the polynomials $f_i(z_1, ..., z_r)$),
4) the two affine linear maps L_1, L_2.

2.6 Decryption

Once we have the ciphertext $(y'_1, ..., y'_n)$, the decryption includes the following steps:

I) compute $(\bar{y}_1, ..., \bar{y}_n) = L_1^{-1}(y'_1, ..., y'_n)$;

II) take all the elements one by one (λ, μ) in P, compute

$$(y_{\lambda 1}, ..., y_{\lambda n}) = \phi^{-1} \circ F^{-1}((\bar{y}_1, ..., \bar{y}_n) + \lambda),$$

and check if $Z(y_{\lambda 1}, ..., y_{\lambda n})$ is the same as the corresponding μ, if no, discard it, if yes, go to next step;

III) compute $(x_{\lambda 1}, ..., x_{\lambda n}) = L_2^{-1} \circ \phi(y_{\lambda 1}, ..., y_{\lambda n})$. If there is only one solution, it is the plaintext. However, it is very possible that we have more than one solution, then we can use the same technique as suggested for the HFE method, namely we can use a few hash functions to differentiate which one is the right one. In our computer experiments, it seems that, in general the multiplicity seems to be surprisingly small, and the multiplicity of solutions behaves as expected like that of randomly chosen functions.

We call our system a perturbed Matsumoto-Imai cryptosystem (PMI). It is evident that our method is a very general method that it can be used to perturb any multivariable cryptosystem, such as that the HFE cryptosystem. After perturbation, the security should be much stronger, but the decryption process is slower (by a factor of q^r).

3 A Practical Implementation

For practical use, we suggest a 136 bits implementation of the PMI system.

We choose k to be F_2.

We choose $\bar{K}$ to be an 136 degree extension of F_2 and

$$g(x) = 1 + x + x^2 + x^{11} + x^{136}.$$

We choose r to be 6, which means the dimension of the perturbation space is 6.

We choose

$$F(X) = X^{2^{5\times 8}+1},$$

In general, to have a security level of 2^{80}, we suggest n to be at least 96 and r not less than 5. Our implementation example has a much stronger security level at around 2^{136}.

3.1 Implementation

Public Key Size The public key contains 136 quadratic polynomials. Each polynomial has 136 × 137/2 quadratic terms, 136 linear terms and one constant term. The key size is about 100K bytes, which is rather big, but should not be a problem for any PC.

Encryption Computation Complexity For encryption, we need to compute the value of a set of quadratic polynomials for a given set of $x_1, ..., x_n$, we can rewrite a quadratic polynomial in the following way:

$$\sum_1^n x_i(b_i + \sum_{j=i}^n a_{i,j}x_j) + c,$$

which allows us to compute the value at roughly one and an half times of the speed of a direct calculation. Therefore we need roughly 19,000 binary (including both addition and multiplication) operations to calculate the value of each polynomial. Therefore, on average, each message bit needs 19,000 binary operations.

Private Key Size The private key is much smaller in general, the main parts are: the 8 linear functions z_i, which is of the size 127×8 bits, the two linear transformations L_1 and L_2 and their inverses, which needs $127 \times 128 \times 4$ bits and the perturbation set P, which needs roughly $64 \times 3 \times 64$ bits. The total is around 80,000 bits.

Decryption Computation Complexity For decryption, we need first calculate the action of L_1^{-1} on the ciphertext vector, which needs roughly 136 x 136 + 136 calculations, which can be neglected compared to the computations required for the second step. The same is true for the third step. The main part of the decryption process is the step II, where we need to calculate 64 times the values of F^{-1} and the values of z_i and compare it with the second components of the corresponding element in P. The main part surely is to calculate F^{-1}. Due to the linearization method by Patarin, we can actually find F^{-1} by solving a set of homogeneous linear equations. In this case, we will implement a fast algorithm to accomplish this as follows.

1) We identify $\bar{K}$ as a degree 17 extension of a field $\tilde{K}$, which is a degree 8 extension over F_2. In this case we can identify $\bar{K}$ as $\tilde{K}^8$.

2) The map $F(X) = X^{2^{5\times 8}+1}$ can then be identified again as a quadratic map on $\tilde{K}^8$. Then with the relinearization by Patarin, finding the inverse of K becomes the process of solving a set of 17 homogeneous linear equations of rank 16, and then solving an equation in the form $x^2 = b$ over the field $\tilde{K}$.

3)This process can be performed by making a multiplication table for the field $\tilde{K}$. The table takes $2^{16} \times 24$ bits and each search is on a space of 2^{16} bits. Overall, each F^{-1} calculation becomes a process mainly to solve a set of 17 linear equations over the field $\tilde{K}$. Because the message is 136 bits, one can conclude that the decryption process will take roughly half of the time to solve a 17 linear equations over $\tilde{K}$ per bit.

We may also use the algorithm in [ACDG] to make this process even faster.

3.2 Security Analysis

In general, a set of 136 quadratic polynomials with 136 variables, are difficult to solve. However, special methods are invented to attack specially designed systems. The Matsumoto-Imai system itself is not secure, which mainly is due to the linearization attack. Namely, any given a plaintext $(x_1, ..., x_n)$ and its ciphertext $(y_1, ..., y_n)$ satisfy a set of linearization equations in the form

$$\sum_i x_i \sum_j a_{i,j} y_j = 0.$$

These equations essentially allow us to find enough linear equations satisfied by the plaintext from a ciphertext to defeat the system. Since then, new methods have been invented to attack multivariable cryptosystems, mainly the algebraic method [KS] and its extension the XL method, and for the case of Matsumoto-Imai Minus system, a method to search for "missing" terms.

Next, we will analyze one by one the impact of all the existing attacking methods on the perturbed Matsumoto-Imai system.

The Attack by Linearization Method From the name, we can see the PMI system should have a lot in common with the original MI system.

Let

$$H_1 = \{Y | Y = \sum_i a_i \bar{F}_i(x_1, ..., x_n)\},$$

where $\bar{F}_i$ are components of the original Matsumoto-Imai cipher.

Let

$$H_2 = \{Y | Y = \sum_i a_i \bar{y}_i(x_1, ..., x_n)\},$$

where $\bar{y}_i$ are components of the perturbed Matsumoto-Imai cipher.

Let

$$H_3 = H_1 \cap H_2.$$

Because the perturbation dimension is 6, the dimension of all the linear and quadratic polynomials of a dimension 6 space of F_2 is 21, where 15 are from quadratic terms and 6 are from linear terms, the dimension of H_3 is, therefore 115=136-21. Intuitively, one can view our system as if we take out 21 terms out of the total 136 public polynomials. This clearly eliminates all the possible linearization equations, which we confirm by our computer experiment. Therefore, the linearization method cannot be applied here to attack the system.

The Attack Methods Related to the MI Minus Systems The MI Minus systems are suggested for signature purpose. The method simply takes out a few public quadratic polynomials (Minus method) to improve the security. The main attack method of this system is to search for quadratic polynomials we can add to the system such that it becomes the original MI system. The search process uses the property that the map F is a permutation polynomial on the field $\bar{K}$. This will allow the algebraic attack using linearization equations by Patarin to be applied successfully again.

For the PMI case, this method is not applicable due to mainly two reasons.

1) In the PMI systems, the perturbed map is not any more an injective (also not surjective) map, therefore the properties of permutation polynomials can no longer be applied here to search for the missing terms, because no terms is actually missing.

2) For our case, finding the "missing terms" is essentially to purge out the perturbation. For the pure Matsumoto-Imai Minus system, the attacker uses the fact that there is a good set of polynomials, namely the given polynomials actually come from the original MI system. For our case this is no longer the case, as all terms are mixed together. Therefore, there does not exist a good way to find the subspace of dimension 115 of the polynomials from the original MI system, namely the subspace H_3. One possible way is certainly just to guess which one is from H_3 and the probability to guess a right one is 1/64, which is

not bad at all. The problem is that we have no way to judge if anyone is the right guess or not. We conclude it is essentially impossible to find the missing terms through this way.

The PMI and the Matsumoto-Imai Plus-Minus systems can be viewed in a very similar way. The similarity is that in the PMI system, we take out 21 quadratic polynomials, and add 21 new polynomials, except that in our case, we did not add randomly 21 polynomials, but 21 perturbed polynomials. The attack on the Matsumoto-Imai Plus-Minus system is essentially the XL method, which we will discuss below.

The Attack Methods on the HFE The special advantage of the perturbed MI system, is its resistance to the algebraic attack methods that first was suggested for attacking the HFE systems. The basic attacking point of the algebraic attack method [KS] is that the quadratic part of the HFE: $\sum_{i,j}^{A} a_{ij} X^{q^i+q^j}$ can be viewed as a quadratic form with its variables being X^{q^i}, and the attack is to find a transformation to reduce a quadratic form into the above form with low rank using Minrank method. For the PMI systems, this method is not applicable due to the fact that there is no way that when using the above method, the perturbed polynomials can be rewritten into low rank quadratic form. The reason for this is that, in the attack process using the algebraic method in [KS], the map Z from k^{136} to k^6 is lifted as an embedding map $\bar{Z}$ from k^n to k^n:

$$\bar{Z}(x_1, ..., x_n) = (Z(x_1, ..., x_n), 0, .., 0);$$

then it is further lifted as a k affine linear map from $\bar{K}$ to $\bar{K}$ in the form of

$$\tilde{Z}(X) = \sum_{i}^{\bar{A}} a_j X^{q^i},$$

where the highest term $\bar{A}$ should be at least 130, because the dimension of the pre-image of any point of this map is 130. Therefore, from the analysis of the efficiency of this method [KS, C], we know it should take much more than 2^{136} computations to defeat the system by this method. This suggests that it should resist other related attacks as well [CDF, FJ].

XL Attack The XL method is a very general method for solving multivariable equations. This method can be viewed as a generalization of the algebraic attack using linearization equations, where one basically has to search for functions of only one variable in the ideal generated by $Y_i = y_i(x_1, ..., x_n) - y_i'$ by looking at linear combinations of terms like

$$\sum_{l \le D-2} a_{i_1 i_2 i_3 ... i_l} x_{i_1} x_{i_2} x_{i_3} ... x_{i_l} Y_i,$$

where y_i is the i-th public polynomial and y_i' is the corresponding component of the ciphertext. The success of this method depends on the degree D. In

[CKPS], there is an argument about asymptotic growth of the complexity to attack a system with more equations than variables, but no final conclusion about the complexity is given. What is given are estimates based on some computer experiments, in particular, the case when there are 2 or more equations than variables, which for our case can be easily achieved by guessing values of any 2 variables. According to their estimate, for our case, D should be 12, which is given by the square root of 136, and the XL attack needs to do a Gaussian elimination on about 136!/124!12! roughly 2^{55} variables, which requires more than 2^{136} operations.

For the case of F_2, there exist improved versions of XL [CP], for example, by adding the equations $x_i^2 = x_i$ into the system and one may argue that the PMI system is not a general system, but a system based on the perturbation of the MI system. Therefore the attack complexity might be different. It is reasonable to believe that D should be determined by r in our case. For this, we did some computer experiments, which suggests that the security level is about the level mentioned above. But our experiment is on a much smaller scale $(n = 27, r = 3)$. There is some evidence suggesting that D should be roughly $r(r-1)/2$, when n is much bigger than r. According to such an estimate, the complexity of the attack is bigger than 2^{100}. However this formula is not a proven formula, but a conjecture, and it is an open question to find a precise formula of D for the PMI system in terms of both n and r, which then will tell us how we should choose r given n to ensure the desired security.

From, the argument, we believe (not proven) that, with all the known attack methods, the security of our system has the attack complexity of 2^{100}.

3.3 Comparison with Other Cryptosystem

In this section, we would like to compare our system with other cryptosystems.

Comparison with RSA In the case of RSA, we know that a minimum of 512 bits is required at this moment to ensure a security level of 2^{100}. First the key size of RSA surely is very small for both private key and public keys, much smaller than the PMI system.

The case of public and private computation complexity is however a different story. In the encryption and decryption process, each needs roughly 512 multiplications of two numbers of 512 bits modulo a number of 512 bits to process a message 512 bits long. Therefore, each bit of information requires two operations of multiplying two numbers with 512 bits modulo a 512 bits long number.

The conclusion is that the public key size for the PMI system is much bigger than for the RSA system, (1M versus 1.5K) which however should not be a problem for any PC. In terms of per bit efficiency, the comparison is between, on one hand, the RSA, which requires two multiplications of two 512 bit number modulo another 512 bit number, on the other hand, the PMI system, which is an operation to solve 17 linear equation over a finite field of size 2^8 with a given multiplication table. Our preliminary test indicates that the PMI is much faster.

However this is based on our own implementation and the assumption of the security of the system. The situation will be different if we have to increase r for security purpose. If we assume that our system is secure, and since the key transmission is only a one time transaction, when substantial use is required, the PMI system could be better than the USA system.

Comparison with Other Multivariable Cryptosystems The implementation of multivariable cryptosystem is for either authentication purpose or encryption purpose.

The main examples of signature schemes are Quartz schemes, which are based on the HFE and Oil-Vinegar method, and the Sflash schemes, which are based on the Matsumoto-Imai Minus Method. Both of them were accepted for submission to the New European Schemes for Signatures, Integrity, and Encryption: IST-1999-12324, and Sflash was accepted in the final selection.

Current multivariable schemes that are still deemed secure and practical for encryption purpose are basically the HFE scheme, and the HFE and Oil-Vinegar schemes.

In terms of a broader point of view, the Matsumoto-Imai Minus-Plus method can also be viewed as a form of perturbation, except that the perturbation is done through taking out components and adding randomly more components. In this case, each component taken out means a one dimensional exhaustive search in the decryption process and if r components are taken out then a search on an r dimensional space is needed. However for our case, if we perturb by an r dimensional space, we basically perform an $r(r+1)/2$ dimensional Minus and then Plus operation, except here the Plus operation is not just to add randomly a set of components, rather a set of "perturbed" components. In this context, we believe our perturbation method is a better choice compared with the Matsumoto-Imai Minus-Plus system.

It is surely possible to modify the PMI system by the Minus method for a signature scheme as well. What we believe is that it will be a more secure but slower scheme. This is because the perturbed map is not bijective as is case for MI, and therefore one might have to go through a random search process during the signature process.

As we explained above, the idea of HFE is to replace the map F by a small degree map

$$F : X \longmapsto \sum_{i,j}^{A} a_{ij} X^{q^i+q^j} + \sum_{i}^{B} b_i X^{q^i} + c,$$

but the degree cannot be too small due to the algebraic attack [KS] and the XL attack. For the case of a 128 bits implementation over F_2, the degree needs to be 2^{11} to ensure security level as in our implementation example. However to solve a degree 2^{11} polynomial equation on the 128 bits field, it needs about $2^{22} \times 128$ computations over the field to solve the equation for the decryption process, which is much slower than our scheme. Therefore, we believe that the PMI is

a better scheme if our claim on the security is right, otherwise some version of HFE mixed with PMI will be even better.

Due to the fact that the HFE and Oil-Vinegar scheme is an "external" perturbation, namely a new set of variables is introduced to perturb the system. However, our recent observation shows that due to the nature of the "external" perturbation, we can extend the algebraic method [KS] from one variable to two variables case. It seems that we can actually use this generalized algebraic method of that in [KS] to purge out the perturbation if the polynomial F for the HFE equation before the perturbation is small. Once this is done, the original algebraic method [KS] and the XL can be used to attack the system again. Therefore, we think the security of the HFE and Oil-Vinegar scheme is based on the security of HFE part not the oil vinegar part [CDF, FJ]. The detail of this work will be given in a separate paper.

4 Discussion

This paper is a suggestion of a new multivariable cryptosystem, the Perturbed Matsumoto-Imai system, the PMI system. This new system is based on a new theoretical idea of "internal" perturbation. The practical scheme we suggest is an implementation of the idea that creates a 136 bits open-key cryptosystem and the key size is big (1M). However the main purpose of this paper is to introduce the theoretical idea of "internal" perturbation, which, we believe, is a very general and applicable idea. Actually our perturbation idea is not just restricted to the MI systems. We realizes actually it may be a much better idea to combine the HFE method with our "internal" perturbation method, rather than the "external" Oil-Vinegar scheme, namely we will perturb the HFE with a small subspace inside and we do not introduce any new variables. The security is improved because of the perturbation and the impossible task to purge out the perturbation. The reason for this is exactly due to the fact that it is internal, which therefore is fully mixed into the system unlike the case of Oil-Vinegar mixing.

The argument about security and efficiency in this paper is based on intuitive and rough ideas and not on strict mathematical arguments. We do not understand why we can do so as well and we believe it is a very interesting problem. Therefore, we plan to perform more computer simulations, which may give some ideas how things really are.

Acknowledgments

We thank the referee for suggestions. We thank Professor Dingfeng Ye and Professor Dieter Schmidt for useful discussions. Our research is partially supported by the Taft Fund at the University of Cincinnati

References

[ACDG] Mehdi-Laurent Akkar, Nicolas T. Courtois, Romain Duteuil, Louis Goubin, *A Fast and Secure Implementation of Sflash* Volume 2567, pp 267-278 Lecture Notes in Computer Science

[C] Nicolas T. Courtois *The Security of Hidden Field Equations (HFE),* Volume 2020, pp 0266 Lecture Notes in Computer Science

[CDF] Nicolas Courtois, Magnus Daum and Patrick Felke, *On the Security of HFE, HFEv- and Quartz,* PKC 2003, LNCS 2567, Springer, pp. 337-350.

[CP] Nicolas Courtois, Jacques Patarin, *About the XL Algorithm over GF(2),* Volume 2612, pp 141-157 Lecture Notes in Computer Science

[CGP] Nicolas Courtois, Louis Goubin, Jacques Patarin, FLASH, a Fast Multivariate Signature Algorithm, Volume 2020, pp 0298 Lecture Notes in Computer Science

[CGP1] Jacques Patarin, Louis Goubin, Nicolas Courtois, *C-+* and HM: Variations around Two Schemes of T. Matsumoto and H. Imai* Volume 1514, pp 0035 Lecture Notes in Computer Science

[CGP2] Jacques Patarin, Nicolas Courtois, Louis Goubin QUARTZ, 128-Bit Long Digital Signatures, Volume 2020, pp 0282 Lecture Notes in Computer Science

[CKPS] Nicolas Courtois, Alexander Klimov, Jacques Patarin, Adi Shamir *Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations,* Volume 1807, pp 0392 Lecture Notes in Computer Science

[Dm] Dickerson, Matthew, *The inverse of an automorphism in polynomial time.* J. Symbolic Comput. 13 (1992), no. 2, 209–220.

[KS] Aviad Kipnis, Adi Shamir, *Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization* Volume 1666, pp 0019, Lecture Notes in Computer Science

[FJ] Jean-Charles Faugère, Antoine Joux, *Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems Using Gröbner Bases* Advances in cryptology – CRYPTO 2003, Dan Boneh (ed.), Volume 2729, Lecture notes in computer Sciences, Springer, New York 2003

[MI] Matsumoto, T., Imai, H., *Public quadratic polynomial-tuples for efficient signature verification and message encryption,* Advances in cryptology – EUROCRYPT '88 (Davos, 1988), 419–453, Lecture Notes in Comput. Sci., 330, Springer, Berlin, 1988.

[P] Patarin, J., *Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt'88.,* Des. Codes Cryptogr. 20 (2000), no. 2, 175–209.

[P1] Patarin, J., *Hidden field equations and isomorphism of polynomials,* Eurocrypto'96, 1996.

[S] Shamir, Adi, *Efficient signature schemes based on birational permutations,* Advances in cryptology – CRYPTO '98 (Santa Barbara, CA, 1998), 257–266, Lecture Notes in Comput. Sci., 1462, Springer, Berlin, 1998.

Efficient, Verifiable Shuffle Decryption and Its Requirement of Unlinkability

Jun Furukawa

NEC Corporation, 4-1-1 Miyazaki, Miyamae, Kawasaki 216-8555, Japan
j-furukawa@ay.jp.nec.com

Abstract. In this paper, we propose a scheme to simultaneously prove the correctness of both shuffling and decryption. Our scheme is the most efficient of all previous schemes, as a total, in proving the correctness of both shuffling and decryption of ElGamal ciphertexts. We also propose a formal definition for the core requirement of Unlinkability in verifiable shuffle-decryption, and then prove that our scheme satisfies this requirement. The proposed definition may be also useful for proving the security of verifiable shuffle-decryption, hybrid mix network, and other mix-nets.

Keywords: Voting, Shuffle, Decryption, Permutation Hiding

1 Introduction

A mix-net [3] scheme is useful in applications, such as voting, which require anonymity. Crucial to a mix-net scheme is the execution of multiple rounds of shuffling and decryption by multiple, independent mixers, so that none of the output decryptions can be linked to any of the input encryptions.

To ensure the correctness of output, it is desirable to achieve the property of universal verifiability. Early studies, such as those by Sako and Kilian [21] and Abe [1], required vast amounts of computation to prove and verify the correctness of a mix-net without sacrificing unlinkability, However, recently proposed schemes [9,8,17,13] were sufficiently efficient and practical. The schemes of [9,8] use the property of permutation matrixes, and the schemes of [17,13] use the fact that polynomials remain invariant under the permutations of their roots. The schemes of [9], [17], and [13] require the respective computation of $18k$, $42k$, and $12k$ modular exponentiations to prove and verify the correctness of a shuffling of k data. The scheme of [8] requires $19k$ modular exponentiations to prove and verify both shuffling and decryption. Groth's scheme [13] is the most efficient.

A result of these recent works is that proving the correctness of decryption now costs as much as proving the correctness of shuffling. Hence, decreasing the cost of proving decryption has also become important in mix-net. The scheme of [8], which was based on the scheme of [9], made it possible to simultaneously prove the correctness of both a shuffling and a decryption; this is more efficient in terms of computation and communication complexity than proving each of these separately.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 319–332, 2004.

However, as is mentioned in [8], the scheme of [9,8] is not a zero-knowledge, and this simultaneously proving technique never yields a zero-knowledge protocol [1]. A simple combination of two zero-knowledge protocols of a verifiable shuffle and of a verifiable decryption also does not yield a zero-knowledge protocol since the intermediate state cannot be simulated. Therefore, a formal definition for the core requirement of unlinkability in verifiable shuffle-decryption, which notion is *weaker* than that of zero-knowledge, is desired.

Such a formal definition will also be useful for considering the security of verifiable mix-net, hybrid mix network, flush mix, and other mix-net [12,14,15,18]. For example, during the decryptions in a hybrid mix network, servers who decrypt ciphertexts generate many extra data that are not themselves encryptions of plain texts (e.g., encrypted secret keys, MAC code, intermediate states, etc.). Hence, even if each component protocol of a hybrid mix network is zero-knowledge, we must confirm that these extra data do not spoil the unlinkability of the total hybrid mix network.

In this paper, we first propose a formal definition for the core requirement of unlinkability in verifiable shuffle-decryption. Next, we propose the most efficient scheme to simultaneously prove the correctness of both shuffling and decryption, which is an improved version of the scheme of [8]. Finally, we prove that the proposed scheme satisfies the proposed requirement.

Our scheme requires roughly $14k$ exponentiations to prove and verify the correctness of both a shuffle and a decryption of k-data, $1344k$ bits of communication, and five rounds. To prove and verify the correctness of both a shuffle and a decryption of k-data with Groth's protocol [13] by using the standard technique of proving the correctness of decryption, we require $15k$ exponentiations, $2528k$ bits of communication, and seven rounds.

Although the security of the schemes of [9] and [8] have never been proven, We are now able to prove that these schemes satisfy the proposed requirement and are secure. Contrary, it is easy to prove that several hybrid mix-network which are vulnerable against resending message attack, such as [15], do not satisfy the proposed requirement.

Our paper is organized as follows. Section 2 introduces the model of shuffle-decryption. Section 3 proposes a definition for the requirement of unlinkability in verifiable shuffle decryption. Section 4 proposes a protocol by which we are able to simultaneously prove the correctness of a shuffle-decryption in an efficient way. Section 5 compares the efficiency of our protocol to prior work.

2 Notation and Model

2.1 Notation

Let p, q be two primes s.t. $q|p-1$ and $3 \not| (q-1)$, $\mathbb{G}_q$ be an order q subgroup of $(\mathbb{Z}/p\mathbb{Z})^*$, g_0 be an element of $\mathbb{G}_q$, and k be the number of ElGamal ciphertexts to

[1] Whereas, it is easy to make a perfect (and more efficient) zero-knowledge version of the protocol proposed in [9]. This version is presented in an appendix of [7].

be shuffled, and ℓ be the number of shufflers. Let $x'^{(\lambda)} \in_R (\mathbb{Z}/p\mathbb{Z})^*$ be a private key of λ-th shuffler used for the partial decryption and $y^{(\lambda)} = g_0{}^{x'^{(\lambda)}} \bmod p$ be the corresponding public key. Let $m_0{}^{(\lambda)} = \prod_{\kappa=1}^{\lambda} y^{(\kappa)} \bmod p$ be a public key public key used for the shuffle of λ-th shuffler.

Let $(g_i^{(\ell)}, m_i^{(\ell)}) = (g_0{}^{\bar{r}_i}, m_0{}^{\bar{r}_i} M_i)_{i=1,\ldots,k} \bmod p$ be a tuple of ElGamal ciphertexts to be input ℓ-th shuffler where $\{M_i \in \mathbb{G}_q\}_{i=1,\ldots,k}$ is a set of plain texts to be encrypted, $\{\bar{r}_i \in_R \mathbb{Z}/q\mathbb{Z}\}_{i=1,\ldots,k}$ be uniformly and randomly chosen elements of $\mathbb{Z}/q\mathbb{Z}$. Let $(g_i^{(\lambda)}, m_i^{(\lambda)})_{i=1,\ldots,k}$ be a tuple of ciphertexts to be input to λ-th shuffler, who shuffle them with public key $(g_0, m_0^{(\lambda)})$ and then partially decrypts them with the private key $x'^{(\lambda)}$. The resulting tuple of ciphertexts is $(g_i^{(\lambda-1)}, m_i^{(\lambda-1)})_{i=1,\ldots,k}$ which is passed to $(\lambda-1)$-th shuffler. In the rest of the paper, we only consider λ-th shuffler and omit the index (λ).

Treating the public key g_0, m_0 as if it were an element in a ciphertext vector may be awkward, but it gives a more compact and unified representation to variables. Here, the public key is a set, $\{p, q, g_0, m_0, y\}$. P is a prover who shuffles and decrypts and proves the validity of shuffle and decryption to a verifier V.

The only shuffling we have considered in this paper is that of ElGamal cryptosystem, which is the most elegant candidate cryptosystem used for mix-net. However, extensions of the requirements of unlinkability defined in this paper to other cryptosystems are easy.

2.2 ElGamal Shuffle Decryption

ElGamal shuffling is a procedure that, given k ElGamal ciphertexts $(g_i, m_i)_{i=1,\ldots,k}$, outputs ElGamal ciphertexts

$$(g'_i, m'_i) = (g_0{}^{s_i} g_{\phi^{-1}(i)}, m_0{}^{s_i} m_{\phi^{-1}(i)}) \bmod p \quad i = 1, \cdots, k,$$

where $s_i \in_R \mathbb{Z}/q\mathbb{Z}$ for $i = 1, \cdots, k$ and a permutation of indices $\phi : \{1, \ldots, k\} \to \{i = 1, \ldots, k\}$ are chosen uniformly and randomly.

Shuffling of ElGamal ciphertexts results in the following two important properties:

1. There exists a permutation ϕ s.t. equations $D_x((g'_i, m'_i)) = D_x((g_{\phi^{-1}(i)}, m_{\phi^{-1}(i)}))$ hold for all i. Here, $D_x(\cdot)$ is a decryption algorithm that uses the private key x.
2. As long as the decision Diffie-Hellman problem is difficult to solve, no polynomially bounded algorithm, given only $p, q, g, y, (g_i, m_i), (g'_i, m'_i); i = 1, \cdots, k$, has an advantage over the random-guessing algorithm in guessing any part of permutation ϕ for uniformly and randomly chosen $g_0, m_0, s_i, \bar{r}_i, \phi$.

ElGamal shuffle decryption is a combination procedure of ElGamal shuffling and partial decryption that, given k ElGamal ciphertexts $(g_i, m_i); i = 1, \cdots, k$, outputs ElGamal ciphertexts

$$(g'_i, m'_i) = (g_0{}^{s_i} g_{\phi^{-1}(i)}, g_i'^{-x'} m_0{}^{s_i} m_{\phi^{-1}(i)}) \bmod p \quad i = 1, \cdots, k, \qquad (1)$$

where $s_i \in_R \mathbb{Z}/q\mathbb{Z} i = 1, \cdots, k$ and ϕ are chosen uniformly and randomly. Here, the multiplication by $g_i'^{-x'}$ in the second term has the effect of partial decryption.

A sequence of shuffles-decryptions composes a mix-net[21]. In this paper, we propose a formal definition for the core requirement of unlinkability in this verifiable ElGamal shuffle-decryption, and then we propose an efficient verifiable ElGamal shuffle-decryption.

3 Complete Permutation Hiding

We propose here the notion of *complete permutation hiding* (CPH) as a core requirement of unlinkability in verifiable shuffle-decryption. If a verifiable shuffle-decryption is CPH, honest verifiers will learn nothing new about its permutation from an interaction with a prover in an **overwhelming** number of cases of random tape that a prover has chosen uniformly and randomly, whereas, if the protocol is zero-knowledge, verifiers will learn nothing new in **every** case of the random tape. In other words, we define CPH so that verifiers learn nothing about the permutation in an overwhelming number of cases of common input X_n and witness W_n that the generator G_R (defined below) outputs.

Let I_n be a *set* of domain parameters $1^n, p, q$, where p and q are primes and are the lengths of the polynomial of n, private key $\bar{x}$, plain texts $\{M_i \in \mathbb{G}_q\}_{i=1,\ldots,k}$, and random tape Z_n. Let $enc(U)$ be an *encoding of a probabilistic polynomial time (PPT) Turing machine U* which generates cipher-texts $(g_i, m_i)_{i=1,\ldots,k}$ input to the shuffle-decryption procedure. We assume the existence of a knowledge extractor that can extract $\{\bar{r}_i\}_{i=1,\ldots,k}$ such that $g_0^{\bar{r}_i} = g_i$ from U. This assumption is satisfied if all generators of cipher-texts are imposed to prove the knowledge of $\bar{r}_i$, and such a compulsion prevents an adaptively chosen cipher-text attack.

Definition 1. *Given* $I_n (= \{1^n, p, q, \bar{x} \in \mathbb{Z}/q\mathbb{Z}, \{M_i \in \mathbb{G}_q\}_{i=1,\cdots,k}, Z_n\})$ *and* $enc(U)$, instance Generator G_R *chooses* $g_0 \in_R \mathbb{G}_q, x' \in_R \mathbb{Z}/q\mathbb{Z}$, $\{s_i \in_R \mathbb{Z}/q\mathbb{Z}\}_{i=1,\ldots,k}$, *and a permutation* ϕ *uniformly and randomly and computes;*

$$m_0 = {g_0}^{x'+\bar{x}}, y = {g_0}^{x'} \bmod p$$
$$(g_i, m_i) = U(I_n, g_0, y) \in \mathbb{G}_q \times \mathbb{G}_q$$
$$(g_i', m_i') = ({g_0}^{s_i} g_{\phi^{-1}(i)}, {g_i}^{-x'} {m_0}^{s_i} m_{\phi^{-1}(i)}) \bmod p.$$

G_R *then outputs common input* X_n *and witness* W_n*:*

$$X_n = \{p, q, y, \bar{x}, g_0, m_0, \{(g_i, m_i)\}_{i=1,\cdots,k}, \{(g_i', m_i')\}_{i=1,\cdots,k}\},$$
$$W_n = \{\phi, \{s_i\}_{i=1,\cdots,k}, x'\}.$$

In the above definition, U is a PPT Turing machine that plays the role of (malicious and colluding) players who generate cipher-texts $\{(g_i, m_i)\}$. Although U is determined before the public parameter is generated, it does not lose generality because it has this public parameter as an input. In a case where U realizes honest players, it outputs

$$(g_i, m_i) = (g_0{}^{\bar{r}_i}, M_i m_0{}^{\bar{r}_i}) \bmod p$$

using random numbers $\{\bar{r}_i\}_{i=1,\ldots,k}$ generated from the random tape Z_n.

We say X_n and W_n *satisfy relation R* if the following equations are satisfied:

$$m_0 = g_0{}^{x'+\bar{x}}, y = g_0{}^{x'} \pmod p$$
$$(g'_i, m'_i) = (g_0{}^{s_i} g_{\phi^{-1}(i)}, g_i{}^{-x'} m_0{}^{s_i} m_{\phi^{-1}(i)}) \pmod p.$$

We denote this fact as $(X_n, W_n) \in R$. If there exists a witness W_n for a common input X_n that satisfies $(X_n, W_n) \in R$, common input X_n is a *correct shuffle-decryption.* Generator G_R outputs such a X_n.

Definition 2. *Let $View_V^P(X_n, W_n)$ be V 's* view of an interaction *with P, which is composed of the common input X_n, messages V receives from P, random tape input to V, and messages V sends to P during joint computation employing X_n, where P has auxiliary input W_n s.t., $(X_n, W_n) \in R$. $View_V^P$ is an abbreviation of $View_V^P(X_n, W_n)$.*

We consider the case when a semi-honest verifier may collude with malicious players who encrypt the ciphertexts and other provers who shuffle and decrypt in the same mix-net. Such a verifier and players may obtain partial information regarding the plain texts $\{M_i\}$, private key $\bar{x}$ (the sum of other prover's private keys in the mix-net), random tapes of players, and even a part of the permutation ϕ in addition to $View_V^P$. Moreover, they may obtain the results of other shuffle-decryptions executed by the same prover.

Then it is reasonable to describe this extra information as $H(I_n, enc(U), X_n, \phi)$ and input cipher-texts generated by the malicious player as $U(I_n, g_0, y)$ using PPT Turing machines $H(\cdot)$ and $U(\cdot)$. Note that $\{s_i\}$ are not included in the arguments of H, because we consider only the case where the prover never reveals these values to any one and the case where the prover never uses the same $\{s_i\}$ for other shuffle-decryptions.

Even though the verifier and the players may obtain the results of other shuffle-decryptions executed by the same prover who uses x', we do not include x' into the input of U and H. Instead, we assume that there exists a PPT Turing machine K such that the distribution of $View_V^P$ for such H and U and that of $K(I_n, g_0, y, enc(U), \phi)$ are the same. We denote this as $View_V^P \approx K(I_n, g_0, y, enc(U), \phi)$. The exclusion of x' is crucial because it enables us to consider the security of shuffle-decryption over the distribution of X_n i.e., of x'.

We describe information about the permutation ϕ that verifiers try to learn as $f(\phi)$ using PPT Turing machine f. This description can be justified because the expression $f(\phi)$ is sufficient to express any bit of ϕ and any kind of check sum for ϕ.

Now we can say that a verifiable shuffle-decryption protocol hides its permutations completely with respect to G_R - i.e., CPH occurs - if there exists a probabilistic polynomial time algorithm E'^E (which has black box access to E)

with inputs X_n and $H(I_n, enc(U), X_n, \phi)$ that suffers no disadvantage with respect to learning anything about the permutations compared to any probabilistic polynomial time verifier E having input $View_V^P$ and $H(I_n, enc(U), X_n, \phi)$. This leads to,

Definition 3. (complete permutation hiding) *A verifiable shuffle decryption protocol* (P, V, G_R) *achieves* complete permutation hiding *if*

$$\begin{aligned}
&\exists_{E'^E} \forall_E \forall_H \forall_f \forall_U \forall_c > 0 \exists_N \forall_n > N \forall_{I_n} \\
&\Pr[E(View_V^P, H(I_n, enc(U), X_n, \phi)) = f(\phi)] \\
&< \Pr[E'^E(X_n, H(I_n, enc(U), X_n, \phi)) = f(\phi)] + \frac{1}{n^c}, \qquad (2) \\
&and \\
&\exists_K \ View_V^P \approx K(I_n, g_0, y, enc(U), \phi)
\end{aligned}$$

where E', E, H, f, U, K *are PPT Turing machine. The left probability in Eq. (2) is taken over the distribution of the random tapes input to* G_R, [2] *P, V, H, and E. The right probability in Eq.(2) is taken over the distribution of the random tapes input to* G_R, *H,* E', *and E.* E' *may use E as a black box.*

If the verifiable shuffle-decryption protocol is CPH, we can say that for **every** input ciphertexts set $\{(g_i, m_i)\}$ and its corresponding output ciphertexts set $\{(g'_i, m'_i)\}$, whatever an honest verifier who has partial information $(H(I_n, enc(U), X_n, \phi))$ about the common input (X_n), can learn about the permutation (ϕ) after interacting with a prover, can also - in an **overwhelming** number of cases of common input (X_n)- be efficiently computed from that common input (X_n) and that partial information $(H(I_n, enc(U), X_n, \phi))$ alone using a PPT Turing machine E' without interaction with the prover as long as the prover has chosen the private key x', permutation ϕ, and random numbers $\{s_i\}$ uniformly and randomly.

Note that we are considering the case even where malicious and colluding players, who have the results of other shuffle-decryptions with the same x', are engaged in generating $\{(g_i, m_i)\}$ of common input. Hence, CPH guarantees security when shuffle-decryptions with the same private key are repeatedly executed[3].

Extensions of the proposed definition for requirements regarding unlinkability to other mix-net systems (in the sense that verifiers can learn nothing new about the permutation in an overwhelming number of cases of common input) are easy. Hence, extended-CPHs may be suitable measures of the security of verifiable

[2] Since the probability is taken over a distribution containing x', we have excluded any adversary who knows x'.

[3] The definition of shuffle-decryption stated in [8] is "No polynomially bounded adversary can compute any partial information of the permutation from the protocol". Unlike our new definition, this definition does not mention the case where the verifier has already obtained partial information before the protocol begins and where the shuffle-decryptions with the same private key are repeatedly executed. These cases seem to occur quite often.

shuffle-decryptions, verifiable mix-nets, verifiable hybrid mix networks, and other verifiable mix-nets.

4 Proposed Verifiable Shuffle Decryption

In this section, we propose a CPH verifiable shuffle decryption scheme, which is *special* in the sense that the verifier's random tape is identical to its challenge. The proposed protocol is the most efficient of all previous schemes, as a total, to prove the correctness of both shuffling and decryption of ElGamal ciphertexts. The scheme requires five rounds.

4.1 Permutation Matrix

Our scheme uses the property of permutation matrix defined below.

Definition 4. *Let q be a prime. A matrix $(A_{ij})_{i,j=1,\cdots,k}$ is a* permutation matrix over $\mathbb{Z}/q\mathbb{Z}$ *if it satisfies*

$$A_{ij} = \begin{cases} 1 \bmod q & \text{if } \phi(i) = j \\ 0 \bmod q & \text{otherwise} \end{cases}$$

for a permutation function $\phi : \{1, \ldots, k\} \to \{1, \ldots, k\}$.

Using a permutation matrix (A_{ji}), which corresponds to a permutation ϕ, we find that Eq. (1) can be expressed as

$$(g_i', m_i') = (g_0{}^{s_i} \prod_{j=1}^{k} g_j^{A_{ji}}, g_i'^{-x'} m_0^{s_i} \prod_{j=1}^{k} m_j^{A_{ji}}) \bmod p. \tag{3}$$

Therefore, proving the correctness of the shuffle is equivalent to proving the existence of a $x' \in \mathbb{Z}/q\mathbb{Z}$, an $s_i \in \mathbb{Z}/q\mathbb{Z}$ for $i = 1, \ldots, k$ and a permutation matrix $(A_{ji})_{i,j=1,\ldots,k}$ which satisfy Eq. (3).

The following theorem is the key to constructing the proposed protocol.

Theorem 1. ([9] Theorem 1) *Let q be a prime. A matrix $(A_{ij})_{i,j=1,\ldots,n}$ is a permutation matrix over $\mathbb{Z}/q\mathbb{Z} \Leftrightarrow$*

$$\sum_{h=1}^{n} A_{hi} A_{hj} A_{hk} = \delta_{ijk} \triangleq \begin{cases} 1 \pmod q & \text{if } i = j = k \\ 0 \pmod q & \text{if otherwise} \end{cases} \quad \text{and} \tag{4}$$

$$\sum_{h=1}^{n} A_{hi} A_{hj} = \delta_{ij} \triangleq \begin{cases} 1 \pmod q, & \text{if } i = j \\ 0 \pmod q, & \text{if } i \neq j \end{cases} \tag{5}$$

for all i, j, and k.

Proof. See the proof of Theorem 1 in [9] or appendix of [7].

Theorem 2. *For* $3 \not| (q-1)$, *a matrix* $(A_{ij})_{i,j=1,\ldots,n}$ *is a permutation matrix over* $\mathbb{Z}/q\mathbb{Z} \Leftrightarrow$ *Eq. (4) holds.*

Proof. ($\Rightarrow$) is trivial. ($\Leftarrow$); From the proof of Theorem 1 in [9], if Eq.(4) holds, then there is only one non-zero element e_i in the $i-th$ row and it must satisfies $e_i^3 = 1 \bmod q$. Because $3 \not| (q-1)$ implies that 1 is the only cubic root of 1 in $\mathbb{Z}/q\mathbb{Z}$, e_i must be 1. Therefore, matrix $(A_{ij})_{i,j=1,\ldots,n}$ is a permutation matrix over $\mathbb{Z}/q\mathbb{Z}$.

The soundness of our scheme depends directly on Theorem 2.

4.2 Protocol Structure and Tricks for Efficiency

The verifiable shuffle decryption protocol we will propose in this section is almost the same as the scheme proposed in [8]. The proposed scheme and the scheme of [8] are roughly composed of four proofs. These are, (i) generation of $\{f'_i\}_{i=1,\cdots,k}$ and a proof of knowledge of s_i and (A_{ji}) that satisfy

$$f'_i = f_0^{s_i} \prod_{j=1}^{k} f_j^{A_{ji}} \bmod p \quad i = 1, \cdots, k, \tag{6}$$

for uniformly and randomly chosen $f_\mu \in_R \mathbb{G}_q; (\mu = 0, \ldots, k)$, (ii) proof that (A_{ji}) whose knowledge proved in (i) is a permutation matrix (using Theorem 1 or 2), (iii) proof that s_i and (A_{ji}) whose knowledge proved in (i) also satisfies Eq. (3), and (iv) proof of knowledge of the decryption key.

In Proof (ii), there are commitment, challenge, and response phase. The main difference between our scheme and the scheme of [8] is that we have introduced the values f_{-2}, f_{-1} in the proposed scheme. Because of these values f'_is in the commitment are modified from $f'_i = f_0^{A_{0i}} f_{\phi^{-1}(i)}$ to $f'_i = f_{-2}^{A_{-2i}} f_{-1}^{A_{-1i}} f_0^{A_{0i}} f_{\phi^{-1}(i)}$. As a results, we have more redundancy (A_{-2i}, A_{-1i}) to generate the f'_i. Then we adjusted A_{-2i}, A_{-1i} so that some values in the commitment to be zero, which decreased the number of terms in checking equations in the response phase [4]. Another difference between them is that the proposed scheme adopts the prime q such that $3 \not| q-1$. Because of this, verifiers do not need to confirm that Equation (5) holds [5] any more. The other difference between them is with respect to the verification of Eq. (9). A verifying that Eq. (9) holds, is equivalent to verifying equations

$$\prod_{\nu=-2}^{k} f_\nu{}^{r_\nu} = f'_0 \prod_{i=1}^{k} f_i'^{c_i} \ , \quad \prod_{\nu=-2}^{k} f_\nu{}^{r'_\nu} = \tilde{f}'_0 \prod_{i=1}^{k} f_i'^{c_i{}^2} \pmod p$$

hold, where the former is more efficient [6].

[4] The equation related to Equation 12 is the 7-th equation in the verification phase of the scheme of [8]. We can see that terms quadratic and linear to the challenge are disappeared in the proposed protocol.

[5] 8-th equation in the verification phase of the scheme of [8].

[6] r'_μ plays the role of λ' in [8].

4.3 Proposed Protocol

We now describe our verifiable ElGamal shuffle decryption and our scheme. Let public parameters p, q, g_0, y, m_0 and private key x' be as described before. We assume another public key $F_n \triangleq \{f_\nu \in_R \mathbb{G}_q\}_{\nu=-2,\cdots,k}$ are $k+3$ $\mathbb{G}_q$ elements that are uniformly and randomly generated so that neither P nor V can generate non-trivial integers $a, \{a_\nu\}_{\nu=-2,\cdots,k}$ satisfying $g_0^a \prod_{\nu=-2}^{k} f_\nu{}^{a_\nu} = 1 \pmod p$ with non-negligible probability.

ElGamal Shuffle Decryption P uniformly and randomly chooses $A_{0i} \in_R \mathbb{Z}/q\mathbb{Z}$ for $i = 1, \cdots, k$ and a permutation matrix $(A_{ji})_{i,j=1,\cdots,k}$ and then shuffles and decrypts k ElGamal ciphertexts $\{(g_i, m_i)\}_{i=1,\cdots,k}$ to $\{(g_i', m_i')\}_{i=1,\cdots,k}$ as

$$\begin{aligned}(g_i', m_i') &= (g_0{}^{A_{0i}} g_{\phi^{-1}(i)}, g_i'{}^{-x'} m_0{}^{A_{0i}} m_{\phi^{-1}(i)}) \bmod p \\ &= (\prod_{\nu=0}^{k} g_\nu{}^{A_{\nu i}}, g_i'{}^{-x'} \prod_{\nu=0}^{k} m_\nu{}^{A_{\nu i}}) \bmod p. \end{aligned} \tag{7}$$

In our protocol, the witness W_n is a set $\{x', (A_{ji})_{i,j=1,\cdots,k}, \{A_{0i}\}_{i=1,\cdots,k}\}$, and the common input X_n is a set $\{p, q, g_0, y, m_0, F_n, (g_i, m_i)_{i=1,\cdots,k}, (g_i', m_i')_{i=1,\cdots,k}\}$. P is given X_n and W_n, and V is given X_n.

Proving a Shuffle Decryption Commitment-1: P uniformly and randomly chooses $\{A_{\nu 0}, A'_\nu \in_R \mathbb{Z}/q\mathbb{Z}\}_{\nu=-2,\cdots,k}$ and then computes:

$$A_{-1i} = \sum_{j=1}^{k} 3A_{j0}A_{ji} \bmod q \ , \quad A_{-2i} = \sum_{j=1}^{k} 3A_{j0}{}^2 A_{ji} \bmod q \quad i = 1, \cdots, k$$

$$f'_\mu = \prod_{\nu=-2}^{k} f_\nu{}^{A_{\nu\mu}} \bmod p \quad \mu = 0, \cdots, k$$

$$\tilde{f}'_0 = \prod_{\nu=-2}^{k} f_\nu{}^{A'_\nu} \bmod p \ , \quad g'_0 = \prod_{\nu=0}^{k} g_\nu{}^{A_{\nu 0}} \bmod p$$

$$m'_0 = \prod_{\nu=0}^{k} m_\nu{}^{A_{\nu 0}} \bmod p \ , \quad w = \sum_{j=1}^{k} A_{j0}{}^3 - A_{-20} - A'_{-1} \bmod q$$

Then, P sends $g'_0, m'_0, w, \tilde{f}'_0, \{f'_\mu\}_{\mu=0,\cdots,k}$ to V as a commitment.

Challenge-1: V uniformly and randomly chooses $\{c_i\}_{i=1,\cdots,k}$ from $\mathbb{Z}/q\mathbb{Z}$ and sends it to P.

Response-1: P sends V the following response:

$$r_\nu = \sum_{\mu=0}^{k} A_{\nu\mu}c_\mu \bmod q \ , \quad r'_\nu = \sum_{i=1}^{k} A_{\nu i}c_i{}^2 + A'_\nu \bmod q \quad \nu = -2, \cdots, k$$

where $c_0 = 1 \bmod p$.

Commitment-2: P then computes

$$\zeta = \prod_{i=1}^{k} g_i'^{c_i} \bmod p.$$

P uniformly and randomly chooses $\beta \in_R \mathbb{Z}/q\mathbb{Z}$, computes the following commitment, and sends it to V:

$$\eta = \zeta^{x'} \bmod p \ , \quad \eta' = \zeta^{\beta} \bmod p$$
$$y' = g_0{}^{\beta} \bmod p. \tag{8}$$

Challenge-2: V uniformly and randomly chooses c' from $\mathbb{Z}/q\mathbb{Z}$ and sends it to P.
Response-2: P sends V the following response: $r' = c'x' + \beta \bmod q$
Verification: V computes

$$\zeta = \prod_{i=1}^{k} g_i'^{c_i} \bmod p.$$

V accepts the shuffle if the following equations hold for a uniformly and randomly generated $\alpha \in_R \mathbb{Z}/q\mathbb{Z}$:

$$\prod_{\nu=-2}^{k} f_\nu{}^{r_\nu + \alpha r'_\nu} = f'_0 \tilde{f}'_0{}^{\alpha} \prod_{i=1}^{k} f_i'^{c_i + \alpha c_i{}^2} \pmod p \tag{9}$$

$$\prod_{\nu=0}^{k} g_\nu{}^{r_\nu} = \zeta g'_0 \pmod p \tag{10}$$

$$\prod_{\nu=0}^{k} m_\nu{}^{r_\nu} = \eta \prod_{\mu=0}^{k} m'_\mu{}^{c_\mu} \pmod p \tag{11}$$

$$\sum_{j=1}^{k} (r_j^3 - c_j^3) = r_{-2} + r'_{-1} + w \pmod q \tag{12}$$

$$g_0{}^{r'} = y^{c'} y' \pmod p \tag{13}$$

$$\zeta^{r'} = \eta^{c'} \eta' \pmod p \tag{14}$$

The view $View_V^P(X_n, W_n)$ of this protocol is

$$p, q, y, g_0, m_0, \{(g_i, m_i)\}_{i=1,\ldots,k}, \{(g'_i, m'_i)\}_{i=1,\ldots,k},$$
$$\{f_\nu\}_{\nu=-2,\ldots,k}, f'_0, \{f'_i\}_{i=1,\ldots,k}, \tilde{f}'_0, g'_0, m'_0, w, \{c_i\}_{i=1,\ldots,k},$$
$$\{r_\nu\}_{\nu=-2,\ldots,k}, \{r'_\nu\}_{\nu=-2,\ldots,k}, \eta, \eta', y', c', r'.$$

4.4 Properties of the Proposed Scheme

Theorem 3. *The protocol is complete.*

Theorem 4. *The protocol is special sound as long as the discrete logarithm problem is difficult to solve.*

Theorem 3 and 4 can be proved along the lines with [9]. Proof are given in the appendix of [7].

Theorem 5. *If the decision Diffie-Hellman problem is difficult to solve, the verifiable shuffle-decryption protocol* (P, V, G_R) *is special complete-permutation-hiding.*

Proof. The proof is given in the appendix of [7].

4.5 Threshold Decryption

Although it is possible to achieve threshold decryption with the proposed protocol, it does not work as well as ordinary threshold decryption. If we assume that only honest shufflers participate in the shuffle-decryption protocol, there is no disadvantage when using our protocol. However, if a malicious shuffler quits decryption after some other shufflers have finished their decryptions, our protocol gets into trouble.

Suppose we are decrypting or shuffle-decrypting k ElGamal cipher-texts, λ shufflers have finished their partial decryptions, and one shuffler quits its decryption procedure. In the ordinary threshold decryption protocol, the rest of the shufflers and one substituting (new) shuffler are able to continue the threshold decryption protocol only with little modification. However, computation of $k\lambda$ extra modular exponentiations is required to complete the decryption, and the verifier must compute $k\lambda$ extra modular exponentiations to verify the correctness of the decryption.

In our protocol, shufflers that have finished their partial decryptions need to help other players complete the protocol. Each of the shufflers needs to compute k modular exponentiations to modify the cipher-texts that are already shuffle-decrypted by λ shufflers. Each of them needs to prove the correctness of the above computation which requires another computation of k modular exponentiations. Moreover, the verifier needs to compute an extra $2k\lambda$ modular exponentiations to verify the correctness of the protocol.

5 Efficiency

In this section, we compare the efficiency of the proposed protocol described in Section 4 to (FS) the protocol proposed in [9], (FMMOS) the protocol proposed in [8], and (Groth) the protocol proposed in [13]. We have assumed the lengths

of p and q to be 1024 and 160. We have denoted the protocol in Section 4 as (proposed).

Let us first compare them, in Table 1, by the number of exponentiations used in each protocol when the number of ciphertexts is k. "shuffle P" and "shuffle V" denote the number of exponentiations required for P and V to prove and verify a shuffle. "shuffle-decrypt P" and "shuffle-decrypt V" denote the number of exponentiations required for P and V to prove and verify a shuffle-decryption. The numbers for (*FS*), (*FMMOS*), and (*Groth*) are those required to prove a shuffle-decryption in a standard technique

If we adopt the computation tools described in [16], such as the simultaneous multiple exponentiation algorithm and the fixed-base comb method, the number of exponentiations can be heuristically reduced. We estimated that multiple exponentiations cost a 1/3 and fixed-base comb method costs 1/12 (when the number of ciphertexts is large) of that of single exponentiation. Estimates done in this way are in Table 2. Here, "shuffle P", "shuffle V" , "shuffle-decrypt P", and "shuffle-decrypt V" denote the same items as in Table 1.

Table 3 lists the number of communication bits and number of rounds required for protocols. "shuffle" denotes the number of communication bits used when proving a shuffle, "shuffle-decrypt" denotes the number of communication bits used when proving a shuffle-decryption, and "rounds" denotes the number of rounds required for protocols. The numbers for (*FS*), (*FMMOS*), and (*Groth*) include intermediate state data bits, i.e., those of shuffled data.

	(FS)	(FMMOS)	(Groth)	(proposed)
shuffle P	$8k$		$6k$	
shuffle V	$10k$		$6k$	
shuffle-decrypt P	$(9k)$	$9k$	$(7k)$	$8k$
shuffle-decrypt V	$(12k)$	$10k$	$(8k)$	$6k$

Table 1. Numbers of exponentiations required in each protocol

	(FS)	(FMMOS)	(Groth)	(proposed)
shuffle P	$1.4k$		$1.75k$	
shuffle V	$3.3k$		$1.75k$	
shuffle-decrypt P	$(2.4k)$	$1.75k$	$(2.75k)$	$1.9k$
shuffle-decrypt V	$(4.5k)$	$3.3k$	$(3k)$	$2k$

Table 2. Cost of computation required in each protocol

Our protocol and the protocols of [9,8] require a rather long public parameter F_n. Although the protocol of [13] also requires such a parameter, it can be reduced greatly at the cost of increasing the amount of both computation and communication.

	(FS)	(FMMOS)	(Groth)	(proposed)
shuffle	$5044k$		$1184k$	
shuffle-decrypt	$(6388k)$	$5044k$	$(2528k)$	$1344k$
rounds	3	5	7	5

Table 3. Communication bits required in each protocols

From Tables 2 and 3, we can conclude that computational complexity with our proposed protocol represents a 32% improvement in efficiency over that of (Groth)[13], while communication complexity improves by 47%. Our protocol require two rounds less than that of Groth's [13].

6 Conclusion

In this paper, I have proposed formal definition for the core requirement of Unlinkability in verifiable shuffle-decryption. I have also presented a novel method of simultaneously proving both the correctness of both a shuffle and a decryption, and then have proved its security and demonstrated its superior efficiency over that of [13] and [8].

Acknowledgments

The author would like to thank Hiroaki Anada and Satoshi Obana for many helpful discussions.

References

1. M. Abe, *Mix-Networks on Permutation Networks,* Advances in Cryptology — ASIACRYPT '99, LNCS 1716, pp. 258-273, Springer-Verlag, (1999).
2. S. Brands, *An Efficient Off-line Electronic Cash System Based On The Representation Problem,* CWI Technical Report CS-R9323, (1993).
3. D. Chaum, *Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms,* Communications of the ACM, Vol.24, No.2, pp. 84-88, (1981).
4. R. Cramer, I. Damgård, and B. Schoenmakers, *Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols,* Crypto '94, LNCS 839, pp. 174-187, (1994).
5. R. Cramer and V. Shoup, *A practical key cryptosystem provably secure against adaptive chosen ciphertext attack,* Advances in Cryptology — Crypto '98, LNCS 1462, pp. 13-25, (1998).
6. A. Fiat and A. Shamir. *How to prove yourself: Practical solutions to identification and signature problems,* Advances in Cryptology — CRYPTO '86, LNCS 263, pp. 186–194, (1986).
7. J. Furukawa, *Efficient, Verifiable Shuffle Decryption and Its Requirement of Unlinkability (with appendixes),* Available online, http://eprint.iacr.org, or from the author via e-mail.

8. J. Furukawa, K. Mori, S. Obana, and K. Sako, *An Implementation of a Universally Verifiable Electronic Voting Scheme based on Shuffling,* Financial Cryptography 2002.
9. J. Furukawa and K. Sako, *An Efficient Scheme for Proving a Shuffle,* Advances in Cryptology — CRYPTO 2001, LNCS 2139 pp. 368-387 (2001).
10. O. Goldreich, *A Uniform-Complexity Treatment of Encryption and Zero-Knowledge,* Journal of Cryptology, Vol. 6, pp. 21-53, (1993).
11. S. Goldwasser and S. Micali, *Probabilistic Encryption,* JCSS, Vol. 28, No. 2, pp. 270-299, (1984).
12. P. Golle, S. Zhong, D. Boneh, M. Jakobsson, and A. Juels, *Optimistic mixing for exit-polls,* Asiacrypt 2002, LNCS 2501, pp. 451-465 (2002)
13. J. Groth, *A Verifiable Secret Shuffle of Holomorphic Encryptions,* Public Key Cryptography – PKC 2003, LNCS 2567 pp. 145-160 (2003)
14. M. Jakobsson, *A practical mix,* Eurocrypt '98, LNCS 1403, pp. 448-461 (1998)
15. A. Juels and M. Jakobsson, *An optimally robust hybrid mix network,* Proc. of the 20th annual ACM Symposium on Principles of Distributed Computation, 2001
16. A. Menezes, C. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography,* CRC Press, pp. 617-627, (1997).
17. C.A. Neff, *A Verifiable Secret Shuffle and its Application to E-Voting,* ACMCCS 01 pp. 116-125 (2001).
18. M. Ohkubo and M .Abe, *A length-invariant hybrid mix* Asiacrypt 2000, LNCS 1976, pp. 178-191 (2000)
19. W. Ogata, K. Kurosawa, K. Sako, and K. Takatani, *Fault tolerant anonymous channel,* ICICS, LNCS 1334, pp. 440-444 (1997).
20. K. Sako, *Electronic voting schemes allowing open objection to the tally,* Transactions of IEICE, Vol. E77-A No. 1, Jan. (1994).
21. K. Sako and J. Kilian, *Receipt-free mix-type voting scheme –A practical solution to the implementation of voting booth,* Eurocrypt '95, LNCS 921, pp. 393-403 (1995).
22. C. P. Schnorr, *Efficient signature generation by smart cards,* Journal of Cryptology, 4, pp. 161–174, (1991).

A Point Compression Method for Elliptic Curves Defined over $GF(2^n)$

Brian King

Purdue School of Engineering
Indiana Univ. Purdue Univ. at Indianapolis
briking@iupui.edu

Abstract. Here we describe new tools to be used in fields of the form $Gf(2^n)$, that help describe properties of elliptic curves defined over $GF(2^n)$. Further, utilizing these tools we describe a new elliptic curve point compression method, which provides the most efficient use of bandwidth whenever the elliptic curve is defined by $y^2 + xy = x^3 + a_2x^2 + a_6$ and the *trace* of a_2 is zero.

1 Introduction

In [5,9], Koblitz and Miller independently proposed to use elliptic curves over a finite field to implement cryptographic primitives. The benefits for utilizing elliptic curves as a public key primitive are well recognized: smaller bandwidth, fast key exchange and fast signature generation.

The focus of this paper will be with elliptic curves E defined over a field of the form $GF(2^n)$. In particular our contribution will be the development of new tools to be used in $GF(2^n)$ that help describe elliptic curve properties, as well as we develop a new method for point compression, which is the most efficient point compression described so far.[1] Our result answers a question that Seroussi raised in [12]. Here Seroussi stated that it may be possible to improve on his point compression algorithm but that no known efficient method existed. In addition to the point compression method we provide additional results which were derived from the tools developed for the point compression method. Integral to our work is method of *halving a point.*

2 Background Mathematics-Binary Fields $GF(2^n)$ and Elliptic Curves

2.1 The Trace Operator in $GF(2^n)$

The trace function, denoted by Tr, is a homomorphic mapping[2] of $GF(2^n)$ onto $\{0,1\}$. The trace of an element $\alpha \in GF(2^n)$, denoted by $Tr(\alpha)$ can be

[1] Point compression provides an improvement on bandwidth.
[2] $Tr(\alpha + \beta) = Tr(\alpha) + Tr(\beta)$.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 333–345, 2004.

computed (see [15]) as $Tr(\alpha) = \sum_{i=0}^{m-1} \alpha^{2^i}$. (In reality, the trace function can be computed *extremely efficiently,* see *Table 2* in the appendix.) For more information concerning the $Tr()$ operator and its importance see [7]. It can be shown that $Tr()$ is a linear operator which returns a 0 or a 1 and satisfies that $Tr(\alpha^2) = Tr(\alpha)$. In $GF(2^n)$, where n is odd (which is true for all binary fields that we are interested in), then $Tr(1) = 1$ (this can easily be derived given the above equation). Consequently for all $\alpha \in GF(2^n)$ with $Tr(\alpha) = 0$ we have $Tr(\alpha + 1) = 1$ and vice versa. For a given $b \in GF(2^n)$, the quadratic equation $\lambda^2 + \lambda = b$ in $GF(2^n)$ has a solution if and only if $Tr(b) = 0$ [7]. Observe that if λ is a solution to the above quadratic equation, then $\lambda + 1$ is also a solution, and $Tr(\lambda + 1) = Tr(\lambda) + 1$. Hence whenever n is odd, which we always will assume, for each solvable quadratic equation there is a solution with trace 1 and a solution with trace 0.

2.2 Elliptic Curve Operation

For the finite field $GF(2^n)$, the standard equation or Weierstrass equation for a non supersingular elliptic curve is:

$$y^2 + xy = x^3 + a_2x^2 + a_6 \tag{1}$$

where $a_2, a_6 \in GF(2^n)$, $a_6 \neq 0$. The points $P = (x, y)$, where $x, y \in GF(2^n)$, that satisfy the equation, together with the point $\mathcal{O}$, called the point of infinity, form an additive abelian group E_{a_2,a_6}. Here addition in E_{a_2,a_6} is defined by: for all $P \in E_{a_2,a_6}$

- $P + \mathcal{O} = P$,
- for $P = (x, y) \neq \mathcal{O}$, $-P = (x, x + y)$
- and for all $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$, both not equal to the identity and $P_1 \neq -P_2$, $P_1 + P_2 = P_3 = (x_3, y_3)$ where $x_3, y_3 \in GF(2^n)$ and satisfy:

$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a_2$$

and

$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1$$

where $\lambda = \frac{y_1+y_2}{x_1+x_2}$ if $P_1 \neq P_2$ and $\lambda = x_1 + \frac{y_1}{x_1}$ for $P_1 = P_2$.

As stated earlier, the elliptic curve E_{a_2,a_6} is given by the equation $y^2 + xy = x^3 + a_2x^2 + a_6$. If $(x, y) \in E_{a_2,a_6}$ and $x \neq 0$ then $\frac{y^2}{x^2} + \frac{y}{x} = x + a_2 + \frac{a_6}{x^2}$. By making the substitution $z = \frac{y}{x}$ we see that $z^2 + z = x + a_2 + \frac{a_6}{x^2}$. Since this quadratic equation is solvable, we see that $Tr(x + a_2 + \frac{a_6}{x^2}) = 0$. Observe that if β satisfies that $Tr(\beta + a_2 + \frac{a_6}{\beta^2}) = 0$ then there exists a z such that $z^2 + z = \beta + a_2 + \frac{a_6}{\beta^2}$. By setting $y = \beta \cdot z$ we see that $y^2 + \beta y = \beta^3 + a_2\beta^2 + a_6$. Hence $(\beta, y) \in E_{a_2,a_6}$. And so the condition that a nonzero field element β satisfies $Tr(\beta + a_2 + \frac{a_6}{\beta^2}) = 0$ is both a necessary and sufficient condition to determine it the element is the x-coordinate of a point on E_{a_2,a_6}.

In a cryptographic application, the elliptic curve will be selected so that E_{a_2,a_6} will contain a large subgroup of prime order. The cryptographically relevant points will belong to this subgroup of large prime order.

2.3 Point Compression Algorithms

In [15] an algorithm for point compression is described. We summarize it as follows. For a party to send a cryptographically relevant elliptic curve point P they need to send an ordered pair. However, rather than sending an ordered pair it is possible to send the x coordinate and one-bit of information. The corresponding y coordinate can be computed using x and this one-bit. This is because, by equation (1) we have $\frac{y^2}{x^2} + \frac{y}{x} = x + a_2 + \frac{a_6}{x^2}$. The problem is that there are two solutions to this equation, one solution has trace 1 and the other solution has trace 0. Consequently the only information concerning y needed to transmitted by the sender is the trace of trace of $\frac{y}{x}$. So if we are given x we can solve for a λ which satisfies $\lambda^2 + \lambda = x + a_2 + \frac{a_6}{x^2}$. One can determine y from λ, x and this one-bit. This method has been standardized in [16,15] and has been patented. The result is that this method requires $n + 1$ bits to transmit a point on an elliptic curve defined over $GF(2^n)$.

In [2], Seroussi described an even more efficient point compression algorithm. Suppose that $(x_2, y_2) \in E$. Then $Tr(x_2 + a_2 + \frac{a_6}{x_2^2}) = 0$. Again, we assume that (x_2, y_2) is a cryptographically relevant point, that is, it is a point of prime order p. Since (x_2, y_2) is of prime order, it is the double of some point (x_1, y_1). Seroussi, in [2], demonstrated that this implies that $Tr(x_2) = Tr(a_2)$. For completeness (as well as to demonstrate tools that we utilize later) we recreate it here. Suppose (x_2, y_2) is the double of some point $(x_1, y_1) \in E$. Thus $x_2 = x_1^2 + \frac{a_6}{x_1^2}$. Since $(x_1, y_1) \in E$ we have $Tr(x_1 + a_2 + \frac{a_6}{x_1^2}) = 0$. Further since $Tr(x^2) = Tr(x)$ we have

$$Tr(x_1 + a_2 + \frac{a_6}{x_1^2}) = Tr(x_1^2 + a_2 + \frac{a_6}{x_1^2}) = Tr(x_2 + a_2) = 0.$$

Therefore $Tr(x_2) = Tr(a_2)$. It was this property that Seroussi exploits in his compression algorithm. Let $Q = (x_2, y_2)$ be the cryptographically relevant point on the curve E. Consequently Q will belong to a subgroup of prime order and so Q is the double of some point P. Thus $Tr(x_2) = Tr(a_2)$. Given a field element $z = (\zeta_{n-1}, \ldots, \zeta_1, \zeta_0)$ in $GF(2^n)$, it can be represented by n bits. At least one of the bits is used to compute trace, let i denote the smallest index such that ζ_i is used to compute trace (note that it is very likely that $i = 0$), see *Table 2* for examples on how to efficiently compute the trace for the binary fields used in the NIST list of elliptic curves. Suppose $x_2 = (\xi_{n-1}, \xi_{n-2}, \ldots, \xi_1, \xi_0)$. Then to transmit the x-coordinate x_2 we only need to send $n - 1$ bits, since we can transmit $(\xi_{n-1}, \xi_{n-2}, \ldots \xi_{i+1}, \xi_{i-1}, \ldots, \xi_1, \xi_0)$. Now the receiver knows the curve and all of its parameters, thus they know i. Further, the receiver knows that x_2 satisfies $Tr(x_2) = Tr(a_2)$. Consequently the receiver can determine whether ξ_i should be a one or a zero. Once the receiver has x_2, they solve for z such that $z^2 + z = x_2 + a_2 + \frac{a_6}{x_2}$. Then y_2 can be computed by $y_2 = x_2 \cdot z$. The problem again is that there are two solution to this equation in z, one z-solution has trace 1 and the other z-solution has trace 0. Thus the only information needed to transmit y is the trace of the z-value. Hence only one bit needs to be transmitted to communicate y. Therefore Seroussi has demonstrated that only

n bits are needed to be transmit to a receiver a point on the elliptic curve E over $GF(2^n)$.

2.4 Halving a Point

In [6], Knudsen introduced the *halving point coordinates* and the halving a point algorithm. Knudsen introduced the concept of halving a point in elliptic curve over $GF(2^n)$ to compute the scalar multiple kP.[3] Knudsen described how to compute $\frac{1}{2}P$ given a point $P = (x, y) \in E$, where P is a double of some point. At the heart of this computation is the representation of a point. Rather that using the affine coordinates of a point $P = (x, y) \in E$, Knudsen represented P as $P = (x, \lambda_P)$ where $\lambda_P = x + \frac{y}{x}$, which we refer to *halving coordinates.* Observe that given x and λ_P, y can be computed since $y = x(x + \lambda_P)$.
Let $Q = (u, \lambda_Q) = \frac{1}{2}P$ where $P = (x, \lambda_P)$. Then Knudsen [6] demonstrated that the following two equations could determine Q; first λ_Q can be determined by solving:

$$\lambda_Q^2 + \lambda_Q = a_2 + x. \tag{2}$$

Once one solves for λ_Q, u can be determined by computing

$$u = \sqrt{u^2} = \sqrt{x(\lambda_Q + 1) + y} = \sqrt{x(\lambda_Q + \lambda_P + x + 1)}. \tag{3}$$

Observe that $Tr(a_2 + x)$ must equal 0, which is true if and only if P is the double of some point, an observation that is used in both [14,12]. It is trivial to demonstrate that the computed (u, λ_Q) is a "half" of P. Knudsen's algorithm requires one square root, one multiplication, one solve (which is the *halftrace*), and though not illustrated above, *one trace check.* So it will be very efficient.

The primary focus in [6] was with elliptic curves with a cofactor of 2, but Knudsen did not limit his work to only such curves. He provided formulae for the case when the cofactor is 2, as well as when the cofactor is 2^L (where $L > 1$). In [4], an improvement of Knudsen's halving algorithm for curves with a cofactor of 2^L where $L > 1$ was demonstrated.
Integral to our work will be the following algorithms.

SOLVE(s)
  if $Tr(s) \neq 0$
    return **No solution**
  let ζ be an arbitrary solution to
    the equation $w^2 + w = s$
  return ζ

HALF$(P = (x_P, \lambda_P)$
  if $Tr(x_P + a_2) \neq 0$
    return *No half point*
  $\lambda_Q =$ **SOLVE**$(x_P + a_2)$
  $u_Q = \sqrt{x_P(\lambda_Q + \lambda_P + x_P + 1)}$
return (u_Q, λ_Q)

In the **SOLVE** equation, there are two solutions to the quadratic equation. So when ζ is assigned to be an arbitrary solution it meant that any one of the

[3] Independently, Schroepel [11] also developed the method of halving a point to perform cryptographic computations on an elliptic curve.

two solutions is returned. The Theorem described below demonstrates that not only will the **HALF** algorithm produce a half when the input point that can be halved, but that for any input the **HALF** algorithm will produce the correct output.

Theorem 1. *Let $P \in E$ then*
*(i) If Q = **HALF**(P) then $Q \in E$ and $2Q = P$.*
*(ii) If **HALF**(P) returns No half point then for all $Q \in E$, $2Q \neq P$.*

The proof is left as an exercise.

3 Some Observations

Recall that when $(x_2, y_2) \in E$ with $x_2 \neq 0$, we must have $Tr(x_2 + a_2 + \frac{a_6}{x_2^2}) = 0$. Further, if (x_2, y_2) is a double of some point then $Tr(x_2) = Tr(a_2)$. Therefore if (x_2, y_2) is a double of some point then $Tr(\frac{a_6}{x_2^2}) = 0$. This condition can be shown to be both necessary and sufficient to imply that a point is the double of some point in E. The argument is as follows: Suppose $Tr(\frac{a_6}{x_2^2}) = 0$ where $(x_2, y_2) \in E$. Since $Tr(x_2 + a_2 + \frac{a_6}{x_2^2}) = 0$ we see that $Tr(x_2) = Tr(a_2)$. Consider the equation $x^2 + \frac{a_6}{x^2} = x_2$. Observe that if x satisfies this equation then x satisfies $Tr(x^2 + a_2 + \frac{a_6}{x^2}) = Tr(x_2 + a_2) = Tr(x + a_2 + \frac{a_6}{x^2}) = 0$. Thus there exists a y such that $(x, y) \in E$. Now this equation $x^2 + \frac{a_6}{x^2} = x_2$ is solvable, since it reduces to solving $x^4 + x_2 x^2 = a_6$ which is $x_2^2 t^2 + x_2^2 t = a_6$ by letting $x^2 = x_2 t$. This last equation reduces to $t^2 + t = \frac{a_6}{x_2^2}$. Since $\frac{a_6}{x_2^2}$ has trace 0, this is solvable. Once t is found, solve for x by letting $x^2 = x_2 t$ and computing $x = \sqrt{x^2}$.

Consequently the requirement for a point on E to be a double can be solely expressed as a condition existing between x and the parameter a_6. Of course the condition that given x there is some y such that $(x, y) \in E$ can be stated as: $Tr(x + a_2 + \frac{a_6}{x^2}) = 0$. Suppose a_6 is some fixed nonzero field element of $GF(2^n)$, and that x_0 be an arbitrary nonzero field element of $GF(2^n)$ where $Tr(\frac{a_6}{x_0^2}) = 0$. Then x_0 is the x-coordinate for a double of some point for ALL elliptic curves E_{a_2, a_6} which satisfy $Tr(a_2) = Tr(x_0)$.

3.1 A Characterization of Nonzero Elements in $GF(2^n)$

Let a_6 be a fixed nonzero field element in $GF(2^n)$.

Let $x \in GF(2^n)$ with $x \neq 0$, we define the *characterization of x* to be the binary ordered pair $(Tr(x), Tr(\frac{a_6}{x^2}))$. The characterization of x will be helpful to identifying the x-coordinate of points that belong to an elliptic curve or its twist, as well as identifying field elements that are the x-coordinate of points which are doubles. The four possible characterizations are: (1,0), (0,1), (1,1) and (0,0). Those field elements which have characterization of (1,0) and (0,0) represent the field elements which are possible x-coordinates of the double of some elliptic curve point. (Whether a field element is an x-coordinate of a double depends on

the trace of a_2. If $Tr(a_2) = 0$ then it would be those field element with character (0,0), whereas if $Tr(a_2) = 1$ then it would be those field element with character (1,0).)

Now consider the element $\frac{\sqrt{a_6}}{x}$. The characterization of $\frac{\sqrt{a_6}}{x}$ is

$$(Tr(\frac{\sqrt{a_6}}{x}), Tr(\frac{a_6}{\frac{\sqrt{a_6}^2}{x}})) = (Tr(\frac{\sqrt{a_6}}{x}), Tr(x^2)).$$

Since $Tr(x^2) = Tr(x)$ we see that the characterization of $\frac{\sqrt{a_6}}{x}$ is equal to $(Tr(\frac{a_6}{x^2}), Tr(x))$ which is a permutation of the characterization of x. The element $\frac{\sqrt{a_6}}{x}$ is of interest for the following reason: Let $T_2 = (0, \sqrt{a_6})$ then independent of the trace value of a_2 we will always have $T_2 \in E_{a_2,a_6}$. Further $T_2 = -T_2$ If x represents the x-coordinate of some point $P \in E_{a_2,a_6}$ then the x-coordinate of $P + T_2$ is $\frac{\sqrt{a_6}}{x}$.

Observe that if x is an x-coordinate of some point on the elliptic curve E_{a_2,a_6} then the characterization of x satisfies $(Tr(x), Tr(\frac{a_6}{x^2})) = (Tr(x), Tr(x) + Tr(a_2))$. Further the sum of the characterization coordinates of x equals $Tr(a_2)$.

We can define an equivalence relation R on $GF(2^n) \setminus \{0\}$ by: for each $x, y \in GF(2^n) \setminus \{0\}$ we say $x\mathsf{R}y$ provided $y = x$ or $y = \frac{\sqrt{a_6}}{x}$. Each equivalence class contains two elements except for the equivalence class for $\sqrt[4]{a_6}$, which possesses one element. Therefore there are $(2^n - 2)/2 + 1 = 2^{n-1}$ equivalence classes for $GF(2^n) \setminus \{0\}$.

For all $i, j \in \{0, 1\}$ we define

$$\mathcal{A}_{(i,j)} = \{x \in GF(2^n) \setminus \{0, \sqrt[4]{a_6}\} : \ x \text{ has characterization } (i, j)\ \}.$$

For all $x \in \left(\mathcal{A}_{(i,j)} \cup \mathcal{A}_{(1+i,1+j)}\right)$, x will be the x-coordinate of some point on the elliptic curve E_{a_2,a_6} where $Tr(a_2) = i + j$. In fact for all $P \in E_{a_2,a_6} \setminus \mathcal{O}$, if $x_P \notin \{0, \sqrt[4]{a_6}\}$ then $x_P \in \left(\mathcal{A}_{(i,j)} \cup \mathcal{A}_{(1+i,1+j)}\right)$. Of course $\mathcal{A}_{(i+j,0)}$ will contain elements which are the x-coordinate of a double of some point in E_{a_2,a_6} and $\mathcal{A}_{(1+i+j,1)}$ will contain elements which are the x-coordinate of a point in E_{a_2,a_6} which are not doubles.

Let $P_1, P_2 \in E_{a_2,a_6}$. Then the following can be established by utilizing the definition of point addition in E_{a_2,a_6}. If $x_{P_1} \in \mathcal{A}_{(i+j,0)}$ and $x_{P_2} \in \mathcal{A}_{(i+j,0)}$ and $P_1 + P_2 \neq \mathcal{O}$ then $x_{P_1+P_2} \in \mathcal{A}_{(i+j,0)}$. If $x_{P_1} \in \mathcal{A}_{(i+j,0)}$ and $x_{P_2} \in \mathcal{A}_{(1+i+j,1)}$ then $x_{P_1+P_2} \in \mathcal{A}_{(1+i+j,1)}$.

Since we have that for each x, the characterization of $\frac{a_6}{\sqrt{x}}$ is the permutation of the characterization of x, this implies that $|\mathcal{A}_{0,1}| = |\mathcal{A}_{1,0}|$ and that both $|\mathcal{A}_{0,0}|$ and $|\mathcal{A}_{1,1}|$ are even. Also since half of the elements in $GF(2^n)$ have trace 0 and the remaining elements have trace 1, we can infer that if $Tr(a_6) = 1$ then the number of elements of $GF(2^n)$ which have trace 0 is $1 + |\mathcal{A}_{0,1}| + |\mathcal{A}_{0,0}|$, whereas the number of elements which have trace 1 is $1 + |\mathcal{A}_{1,0}| + |\mathcal{A}_{1,1}|$. Thus when $Tr(a_6) = 1$ we have $|\mathcal{A}_{0,0}| = |\mathcal{A}_{1,1}|$. If $Tr(a_6) = 0$ then the number of elements of $GF(2^n)$ which have trace 0 is $1+1+|\mathcal{A}_{0,1}|+|\mathcal{A}_{0,0}|$ and the number of elements which have trace 1 is $|\mathcal{A}_{1,0}| + |\mathcal{A}_{1,1}|$. Therefore when $Tr(a_6) = 0$ we see that $|\mathcal{A}_{1,1}| = |\mathcal{A}_{0,0}| + 2$.

Theorem 2. *The number of points on an elliptic curve E_{a_2,a_6} satisfies:*

(i) $|E_{a_2,a_6}| = 1+1+2\cdot|\mathcal{A}_{0,1}|+2\cdot|\mathcal{A}_{1,0}| = 1+1+2\cdot 2\cdot|\mathcal{A}_{1,0}| = 2+4\cdot|\mathcal{A}_{1,0}|$ *provided that* $Tr(a_2)=1$

(ii) $|E_{a_2,a_6}| = 4+4\cdot|\mathcal{A}_{0,0}|$ *provided that* $Tr(a_2)=0$ *and* $Tr(a_6)=1$

(iii) $|E_{a_2,a_6}| = 8+4\cdot|\mathcal{A}_{0,0}|$ *provided that* $Tr(a_2)=0$ *and* $Tr(a_6)=0$

Proof. The proof of (i): Suppose $Tr(a_2)=1$. The elliptic curve E_{a_2,a_6} will include the point of infinity, and the point $(0,\sqrt{a_6})$. In addition, for each $x \in (\mathcal{A}_{(1,0)} \cup \mathcal{A}_{(0,1)})$ there will exist two values of y such that $(x,y) \in E_{a_2,a_6}$. Lastly recall that $|\mathcal{A}_{(1,0)}| = |\mathcal{A}_{(0,1)}|$. Therefore $|E_{a_2,a_6}| = 1+1+2\cdot|\mathcal{A}_{0,1}|+2\cdot|\mathcal{A}_{1,0}| = 1+1+2\cdot 2\cdot|\mathcal{A}_{1,0}| = 2+4\cdot|\mathcal{A}_{1,0}|$.

The proofs of (ii) and (iii) follow from a similar counting argument.

Recall that $|\mathcal{A}_{(i,i)}|$ is even for $i=0,1$. Therefore an elliptic curve will have a cofactor of 2 iff $Tr(a_2)=1$ and $1+2\cdot|\mathcal{A}_{(0,1)}|$ is prime. An elliptic curve will have a cofactor of 4 iff $Tr(a_2)=0$, $Tr(a_6)=1$ and $1+|\mathcal{A}_{(0,1)}|$ is prime. For $L>2$, an elliptic curve will have cofactor of 2^L iff $Tr(a_2)=0$, $Tr(a_6)=0$ and $1+|\mathcal{A}_{(0,0)}|/2^{L-2}$ is prime.

As described by the above theorem the number of points on an elliptic curve, depends on the characterization of elements in $GF(2^n)$ and the trace of the elliptic curve parameters a_2 and a_6. If we fix the parameter a_6 and vary the parameter a_2 then the characterization for each x in $GF(2^n)$ will be fixed. Therefore we have the following (this same result is provided in [2]).

Theorem 3. *Let $\gamma \in GF(2^n)$ such that $Tr(\gamma)=0$ then for all a_2, a_6 we have*

$$|E_{a_2+\gamma,a_6}| = |E_{a_2,a_6}|$$

Proof. For a fixed a_2 and a γ with $Tr(\gamma)$=0, we have $Tr(a_2+\gamma) = Tr(a_2)$

A consequence of this theorem is that if E_{a_2,a_6} represents a cryptographically relevant elliptic curve defined over $GF(2^n)$. Then there exists 2^{n-1} many cryptographically relevant curves defined over the same field. In [13], it was shown that these curves are isomorphic to each other.

Let $a_2, a_6 \in GF(2^n)$. Then this fixes some elliptic curve E_{a_2,a_6}. Let $\gamma \in GF(2^n)$ where $Tr(\gamma)=0$. Then [13] has established that both E_{a_2,a_6} and $E_{a_2+\gamma,a_6}$ are isomorphic. But we will see that we can make even more inferences concerning the isomorphism. Suppose that E_{a_2,a_6} has a cofactor of 2^L. Then for all $P=(x,y) \in E_{a_2,a_6}$, there exists a $\zeta \in GF(2^n)$ such that $(x,\zeta) \in E_{a_2+\gamma,a_6}$. It can be shown that $\zeta = y + x\cdot\mathbf{Solve}(\gamma)$. That is, $(x, y+x\cdot\mathbf{Solve}(\gamma)) \in E_{a_2+\gamma,a_6}$. Let $\lambda = \frac{y+x\cdot\mathbf{Solve}(\gamma)}{x}$, then $\lambda^2+\lambda = \frac{y^2}{x^2}+\frac{y}{x}+\mathbf{Solve}^2(\gamma)+\mathbf{Solve}(\gamma) = x + a_2 + \frac{a_6}{x^2}+\gamma = x+(a_2+\gamma)+\frac{a_6}{x^2}$. It is obvious by the tools that we have developed, that the point $P=(x,y) \in E_{a_2,a_6}$ is a double of some point iff the point $(x, y+x\cdot\mathbf{Solve}(\gamma)) \in E_{a_2+\gamma,a_6}$ is a double of some point in $E_{a_2+\gamma,a_6}$. Further whenever $P=(x,y) \in G \subset E_{a_2,a_6}$ (where G is the subgroup of large prime order), then $(x, y+x\cdot\mathbf{Solve}(\gamma))$ belongs to a subgroup of $E_{a_2+\gamma,a_6}$ of the same prime order as G. Thus we see that not only are E_{a_2,a_6} and $E_{a_2+\gamma,a_6}$ isomorphic, when $Tr(\gamma)=0$, but that this isomorphism is trivial to compute.

Consequently the only relevant parameters to consider for a_2 are 0 and 1 (as long as n is odd). In the WTLS specification of WAP [16], an elliptic curve identified as *curve 4* in the specification, is defined where the a_2 parameter is described in *Table 1* (see below). Since the $Tr(a_2) = 1$, this curve is isomorphic to E_{1,a_6} where the parameter a_6 is given in Table 1. The elliptic curve E_{1,a_6} has a subgroup of large prime order, the same as the order given in Table 1. This subgroup of E_{1,a_6} has a generator $G' = (g'_x, g'_y)$ where $g'_x = G_x$ and $g'_y = G_y + G_x \cdot \mathbf{SOLVE}(072546B5435234A422E0789675F432C89435DE5243)$. From an implementation point of view it is much more efficient to use the elliptic curve E_{1,a_6} then the curve described in Table 1, for whenever one has to perform a field multiplication with a_2, if $a_2 = 1$ then it is free. This type of field multiplication would always be needed when one implements the elliptic curve using a projective point representation. Thus the parameters of curve 4 in WTLS specification should be changed to reflect this.

generating polynomial	$t^{163} + t^8 + t^2 + t + 1$
a_2	$072546B5435234A422E0789675F432C89435DE5242$
a_6	$00C9517D06D5240D3CFF38C74B20B6CD4D6F9DD4D9$
order of the generator $G = (G_x, G_y)$	$040000000000000000001E60FC8821CC74DAEAFC1$
G_x	$07AF69989546103D79329FCC3D74880F33BBE803CB$
G_y	$01EC23211B5966ADEA1D3F87F7EA5848AEF0B7CA9F$
cofactor	2

Table 1

4 An Improved Point Compression Method

Let G denote the set of points of prime order and let $T_2 = (0, \sqrt{a_6})$.

If $Tr(a_2) = 0$ then $x^2 + \frac{a_6}{x^2} = 0$ is solvable, with solution $x = \sqrt[4]{a_6}$. Now characterization of $\sqrt[4]{a_6}$ is $(Tr(\sqrt[4]{a_6}), Tr(\frac{a_6}{(\sqrt[4]{a_6})^2}) = (Tr(a_6), Tr(a_6))$. Thus T_2 is the double of some point with an x-coordinate of $\sqrt[4]{a_6}$. Let Q_1 and Q_3 denote the two points of E which are $\frac{1}{2}T_2$.

Suppose $Tr(a_6) = 1$ and $Tr(a_2) = 0$. Then the x-coordinates of Q_1 and Q_3 have characterization $(Tr(a_6), Tr(a_6)) = (1, 1)$. Therefore both Q_1 and Q_3 are not doubles of any points. Thus we see that there exists a subgroup of order 4 which contains $\mathcal{O}, Q_1, T_2$, and Q_3. Let $P \in G \setminus \{\mathcal{O}\}$, then the characterization of x_P is (0,0) and the characterization of x_{P+T_2} is (0,0). The characterizations of x_{P+Q_1} and x_{P+Q_3} are (1,1), this follows from that fact that both Q_1 and Q_3 are NOT DOUBLEs of any points. Observe that given an point $P = (x, y)$ in G, the field element $\frac{\sqrt{a_6}}{x}$ is the x-coordinate of an EC point which is in the coset $G + T_2$. Now all points $R \in G + T_2$ do have a half but all of its halves do not have a half. Therefore if we found a y such that $R = (x_R, y) \in E$, and then set $\lambda = x_R + \frac{y}{x_R}$ (so that $R = (x_R, \lambda)$ using Knudsen's definition [6]) and compute $(u, \lambda_U) = \mathbf{HALF}(x_R, \lambda)$ then $\mathbf{HALF}(u, \lambda_U) = \textit{No half point.}$

If $Tr(a_6) = 0$ and $Tr(a_2) = 0$, then the half of T_2 is Q_1 and Q_3, and both Q_1 and Q_3 are doubles. So there exists a subgroup of order 2^{m+1} which contains Q_1, T_2, Q_3. Thus $\frac{1}{2^m}T_2 \in E$, but $\frac{1}{2^m}T_2$ does not have a half. Again if $P = (x, y) \in G$ then $\frac{\sqrt{a_6}}{x}$ is the x-coordinate of $P+T_2$. If we compute y such that $(\frac{\sqrt{a_6}}{x}, y) \in E$, set $\lambda = \frac{\sqrt{a_6}}{x} + \frac{y}{\frac{\sqrt{a_6}}{x}}$ then repeatedly call the **HALF** function eventually we will arrive at *No half point,* i.e. $\mathbf{HALF}^{m+1}(\frac{\sqrt{a_6}}{x}, \lambda) = \textit{No half point.}$

4.1 A Point Compression for E_{a_2,a_6} when $Tr(a_2) = 0$

Let $\alpha \in GF(2^n)$ and represent $\alpha = (\rho_{n-1}, \ldots, \rho_1, \rho_0)$. Let i denote the smallest subscript such that ρ_i is used to compute trace of r (for most fields i will be 0). Let $\zeta = (\xi_{n-1}, ..., \xi_0) \in GF(2^n)$ such that $Tr(\zeta) = 0$ (which equals $Tr(a_2)$=0). If a sender Alice wishes to transmit ζ to the receiver Bob they should send $compress(\zeta) = (\xi_{n-1}, ..\xi_{i+1}, \xi_{i-1}, ..., \xi_0)$ which is merely ζ where we have removed the i^{th} term. If a receiver Bob receives $compress(\zeta)$ then Bob will be able to reconstruct ζ. Since Bob knows all parameters of the elliptic curve he knows both $Tr(a_2) = 0$ and the smallest subscript i which is used to compute the trace. Thus Bob knows which bit ξ_i was omitted, by guessing $\xi_i = 0$ and computing the trace of the corresponding field element, Bob can verify whether his guess was correct. His guess was correct if the trace value equals $Tr(a_2)$. Otherwise, if the trace value doesn't equal $Tr(a_2)$, then Bob knows the correct ζ satisfied $\xi_i = 1$. Thus $n-1$ bits are required to communicate an element $\zeta \in GF(2^n)$ where $Tr(\zeta) = 0$ and where $Tr(a_2) = 0$.

If a receiver is able to compute the x-coordinate of point P then the receiver will compute y as follows: first compute $z = \mathbf{SOLVE}(x + a_2 + \frac{a_6}{x^2})$ then compute $y = x \cdot z$. The problem is that there are two solutions to $\mathbf{SOLVE}(x + a_2 + \frac{a_6}{x^2})$, one with trace 0 and the other with trace 1. So the sender must communicate the trace of $\frac{y}{x}$ which we will denote as ϵ. If $z = \mathbf{SOLVE}(x + a_2 + \frac{a_6}{x^2})$ and if $Tr(z) = \beta$ then $y = x \cdot z$, else if $Tr(z) \neq \epsilon$ then $y = x \cdot (z + 1)$.

We now describe how to accomplish a point compression of $n-1$ bits. Let T_2 denote the point $(0, \sqrt{a_6}) \in E$, then T_2 has a half since $Tr(a_2) = 0$. Let $P = (x, y)$ be a cryptographically relevant point on E. Then P belongs to G a subgroup of prime order, thus the trace of x is 0. The goal is that the sender will submit to the receiver $n-1$ bits such that the receiver will be able to expand these bits to compute P. The sender and the receiver share the elliptic curve parameters, and both know the underlying field. Now for the sender to send $P = (x, y)$, they do the following: If $\frac{y}{x}$ has trace 0 the sender sets $\zeta = x$, else if $Tr(\frac{y}{x}) = 1$ the sender sets $\zeta = \frac{\sqrt{a_6}}{x}$.[4] Then since $Tr(a_2) = 0$ we have $Tr(\zeta) = 0$. Thus to transmit ζ the sender sends $compress(\zeta)$ which is $n-1$ bits. When the receiver receives $compress(\zeta)$ they will be able to reconstruct ζ as described above, since $Tr(\zeta) = 0$. At this time they compute y by first solving $z = \mathbf{SOLVE}(\zeta + a_2 + \frac{a_6}{\zeta^2})$ where z satisfies $Tr(z) = 0$. They then set $y = \zeta \cdot z$. Since

[4] $\frac{x}{\sqrt{a_6}}$ is the x coordinate of the point $P + T_2$, when $Tr(x) = 0$ we have $Tr(\frac{\sqrt{a_6}}{x}) = 0$.

$Tr(a_2) = 0$ there exists an m such that $\frac{1}{2^m}T_2 \in E$ (here $T_2 = (0, \sqrt{a_6})$) but $\frac{1}{2^m}T_2$ does not have a half. Since the receiver knows all elliptic curve parameters they know m. The receiver computes $\mathbf{HALF}^{m+1}(\zeta, \zeta + \frac{y}{\zeta})$, if a point is returned, then the receiver knows $P = (x, y) = (\zeta, y)$. However if $\mathbf{HALF}^{m+1}(\zeta, \zeta + \frac{y}{\zeta})$ returns *No half point* then $\zeta = \frac{\sqrt{a_6}}{x}$. So they compute x by $x = \frac{\sqrt{a_6}}{\zeta}$. Then they compute $z = \mathbf{SOLVE}(x + a_2 + \frac{a_6}{x^2}) = \mathbf{SOLVE}(\frac{\sqrt{a_6}}{\zeta} + a_2 + \zeta^2)$ but this time they select z so that z has trace 1. Finally they compute y by $y = x \cdot z$. Many of the elliptic curves for which $Tr(a_2) = 0$, will have a cofactor of 4 which implies that m will be 1. That is, if $Tr(a_2) = 0$ and the cofactor of the elliptic curve is 4, then T_2 belongs to a subgroup of order 4, thus $\frac{1}{2}T_2$ exists but $\frac{1}{2^2}T_2$ does not exist. All binary elliptic curves in the NIST recommended list of curves [10] for which $Tr(a_2) = 0$ have cofactors of 4.

Theorem 4. *Let E_{a_2,a_6} be an elliptic curve defined over $GF(2^n)$ where $Tr(a_2) = 0$ then there exists an efficient point compression algorithm that will allow a sender to transmit $n - 1$ bits to send a point on the curve of prime order.*

Consequently, we see that this point compression method requires less bandwidth than the patented compression methods described in [2,15] whenever $Tr(a_2) = 0$.

4.2 Point Compression Algorithm for E_{a_2,a_6} where $Tr(a_2) = 1$

Thus we see that if $Tr(a_2) = 0$ there exists a point compression method that is superior to the previous point compression methods. It would be preferred to provide a point compression method which is the most efficient, and which utilizes comparable techniques for all cases. And so we now describe a point compression method for the case $Tr(a_2) = 1$. For the case $Tr(a_2) = 1$ we will demonstrate a method which is as efficient as the method by Seroussi, the benefit is that the form is comparable to the method that we described above.

Let $P = (x, y)$ be a cryptographically relevant point on E. Then P belongs to G a subgroup of prime order. Thus the characterization of x is (0,1). The method is such that the sender will submit to the receiver n bits such that the receiver will be able to expand these bits to compute P. Given x, one computes $z = \mathbf{SOLVE}(x + a_2 + \frac{a_6}{x^2})$ since there are two solutions one needs to know the correct trace value of the z-solution. y then satisfies $y = zx$. To provide a unified approach to point compression we suggest that if $Tr(\frac{y}{x}) = 0$ the sender sets $\zeta = x$, otherwise if $Tr(\frac{y}{x}) = 1$ the sender sets $\zeta = \frac{\sqrt{a_6}}{x}$.

Suppose a sender and a receiver exchange an elliptic curve point. If the receiver receives ζ where $Tr(\zeta) = 0$ then the exchanged point $P = (x, y)$ is such that $x = \zeta$ and y satisfies $Tr(\frac{y}{x}) = 0$. First the receiver computes $\lambda = \mathbf{SOLVE}(\zeta + a_2 + \frac{a_6}{\zeta^2})$ where $Tr(\lambda) = 0$. Then the receiver sets $y = x \cdot \lambda$. If the receiver receives ζ where $Tr(\zeta) = 1$ then the exchanged point $P = (x, y)$ is such that $x = \frac{\sqrt{a_6}}{\zeta}$ and y satisfies $Tr(\frac{y}{x}) = 1$. First the receiver computes $\lambda = \mathbf{SOLVE}(\frac{\sqrt{a_6}}{\zeta} + a_2 + \zeta^2)$ where $Tr(\lambda) = 1$. Then the receiver sets $y = x \cdot \lambda$.

The efficiency (here we measure it in terms of the number of field operations that need to be computed) is as efficient (perhaps slightly more efficient) than Seroussi's method [12]. In our method the receiver will perform (in the worst case) two trace checks, an inversion, a square, a multiply and a **SOLVE.** The receiver may have precomputed and stored the $\sqrt{a_6}$. Although in [4], it was demonstrated that a square root can be computed as nearly as efficient as a square (even when using a polynomial basis to represent a field element) for many fields $GF(2^n)$. In Seroussi's method a bit needs to be guessed, inserted into the stream, a trace check, a bit may need to be changed, a square, a multiply, an inversion, a **SOLVE**, and one more trace check.

5 Attacking a Users Key Using Invalid ECC Parameters

Our last observation concerning utilizing the tools that we have developed in this paper, is its use to efficiently check an elliptic curve parameter. It is important that during a key exchange a receiver checks elliptic curve parameters before utilizing these parameters with their private key [1]. One important parameter check is to verify that a received point is a point of prime order. Here we will assume that the sender and receiver are performing some type of elliptic curve key exchange and that the receiver receives a point $J_{received} = (x, y)$. The receiver has private key k and will compute $kJ_{received} = (a, b)$. In the end both receiver and sender will have derived (a, b). Of course they will *hash* a. If the receiver does not check that $J_{received}$ is of prime order then the sender may be able to detect a bit of the receivers key k.

We will describe the attack and the remedy for the case when the elliptic curve parameter a_2 satisfies $Tr(a_2) = 0$. Let G represent the subgroup of E of prime order. The attack made by the sender is as follows. The sender sends a point $J_{received} \in G + T_2$, of course the x-coordinate of $J_{received}$ has trace 0. The only way the receiver can determine that J belongs to the coset $G + T_2$, is to compute $pJ_{received}$ where p is the prime order of G. If the receiver does not check the order of $J_{received}$ then when the receiver computes $kJ_{received}$, if $k_0 = 0$ then $kJ_{received} \in G$, if $k_0 = 1$ then $kJ_{received}$ belongs to the coset $G+T_2$. Thus the low bit of the key is vulnerable to this attack. A solution is that if G is a subgroup of order p then the receiver should compute $pJ_{received}$ to verify that it is the identity $\mathcal{O}$, but this will be at a cost of performance. If an elliptic curve has a cofactor of 2^m (which is true for all curves in [10,16]), then there is an efficient method which will allow us to distinguish between a point in G and a point in the coset $G + T_2$. The alternative (the efficient check) is to first determine m such that $\frac{1}{2^m}T_2 \in E$ but where $\frac{1}{2^m}T_2$ does not have a half. Then the receiver computes $\mathbf{Half}^{m+1}(x, x + \frac{y}{x})$. If the result is a point then element was of prime order, otherwise it belonged to the coset.

In some cases this parameter check will be trivial. For example suppose that the elliptic curve has a cofactor of 2. Then a parameter check is trivial, simply determine if $(x, y) \in E$ and $Tr(x) = 1$.

6 Conclusion

Our work has provided several new tools in $GF(2^n)$ that provided great insight into elliptic curve defined over $GF(2^n)$. It has provided a new way to view the number of points on an elliptic curve. As well as provide us a mean to choose more efficient elliptic curve parameters (for example curve 4 in the WTLS list). Our main result is new point compression method which is superior to prior methods whenever $Tr(a_2) = 0$. Lastly we have demonstrated how the halving algorithm can be utilized to check elliptic curve parameters.

References

1. Adrian Antipa, Daniel R. L. Brown, Alfred Menezes, Ren Struik, Scott A. Vanstone: "Validation of Elliptic Curve Public Keys". *Public Key Cryptography - PKC 2003* 211-223
2. I.F. Blake, Nigel Smart, and G. Seroussi, *Elliptic Curves in Cryptography.* London Mathematical Society Lecture Note Series. Cambridge University Press, 1999.
3. Darrel Hankerson, Julio Lopez Hernandez and Alfred Menezes. "Software Implementation of Elliptic Curve Cryptography over Binary Fields". In *CHES 2000.* p. 1-24.
4. B. King and B. Rubin. "Revisiting the point halving algorithm". *Technical Report.*
5. Neal Koblitz, *Elliptic curve cryptosystems,* Mathematics of Computation, Vol. 48, No. 177, 1987, 203-209.
6. Erik Woodward Knudsen. "Elliptic Scalar Multiplication Using Point Halving". In *Advances in Cryptology - ASIACRYPT '99.* LNCS Vol. 1716, Springer, 1999, p. 135-149
7. R. Lidl and H. Niederreiter. *Finite Fields,* Second edition, Cambridge University Press, 1997.
8. Alfred Menezes, *Elliptic Curve Public Key Cryptosystems,* Kluwer Academic Publishers, 1993.
9. Victor S. Miller, "Use of Elliptic Curves in Cryptography", In *Advances in Cryptology CRYPTO 1985*,Springer-Verlag, New York, 1985, pp 417-42
10. NIST, *Recommended elliptic curves for federal use,* http://www.nist.gov
11. Rich Schroeppel. "Elliptic Curves: Twice as Fast!". In *Rump session of CRYPTO 2000.*
12. G. Seroussi. "Compact Representation of Elliptic Curve Points over F_{2^n}", *HP Labs Technical Reports* , `http://www.hpl.hp.com/techreports/98/HPL-98-94R1.html` , pg. 1-6.
13. J. Silverman. *The Arithmetic of Elliptic Curves.* Springer-Verlag.New York. 1986.
14. N. P. Smart. A note on the x-coordinate of points on an elliptic curve in characteristic two. Information Processing Letters, 80(?):261–263, October 2001
15. *IEEE P1363 Appendix A.* http://www.grouper.org/groups/1363
16. *WTLS Specification,* http://www.wapforum.org

7 Appendix

7.1 NIST Recommended Curves in $GF(2^n)$

In July 1999 NIST releases a list of recommended but not required curves to use for Elliptic curve cryptography when dealing with federal agencies. Today several of these curve have been adopted by many standards. Our interest is in those curves over the binaryfield $GF(2^n)$. The curves listed are: K-163, B-163, K-233, B-233, K-283, B-283, K-409, B-409,K-571, and B-571 where the K-*** refers to a Koblitz curve whose Weierstrass equation if of the form

$$y^2 + xy = x^3 + a_2x^2 + 1$$

and B-*** refer to a "random curve" whose Weierstrass equation is of the form

$$y^2 + xy = x^3 + x^2 + b$$

For Koblitz curve K-163 the coefficient $a = 1$, for the remaining Koblitz curves K-233, K-283, K-409, and K-571 the coefficient $a = 0$. Thus K-163 the $Tr(a_2) = 1$ and for the other four Koblitz curves K-233, K-283, K-409, and K-571 the $Tr(a_2) = 0$. The table provided below demonstrate a very efficient way to perform a trace check when utilizing a NIST curve. We have reproduced this table, which was originally given in [4].

Curve types	Generating polynomial	condition for $\mu \in GF(2^n)$ to satisfy $Tr(\mu) = 0$
K-163, B-163	$p(t) = t^{163} + t^7 + t^6 + t^3 + 1$	$\mu_0 = \mu_{157}$
K-233, B-233	$p(t) = t^{233} + t^{74} + 1$	$\mu_0 = \mu_{159}$
K-283, B-283	$p(t) = t^{283} + t^{12} + t^7 + t^5 + 1$	$\mu_0 = \mu_{277}$
K-409, B-409	$p(t) = t^{409} + t^{87} + 1$	$\mu_0 = 0$
K-571, B-571	$p(t) = t^{571} + t^{10} + t^5 + t^2 + 1$	$\mu_0 + \mu_{561} + \mu_{569} = 0$

Table 2

On the Optimal Parameter Choice for Elliptic Curve Cryptosystems Using Isogeny

Toru Akishita[1]* and Tsuyoshi Takagi[2]

[1] Sony Corporation, Ubiquitous Technology Laboratories,
6-7-35 Kitashinagawa Shinagawa-ku, Tokyo, 141-0001 Japan
akishita@pal.arch.sony.co.jp
[2] Technische Universität Darmstadt, Fachbereich Informatik,
Alexanderstr.10, D-64283 Darmstadt, Germany
ttakagi@cdc.informatik.tu-darmstadt.de

Abstract. The isogeny for elliptic curve cryptosystems was initially used for the efficient improvement of order counting methods. Recently, Smart proposed the countermeasure using isogeny for resisting the refined differential power analysis by Goubin (Goubin's attack). In this paper, we examine the countermeasure using isogeny against zero-value point (ZVP) attack that is generalization of Goubin's attack. We show that some curves require higher order of isogeny to prevent ZVP attack. Moreover, we prove that this countermeasure cannot transfer a class of curve to the efficient curve that is secure against ZVP attack. This class satisfies that the curve order is odd and $(-3/p) = -1$ for the base field p, and includes three SECG curves. In the addition, we compare some efficient algorithms that are secure against both Goubin's attack and ZVP attack, and present the most efficient method of computing the scalar multiplication for each curve from SECG. Finally, we discuss another improvement for the efficient scalar multiplication, namely the usage of the point $(0, y)$ for the base point of curve parameters. We are able to improve about 11% for double-and-add-always method, when the point $(0, y)$ exists in the underlying curve or its isogeny.

Keywords: elliptic curve cryptosystems, isomorphism, isogeny, side channel attack, zero-value point attack.

1 Introduction

Elliptic curve cryptosystem (ECC) is an efficient public-key cryptosystem with a short key size. ECC is suitable for implementing on memory-constraint devices such as mobile devices. However, if the implementation is careless, side channel attack (SCA) might reveal the secret key of ECC. We have to carefully investigate the implementation of ECC in order to achieve the high security.

The standard method of defending SCA on ECC is randomizing the curves parameters, for instance, randomizing a base point in projective coordinates [5]

* This work was done while the first author stayed at Technische Universität Darmstadt, Germany.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 346–359, 2004.

and randomizing curve parameters in the isomorphic class [11]. However, Goubin pointed out that the point $(0, y)$ cannot be randomized by these methods [7]. He proposed a refined differential power analysis using the point $(0, y)$. This attack has been extended to the zero value of the auxiliary registers, called the zero-value point (ZVP) attack [1]. Both Goubin's attack and the ZVP attack assume that the base point P can be chosen by the attacker and the secret scalar d is fixed, so that we need to care these attacks in ECIES and single-pass ECDH, but not in ECDSA and two-pass ECDH.

In order to resist Goubin's attack, Smart proposed to map the underlying curve to the isogenous curve that does not have the point $(0, y)$ [17]. This countermeasure with a small isogeny degree is faster than randomizing the secret scalar d with the order of the curve. However, the security of this countermeasure against the ZVP attack has not been discussed yet — it could be vulnerable to the ZVP attack.

1.1 Contribution of This Paper

In this paper, we examine the countermeasure using isogeny against the ZVP attack. The zero-value points *(ED1)* $3x^2 + a = 0$, *(MD1)* $x^2 - a = 0$, and *(MD2)* $x^2 + a = 0$ were examined. We show that some curves require higher order of isogeny to prevent the ZVP attack. For example, SECG secp112r1 [18] is secure against Goubin's attack, but insecure against the ZVP attack. Then, the 7-isogenous curve to secp112r1 is secure against both attacks. We require isogeny of degree 7 to prevent the ZVP attack. For each SECG curve we search the minimal degree of isogeny to the curve that is secure against both Goubin's attack and the ZVP attack. Since the ZVP attack strongly depends on the structure of addition formula, the minimal degree of isogeny depends on not only the curve itself but also addition formula. Interestingly, three SECG curves cannot be mapped to the curve with $a = -3$ that is secure against the ZVP attack. The curve with $a = -3$ is important for efficiency. We prove that this countermeasure cannot map a class of curve to the curve with $a = -3$ that is secure against the ZVP attack. This class satisfies that the curve order is odd and $(-3/p) = -1$ for the base field p, and these three curves belong to this class.

Moreover, we estimate the total cost of the scalar multiplication in the necessity of resistance against both Goubin's attack and the ZVP attack. We compare two efficient DPA-resistant methods, namely the window-based method and Montgomery-type method, with the countermeasure using isogeny, and present the most efficient method to compute the scalar multiplication for each SECG curve.

Finally we show another efficient method for computing the scalar multiplication, namely using the point $(0, y)$ for the base point. We can prove the discrete logarithm problem with the base point $(0, y)$ is as intractable as using a random one thanks to the random self reducibility. Comparing with the previous method we are able to achieve about 11% faster scalar multiplication using the double-and-add-always method. This base point can also save 50% memory space without any compression trick. We propose the scenario to utilize

the proposed method efficiently and show the example of a curve to achieve this scenario.

This paper is organized as follows: Section 2 briefly reviews known results about elliptic curve cryptosystems. Section 3 describes the choices of secure curve against the ZVP attack using isogeny. In Section 4 we show the efficient implementations using isogeny. In Section 5 we state concluding remarks.

2 Elliptic Curve Cryptosystems

In this section we review some results on elliptic curve cryptosystems related to isogeny. Let $K = \mathbb{F}_p$ be a finite field, where $p > 3$. The Weierstrass form of an elliptic curve over K is described as

$$E : y^2 = x^3 + ax + b \ \ (a, b \in K,\ \Delta = -16(4a^3 + 27b^2) \neq 0).$$

The set of all points $P = (x, y)$ satisfying E, together with the point of infinity $\mathcal{O}$, is denoted by $E(K)$, which forms an Abelian group. Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be two points on $E(K)$ that don't equal to $\mathcal{O}$. The sum $P_3 = P_1 + P_2 = (x_3, y_3)$ can be computed as $x_3 = \lambda(P_1, P_2)^2 - x_1 - x_2$, $y_3 = \lambda(P_1, P_2)(x_1 - x_3) - y_1$, where $\lambda(P_1, P_2) = (3{x_1}^2 + a)/(2y_1)$ for $P_1 = P_2$, and $\lambda(P_1, P_2) = (y_2 - y_1)/(x_2 - x_1)$ for $P_1 \neq \pm P_2$. We call the former, $P_1 + P_2$ ($P_1 = P_2$), the elliptic curve doubling (ECDBL) and the latter, $P_1 + P_2$ ($P_1 \neq \pm P_2$), the elliptic curve addition (ECADD) in affine coordinate (x, y). These two addition formulae respectively need one inversion over K, which is much more expensive than multiplication over K. Therefore, we transform affine coordinate (x, y) into other coordinates where inversion is not required. In this paper we deal with Jacobian coordinates $(X : Y : Z)$ setting $x = X/Z^2$ and $y = Y/Z^3$. The doubling and addition formulae can be represented as follows.

ECDBL in Jacobian Coordinates ($\mathrm{ECDBL}^{\mathcal{J}}$) :
$X_3 = T$, $Y_3 = -8{Y_1}^4 + M(S - T)$, $Z_3 = 2Y_1Z_1$,
$S = 4X_1{Y_1}^2$, $M = 3{X_1}^2 + a{Z_1}^4$, $T = -2S + M^2$.

ECADD in Jacobian Coordinates ($\mathrm{ECADD}^{\mathcal{J}}$) :
$X_3 = -H^3 - 2U_1H^2 + R^2$, $Y_3 = -S_1H^3 + R(U_1H^2 - X_3)$, $Z_3 = Z_1Z_2H$,
$U_1 = X_1{Z_2}^2$, $U_2 = X_2{Z_1}^2$, $S_1 = Y_1{Z_2}^3$, $S_2 = Y_2{Z_1}^3$, $H = U_2 - U_1$, $R = S_2 - S_1$.

We call these formulae as the standard addition formulae. For $\mathrm{ECADD}^{\mathcal{J}}$ we require 16 multiplications when $Z_1 \neq 1$ and 11 ones when $Z_1 = 1$. For $\mathrm{ECDBL}^{\mathcal{J}}$ we require 10 multiplications in general, 9 ones when a is small, and only 8 ones when $a = -3$ by $M = 3(X_1 + {Z_1}^2)(X_1 - {Z_1}^2)$. Thus all SECG random curves over $\mathbb{F}_p$ with prime order satisfy $a = -3$. In this paper, we are interested in the curves with prime order such as these curves.

2.1 Scalar Multiplication and Side Channel Attack

The scalar multiplication evaluates dP for a given integer d and a base point P of ECC. A standard algorithm of computing dP is a binary method, which is implemented by repeatedly calling ECDBL and ECADD. Let $d = (d_{n-1} \cdots d_1 d_0)_2$ be the binary representation of d where $d_{n-1} = 1$ The binary method is as follows:

Binary method	Double-and-add-always method
Input: an n-bit d, a base point P	**Input:** an n-bit d, a base point P
Output: scalar multiplication dP	**Output:** scalar multiplication dP
1. $Q \leftarrow P$	1. $Q[0] \leftarrow P$
2. For $i = n-2$ to 0	2. For $i = n-2$ to 0
2.1. $Q \leftarrow \mathrm{ECDBL}(Q)$	2.1. $Q[0] \leftarrow \mathrm{ECDBL}(Q[0])$
2.2. if $d_i = 1$ then	2.2. $Q[1] \leftarrow \mathrm{ECADD}(Q[0], P)$
$Q \leftarrow \mathrm{ECADD}(Q, P)$	2.3. $Q[0] \leftarrow Q[d_i]$
3. Return Q	3. Return $Q[0]$

The SPA uses a single observation of the power consumption to obtain the information of secret key. The binary method is vulnerable to SPA. Since ECADD is computed only if the underlying bit is 1 and a SPA attacker can distinguish ECDBL and ECADD, he can detect the secret bit. Coron proposed a simple countermeasure called as the double-and-add-always method [5]. The attacker cannot guess the bit information because this method always computes ECADD whether $d_i = 0$ or 1. Two more efficient methods have been proposed. The first is window-based method [13,14,16] and the second is Montgomery-type method [3,6,8,9,10].

The DPA uses many observations of the power consumption together with statistical tools. To enhance SPA security to DPA security, we must insert random numbers during computation of dP. The standard randomization methods for the base point P are Coron's 3rd countermeasure [5] and Joye-Tymen countermeasure [11]. In order to randomize the representation of the processing point, Coron's 3rd countermeasure uses randomized representation of Jacobian (projective) coordinates and Joye-Tymen countermeasure uses randomized isomorphism of an elliptic curve.

2.2 Efficient Method Secure against DPA

Window-Based Method The window-based method secure against SPA was first proposed by Möller [13,14], and optimized by Okeya and Takagi [16]. This method uses the standard addition formulae the same as the double-and-add-always method. It makes the fixed pattern $|0 \cdots 0x|0 \cdots 0x| \cdots |0 \cdots 0x|$ for some x. Though the SPA attacker distinguishes ECDBL and ECADD in the scalar multiplication by measuring the power consumption, he obtains only the identical sequence $|D \cdots DA|D \cdots DA| \cdots |D \cdots DA|$, where D and A denote ECDBL and ECADD, respectively. Therefore, he cannot guess the bit information. This method reduces ECADD as compared with the double-and-add-always method

and thus enables efficiency. In order to enhance this method to be DPA-resistant, we have to insert a random value using Coron's 3rd countermeasure or Joye-Tymen countermeasure. Moreover, we have to randomize the value of table to protect 2nd order DPA. We estimate the computational cost of the scalar multiplication dP according to [16]. Denote the computational cost of multiplication and inversion in the definition field by M and I, respectively. The total cost is estimated as $(16 \cdot 2^w + (9w + 21)k - 18))M + I$ when a is small and $(16 \cdot 2^w + (8w + 21)k - 18))M + I$ when $a = -3$, where n is the bit length of d, w is the window size, and $k = \lceil n/w \rceil$.

Montgomery-Type Method Montgomery-type method was originally proposed by Montgomery [15] and enhanced to the Weierstrass form of elliptic curves over K [3,6,8,9,10]. This method always computes ECADD and ECDBL whether $d_i = 0$ or 1 as the double-and-add-always method, and thus satisfies SPA-resistance. In this method, we don't need to use y-coordinate (Y-coordinate in projective coordinates) to compute the scalar multiplication dP. This leads the efficiency of Montgomery-type method. In the original method ECADD and ECDBL are computed separately. However, Izu and Takagi encapsulated these formulae into one formula mECADDDBL to share intermediate variables and cut two multiplications [10]. Let $P_1 = (X_1 : Z_1)$ and $P_2 = (X_2 : Z_2)$ in projective coordinates, which don't equal to $\mathcal{O}$, by setting $x = X/Z$. In the following we describe the encapsulated formula $\mathrm{mECADDDBL}^{\mathcal{P}}$, which compute $P_3 = (X_3 : Z_3) = P_1 + P_2$ and $P_4 = (X_4 : Z_4) = 2P_1$, where $P_1 \neq \pm P_2$, $P_3' = (X_3' : Z_3') = P_1 - P_2$ and $(X_3', Z_3' \neq 0)$.

ECADDDBL in Montgomery-Type Method ($\mathrm{mECADDDBL}^{\mathcal{P}}$) :
$X_3 = Z_3'(2(X_1Z_2 + X_2Z_1)(X_1X_2 + aZ_1Z_2) + 4bZ_1{}^2Z_2{}^2)) - X_3'(X_1Z_2 - X_2Z_1)^2$,
$Z_3 = Z_3'(X_1Z_2 - X_2Z_1)^2$,
$X_4 = (X_1{}^2Z_2{}^2 - aZ_1{}^2Z_2{}^2)^2 - 8bX_1Z_1{}^3Z_2{}^4$,
$Z_4 = 4Z_1Z_2(X_1Z_2(X_1{}^2Z_2{}^2 + aZ_1{}^2Z_2{}^2) + bZ_1{}^3Z_2{}^3)$.

We call this formula as Montgomery-type addition formula. mECADDDBL requires 17 multiplications in general and 15 ones when a is small. In order to enhance this method to DPA-resistant, we have to use Coron's 3rd countermeasure or Joye-Tymen countermeasure. The total cost of scalar multiplication dP is estimated as $(17n+8)M+I$ in general and $(15n+10)M+I$ when a is small, where n is the bit length of the scalar d (see [8]).

2.3 Isomorphism and Isogeny

Two elliptic curves $E_1(a_1, b_1)$ and $E_2(a_2, b_2)$ are called isomorphic if and only if there exists $r \in K^*$ such that $a_1 = r^4a_2$ and $b_1 = r^6b_2$. The isomorphism is given by

$$\psi : \begin{cases} E_1 & \longrightarrow & E_2 \\ (x, y) & \longmapsto & (r^{-2}x, r^{-3}y) \end{cases} .$$

There are $(p-1)/2$ isomorphic classes.

Let $\Phi_l(X,Y)$ be a modular polynomial of degree l. Two elliptic curves $E_1(a_1,b_1)$ and $E_2(a_2,b_2)$ are called l-isogenous if and only if $\Phi_l(j_1,j_2)=0$ satisfies, where j_i are j-invariant of curve E_i for $i=1,2$. Isogenous curves have the same order. The isogeny is given by

$$\psi : \begin{cases} E_1 & \longrightarrow & E_2 \\ (x,y) & \longmapsto & \left(\frac{f_1(x)}{g(x)^2}, \frac{y \cdot f_2(x)}{g(x)^3}\right) \end{cases},$$

where f_1, f_2 and g are polynomials of degree l, $(3l-1)/2$ and $(l-1)/2$ respectively (see details in [2, Chapter VII]). By Horner's rule, the computational cost of this mapping is estimated as $(l+(3l-2)/2+(l-1)/2+5)M+I=(3l+4)M+I$.

The usage of isogeny for elliptic curve cryptosystem initially appeared for improving the order counting method (see, for example, [12]). Recently, some new applications of isogeny have been proposed, namely for improving the efficiency of the scalar multiplication [4], and for enhancing the security for a new attack [17].

Brier and Joye reported that isogeny could be used for improving the efficiency of $\mathrm{ECDBL}^{\mathcal{J}}$ [4]. Recall that if the curve parameter a of an elliptic curve is equal to -3, the cost of $\mathrm{ECDBL}^{\mathcal{J}}$ is reduced from 10 multiplications to 8 ones. If there is an integer r such that $-3=r^4a$, then we can transform the original elliptic curve to the isomorphic curve with $a=-3$. However, its success probability is about 1/2 when $p \equiv 3 \pmod 4$ or about 1/4 when $p \equiv 1 \pmod 4$. They proposed that the isogeny of the original curve could have a curve with $a=-3$.

Goubin proposed the new power analysis on ECC [7]. This attack utilizes the points $(x,0)$ and $(0,y)$ that cannot be randomized by the above two standard randomization techniques. Goubin's attack is effective on the curves that have point $(x,0)$ or $(0,y)$ in such protocols as ECIES and single-pass ECDH. The point $(x,0)$ is not on the curve with prime order because the order of $(x,0)$ is 2. The point $(0,y)$ appears on the curve if b is quadratic residue modulo p, which is computed by solving $y^2=b$. As a countermeasure to Goubin's attack, Smart utilized isogeny [17]. He proposed that if the original curve E has the point $(0,y)$, the isogenous curve E' to E could have no point $(0,y)$. If we can find E' which has no point $(0,y)$, we transfer the base point $P \in E$ to $P' \in E'$ using the isogeny $\psi : E \to E'$. Instead of computing scalar multiplication $Q=dP$, we compute $Q'=dP'$ on E' and then pull back $Q \in E$ from $Q' \in E'$ by the mapping $\psi^{-1} : E' \to E$. The mappings ψ, ψ^{-1} require $(3l+4)M+I$ respectively, so that the additional cost for this countermeasure is $(6l+8)M+2I$.

At ISC'03, we proposed the zero-value point (ZVP) attack which is extension of Goubin's attack [1]. We pointed out that if the point has no zero-value coordinate, the auxiliary registers might take zero-value. We found several points (x,y) which cause the zero-value registers and called these points as the zero-value points (ZVP). ZVP strongly depend on the structure of addition formula, and namely ZVP for the standard addition formulae are different from those for Montgomery addition formula. The points with the following conditions from ECDBL are effectively used for the ZVP attack.

- *(ED1)* $3x^2 + a = 0$ for the standard addition formulae
- *(MD1)* $x^2 - a = 0$ and *(MD2)* $x^2 + a = 0$ for Montgomery addition formula

The attacker can utilize the points that cause the zero-value registers in ECADD, however finding ZVP in ECADD is much more difficult than in ECDBL. In this paper we consider only the above points *(ED1), (MD1),* and *(MD2)).*

3 Isogeny Countermeasure against ZVP Attack

In this section we examine the countermeasure using isogeny against the ZVP attack. In order to prevent the ZVP attack, we have to choose the curve which has neither the point $(0, y)$ nor *(ED1)* for the methods using the standard addition formulae, and neither $(0, y)$, *(MD1)* nor *(MD2)* for Montgomery-type method. The degree of isogeny depends on not only a curve itself but also addition formulae. We examine the standard curves from SECG [18].

3.1 Example from SECG Curve

For example, we mention the curve secp112r1 from SECG curves [18]. secp112r1 $E : y^2 = x^3 + ax + b$ over $\mathbb{F}_p$ is defined by

$$\begin{cases} p = 4451685225093714772084598273548427, \\ a = 4451685225093714772084598273548424 = -3, \\ b = 2061118396808653202902996166388514. \end{cases}$$

This curve does not have $(0, y)$, but has *(ED1)* $3x^2 + a = 0$ as

$$(x, y) = (1, 1170244908728626138608688645279825).$$

Therefore secp112r1 is secure against Goubin's attack, but vulnerable against the ZVP attack for the methods using the standard addition formulae. However, the 7-isogenous curve $E' : y^2 = x^3 + a'x + b'$ over $\mathbb{F}_p$ defined by

$$\begin{cases} a' = 1, \\ b' = 811581442038490117125351766938682, \end{cases}$$

has neither $(0, y)$ nor *(ED1)* $3x^2 + a' = 0$. Thus E' is secure against both Goubin's attack and the ZVP attack for the methods using the standard addition formulae. We don't require isogeny defense to prevent Goubin's attack, but require the isogeny of degree 7 to prevent the ZVP attack.

3.2 Experimental Results from SECG Curves

For each SECG curve we search the minimal degree of isogeny to a curve which has neither $(0, y)$ nor ZVP as described above. If the original curve has neither

$(0, y)$ nor ZVP, we specify this degree as 1. For the standard addition formulae, we also search the minimal isogeny degree to a curve which we prefer for particularly efficient implementation, namely $a = -3$ as described in section 2. We call the former as the minimal isogeny degree and the latter as the preferred isogeny degree, and define l_{std}, l_{prf}, and l_{mnt} as follows:

- l_{std} : the minimal isogeny degree for the standard addition formulae.
- l_{prf} : the preferred isogeny degree for the standard addition formulae,
- l_{mnt} : the minimal isogeny degree for Montgomery-type addition formula.

Here we show the searching method of these degrees for the standard addition formulae.

Algorithm 1: Searching method for the standard addition formulae

Input: $E : y^2 = x^3 + ax + b$ over $\mathbb{F}_p$, $j = j$-invariant of E
Output: minimal isogeny degree l_{std} and preferred isogeny degree l_{prf}
1. Set $l \leftarrow 3$.
2. Solve the equation $\Phi_l(j', j) = 0$.
3. If the equation has no solution then go to Step 4, else then
   3.1. Construct $E' : y^2 = x^3 + a'x + b'$ where $j' = j$-invariant of E'.
   3.2. Check E' has the point $(0, y)$ and *(ED1)*.
   3.3. If E' has then go to Step 4, else then
      3.3.1. If l_{std} is null, set $l_{\mathrm{std}} \leftarrow l$.
      3.3.2. Check $r \in \mathbb{F}_p{}^*$ exists where $r^4 a' = -3 \bmod p$.
      3.3.3. If exists then set $l_{\mathrm{prf}} \leftarrow l$ and stop, else then go to Step 4.
4. If $l > 107$ then stop, else then $l \leftarrow \mathrm{nextprime}(l)$ and go to Step 2.

In this algorithm $\mathrm{nextprime}(l)$ is a function which returns the smallest prime number larger than l. For l_{mnt}, we check *(MD1)* and *(MD2)* instead of *(ED1)* in Step 3.2.

Table 1 shows isogeny degrees l_{std}, l_{prf}, and l_{mnt} for SECG curves. The number in $(\cdot)$ is the minimal isogeny degree listed in [17], which considers only Goubin's point $(0, y)$ (not the ZVP). In order to prevent the ZVP attack, some curves require higher degree of isogeny, e.g., secp112r1 for l_{std}. These isogeny degrees depend on not only the curve itself but also the addition formula, namely some curves require different isogeny degrees for the standard addition formulae and Montgomery-type addition formula. Interestingly, we have not found preferred isogeny degree up to 107 for secp112r1, secp192r1, and secp384r1.

3.3 Some Properties of ZVP Attack

Here we show some properties of the zero-value point attack.

Theorem 1. *Let E be an elliptic curve over prime field $\mathbb{F}_p$ defined by $y^2 = x^3 + ax + b$. The elliptic curve E has point $(0, y)$, if E satisfies (MD2) $x^2 + a = 0$.*

Proof. If $a = 0$ or $b = 0$ holds, then the assertion is trivial. We assume that $a \neq 0$ and $b \neq 0$. Note that $(0, y)$ exists on curve E if b is a quadratic residue in $\mathbb{F}_p^*$. Let $s \in \mathbb{F}_p^*$ be the solution of equation $x^2 + a = 0$. Condition *(MD2)* implies that there is a solution $y = t$ of equation $y^2 = s^3 + as + b$. Thus E has point $(0, t)$ due to $t^2 = s^3 + as + b = (s^2 + a)a + b = b$.

	l_{std}	l_{prf}	l_{mnt}
secp112r1	7 (1)	> 107 (1)	1 (1)
secp128r1	7 (7)	7 (7)	7 (7)
secp160r1	13 (13)	13 (13)	19 (13)
secp160r2	19 (19)	41 (41)	19 (19)
secp192r1	23 (23)	> 107 (73)	23 (23)
secp224r1	1 (1)	1 (1)	1 (1)
secp256r1	3 (3)	23 (11)	3 (3)
secp384r1	31 (19)	> 107 (19)	19 (19)
secp521r1	5 (5)	5 (5)	7 (5)

Table 1. Minimal and preferred isogeny degree for SECG curves

All curves which satisfy condition *(MD2)* have Goubin point $(0, y)$. These curves are insecure against both Goubin's attack and the ZVP attack.

Theorem 2. *Let E be an elliptic curve over prime field $\mathbb{F}_p$ defined by $y^2 = x^3 + ax + b$. The elliptic curve E satisfies condition (ED1) $3x^2 + a = 0$, if E satisfies the following three conditions: (1)$a = -3$, (2)$\#E$ is odd, and (3)p satisfies $(-3/p) = -1$, where $(\cdot/\cdot)$ is Legendre symbol.*

Proof. Since E has odd order, E does not have the point $(x, 0)$, and thus the equation $x^3+ax+b = 0$ has no root. Then the definition of discriminant Δ yields $(\Delta/p) = 1$. Note that condition $(-3/p) = -1$ implies $((b+2)(b-2)/p) = -1$ due to $\Delta = -16(4(-3)^3)+27b^2) = -3(12)^2(b+2)(b-2)$. Thus either $((b+2)/p) = 1$ or $((b-2)/p) = -1$ holds. In other words, equation $y^2 = x^3+ax+b$ with $a = -3$ and $x = \pm 1$ are solvable in y. Consequently, elliptic curve E with the above three conditions satisfies *(ED1)* $3x^2 + a = 0$.

The definition fields $\mathbb{F}_p$ that satisfy $(-3/p) = -1$ in Table 1 are secp112r1, secp192r1, and secp384r1. These curves also have odd order and satisfy $a = -3$. Therefore, these curves satisfy *(ED1)* and are vulnerable to the ZVP attack.

Since the isogenous curve has same order as E, any isogenous curve with $a = -3$ always satisfies *(ED1)* and thus is insecure against the ZVP attack. We have the following corollary.

Corollary 1. *Let E be an elliptic curve over prime field $\mathbb{F}_p$. We assume that $\#E$ is odd and $(-3/p) = -1$. Any isogeny cannot map E to the curve with $a = -3$ that is secure against the ZVP attack.*

Corollary 1 shows that it is impossible to find the isogenous curve with $a = -3$ which does not satisfy *(ED1)*, namely l_{prf}-isogenous curve, for these three curves.

4 Efficient Implementation Using Isogeny

4.1 Most Efficient Method for Each SECG Curve

We estimate the total cost of the scalar multiplication in the necessity of resistance against both Goubin's attack and the ZVP attack. This situation corresponds to the scalar multiplication in ECIES and single-pass ECDH.

Here we notice the two efficient DPA-resistant methods, namely the window-based method and Montgomery-type method. We have to use the window-based method on l_{std}-isogenous curve because this method uses the standard addition formulae. Isomorphism enables the efficient implementation with small a. Moreover, more efficient implementation with $a = -3$ can be achieved on l_{prf}-isogenous curve. On the other hand, we have to use Montgomery-type method on l_{mnt}-isogenous curve. Isomorphism also enables the efficient implementation with small a.

Therefore, we mention the following three methods:

Method 1 Window-based method with small a on l_{std}-isogenous curve,
Method 2 Window-based method with $a = -3$ on l_{prf}-isogenous curve,
Method 3 Montgomery-type method with small a on l_{mnt}-isogenous curve.

From section 2 we estimate the total cost of each method as follows:

Method 1 $T_1 = (16 \cdot 2^w + (9w + 21)k + 6l_{\mathrm{std}} - 10)M + 3I$.
Method 2 $T_2 = (16 \cdot 2^w + (8w + 21)k + 6l_{\mathrm{prf}} - 10)M + 3I$,
Method 3 $T_3 = (15n + 6l_{\mathrm{mnt}} + 18)M + 3I$.

If the isogeny degree equals to 1, the cost of isogeny ($14M + 2I$) is cut.

Table 2 shows the estimated cost for each SECG curve. Method 2 cannot be used for some curves because there is no preferred isogeny degree l_{prf} (notation '—' indicates these curves). We emphasize the most efficient method for each curve with the bold letter. The most efficient method differs on each curve because the isogeny depends on the curve and implementation method.

4.2 Efficient Scalar Multiplication Using $(0, y)$

In this section we propose another improvement for computing the efficient scalar multiplication.

In order to clearly describe our method, we categorize the improvement of efficiency into five classes, namely, (1) curve parameter (e.g. $a = -3$, $Z = 1$, etc), (2)addition chain (e.g. binary method, NAF, etc), (3)base field (e.g. optimal normal base, OEF, etc), (4)coordinate (e.g. projective coordinates, Jacobian coordinates, etc). (5)curve form (e.g. Montgomery form, Hessian form, etc). The proposed method belongs to class (1), but its improvement is related to classes (2), (4), and (5). Our improvement can be simultaneously used with other methods in class one. For sake of convenience, we discuss the improvement for the double-and-add-always method in section 2 on the curve with parameter $a = -3, Z = 1$, Jacobian coordinate, and Weierstrass form.

	Method 1	Method 2	Method 3
secp112r1	$1884M + 3I$ $(w = 4)$	—	**1690M + I**
secp128r1	$2112M + 3I$ $(w = 4)$	$1984M + 3I$ $(w = 4)$	**1980M + 3I**
secp160r1	$2604M + 3I$ $(w = 4)$	**2444M + 3I (w = 4)**	$2532M + 3I$
secp160r2	$2640M + 3I$ $(w = 4)$	$2612M + 3I$ $(w = 4)$	**2532M + 3I**
secp192r1	$3120M + 3I$ $(w = 4)$	—	**3036M + 3I**
secp224r1	$3430M + I$ $(w = 4)$	**3206M + I (w = 4)**	$3370M + I$
secp256r1	$3912M + 3I$ $(w = 4)$	**3776M + 3I (w = 4)**	$3876M + 3I$
secp384r1	**5770M + 3I (w = 5)**	—	$5892M + 3I$
secp521r1	$7462M + 3I$ $(w = 5)$	**6937M + 3I (w = 5)**	$7875M + 3I$

Table 2. Total cost of scalar multiplication to resist Goubin's attack and the ZVP attack

The main idea of the improvement is to use the point $(0, y)$ for the base point of the underlying curve, namely the point with the zero x-coordinate. The double-and-add-always method in section 2 is a left-to-right method, and thus the base point P is fixed during the scalar multiplication dP. The addition formula with the point $X = 0$ is represent as follows:

ECADD in Jacobian Coordinates with $X = 0$ ($\mathrm{ECADD}^{\mathcal{J}}_{X=0}$)
$X_3 = -H^3 + R^2$, $Y_3 = -S_1 H^3 - R X_3$, $Z_3 = Z_1 Z_2 H$,
$H = X_2 {Z_1}^2$, $S_1 = Y_1 {Z_2}^3$, $S_2 = Y_2 {Z_1}^3$, $R = S_2 - S_1$.

We denote by $\mathrm{ECADD}^{\mathcal{J}}_{X=0}$ the addition formula for ECADD in Jacobian Coordinates with $X = 0$. Formula $\mathrm{ECADD}^{\mathcal{J}}_{X=0}$ requires only 14 multiplications when $Z_1 \neq 1$ and 9 multiplications when $Z_1 = 1$.

Therefore, we have the following estimation for n-bit scalar multiplication with $a = -3, Z = 1$ using Jacobian coordinates and the double-and-add-always method in section 2. The propose scheme can achieve about 11% improvement over the scheme $X \neq 0$.

	n-bit ECC	160-bit ECC
Scheme $X \neq 0$	$19nM$	$3040M$
Scheme $X = 0$	$17nM$	$2720M$

Table 3. Comparison of efficiency with $X \neq 0$ and $X = 0$

Here we have a question about the security of choosing the base point $(0, y)$. The following theorem can be easily proven thank to the random self reducibility.

Theorem 3. *Let E be an elliptic curve over $\mathbb{F}_p$. We assume that #E is a prime order. Breaking the discrete logarithm problem with base point $(0, y)$ is as intractable as doing with a random base point.*

Proof. ($\Leftarrow$) Let $\log_{G_0} P_0$ be the discrete logarithm problem for the base point $G_0 = (0, y)$ and a point P_0. We can randomize these points by multiplying random exponents $r, s \in [1, \#E]$, namely let $G = rG_0, P = sP_0$ be randomized points. From the assumption, we can solve a discrete logarithm problem $\log_G P$, and thus the discrete logarithm $\log_{G_0} P_0 = (\log_G P)r/s \bmod \#E$.
($\Rightarrow$) Let A_0 be an oracle which solves the discrete logarithm problem for the base point $G_0 = (0, y)$, namely A_0 answers $\log_{G_0} P_0$ for a random point P_0. We try to construct algorithm A that solves the discrete logarithm problem with a random base. Algorithm A is going to compute $\log_G P$ for random inputs G, P. Algorithm A randomizes G with a random exponent $t \in [1, \#E]$ and obtains discrete logarithm $\log_{G_0} G$ by asking tG, G_0 to oracle A_0. Similarly, algorithm A obtains $\log_{G_0} P$. Then algorithm A returns the discrete logarithm $\log_G P = (\log_{G_0} P)/(\log_{G_0} G) \bmod \#E$.

From this theorem, there is no security disadvantage of using the based point $(0, y)$. Another advantage of using the base point $(0, y)$ is that memory required for base point is reduced to half.

In order to utilize the proposed method efficiently, we propose the following scenario. If we need to resist against both Goubin's attack and the ZVP attack as ECIES and single-pass ECDH, we compute the scalar multiplication on the original curve which has neither Goubin's point $(0, y)$ nor ZVP. Otherwise as ECDSA and two-pass ECDH, we compute on the isogenous curve of a small degree which has a point $(0, y)$, and map the result point to the original curve using isogeny.

We show the example of a curve to achieve this scenario. The curve $E : y^2 = x^3 + ax + b$ over $\mathbb{F}_p$ defined by

$$\begin{cases} p = 1461501637330902918203684832716283019653785059327, \\ a = 1461501637330902918203684832716283019653785059324 = -3, \\ b = 650811658836496945486322213172932667970910739301, \\ \#E = 1461501637330902918203686418909428858432566759883, \end{cases}$$

has neither $(0, y)$ nor *(ED1)* $3x^2 + a = 0$. Therefore this curve is secure against both Goubin's attack and the ZVP attack for the methods using the standard addition formulae. Then, the 3-isogenous curve $E' : y^2 = x^3 + a'x + b'$ over $\mathbb{F}_p$ defined by

$$\begin{cases} a' = 1461501637330902918203684832716283019653785059324 = -3, \\ b' = 457481734813551707109011364830625202028249398260, \end{cases}$$

has the point $G' = (0, y)$ such as

$$G' = (0, 914154799534049515652763431190255872227303582054).$$

The isogeny $\psi : E \to E'$ and $\psi^{-1} : E' \to E$ cost only $13M + I$ respectively. This cost is much smaller than improvement of the proposed method. The details of finding such a map are described in [2, Chapter VII].

5 Conclusion

We examined the countermeasure using isogeny against the ZVP attack. We showed that a class of curves (including some SECG curves) is still insecure against the ZVP attack despite the countermeasure — it can be never mapped to the efficient curve that is secure against the ZVP attack. This class satisfies the following three conditions: $a = -3$, E has odd order, and $(-3/p) = -1$. The condition $a = -3$ and E has prime order are important for security or efficiency. Thus the base field $\mathbb{F}_p$ with $(-3/p) = 1$ may be recommended.

In the addition, we compare some efficient methods of computing the scalar multiplication for each curve from SECG in consideration of the resistance against the ZVP attack. Finally we proposed a positive use of Goubin's point. If Goubin's point is used for the base point of scalar multiplication, we can improve about 11% for the double-and-add-always method.

References

1. T. Akishita and T. Takagi, "Zero-Value Point Attacks on Elliptic Curve Cryptosystem", *Information Security - ISC 2003,* LNCS 2851, pp. 218-233, Springer-Verlag, 2003.
2. I. Blake, G. Seroussi, and N. Smart, *Elliptic Curve in Cryptography,* Cambridge University Press, 1999.
3. E. Brier and M. Joye, "Weierstrass Elliptic Curve and Side-Channel Attacks", *Public Key Cryptography - PKC 2002,* LNCS 2274, pp. 335-345, Springer-Verlag, 2002.
4. E. Brier and M. Joye, "Fast Point Multiplication on Elliptic Curves through Isogenies", *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes - AAECC 2003,* LNCS 2643, pp. 43-50, Springer-Verlag, 2003.
5. J.-S. Coron, "Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems", *Cryptographic Hardware and Embedded Systems - CHES '99,* LNCS 1717, pp. 292-302, Springer-Verlag, 1999.
6. W. Fischer, C. Giraud, E. W. Knundsen, and J. -P. Seifert, "Parallel Scalar Multiplication on General Elliptic Curves over $\mathbb{F}_p$ Hedged against Non-Differential Side-Channel Attacks", IACR Cryptology ePrint Archive 2002/007. http://eprint.iacr.org/2002/007/
7. L. Goubin, "A Refined Power-Analysis Attack on Elliptic Curve Cryptosystems", *Public Key Cryptography - PKC 2003,* LNCS 2567, pp. 199-211, Springer-Verlag, 2003.
8. T. Izu, B, Möller, and T. Takagi, "Improved Elliptic Curve Multiplication Methods Resistant against Side Channel Attacks", *Progress in Cryptology - INDOCRYPT 2002,* LNCS 2551, pp. 296-313, Springer-Verlag, 2002.
9. T. Izu and T. Takagi, "A Fast Parallel Elliptic Curve Multiplication Resistant against Side Channel Attacks", *Public Key Cryptography - PKC 2002,* LNCS 2274, pp. 280-296, Springer-Verlag, 2002.
10. T. Izu and T. Takagi, "A Fast Parallel Elliptic Curve Multiplication Resistant against Side Channel Attacks", Technical Report CORR 2002-03. http://www.cacr.math.uwaterloo.ca/techreports/2002/corr2002-03.ps

11. M. Joye and C. Tymen, "Protection against Differential Analysis for Elliptic Curve Cryptography", *Cryptographic Hardware and Embedded Systems - CHES 2001,* LNCS 2162, pp. 377-390, Springer-Verlag, 2001.
12. R. Lercier and F. Morain, "Counting the Number of Points of on Elliptic Curves over Finite Fields: Strategies and Performances", *Advances in Cryptology — Eurocrypt '95,* LNCS 921, pp.79-94, Springer-Verlag, 1995.
13. B. Möller, "Securing Elliptic Curve Point Multiplication against Side-Channel Attacks", *Information Security - ISC 2001,* LNCS 2200, pp.324-334, Springer-Verlag, 2001.
14. B. Möller, "Parallelizable Elliptic Curve Point Multiplication Method with Resistance against Side-Channel Attacks", *Information Security - ISC 2002,* LNCS 2433, pp.402-413, Springer-Verlag, 2002.
15. P. L. Montgomery, "Speeding the Pollard and Elliptic Curve Methods of Factorization" , *Mathematics of Computation,* vol. 48, pp. 243-264, 1987.
16. K. Okeya and T. Takagi, "The Width-w NAF Method Provides Small Memory and Fast Elliptic Scalar Multiplications Secure against Side Channel Attacks", *Cryptographer's Track RSA Conference - CT-RSA 2003,* LNCS 2612, pp. 328-343, Springer-Verlag, 2003.
17. N. Smart, "An Analysis of Goubin's Refined Power Analysis Attack", *Cryptographic Hardware and Embedded Systems - CHES 2003,* LNCS 2779, pp. 281-290, Springer-Verlag, 2003.
18. Standard for Efficient Cryptography (SECG), *SEC2: Recommended Elliptic Curve Domain Parameters,* Version 1.0, 2000. http://www.secg.org/

On the Security of Multiple Encryption or CCA-security+CCA-security=CCA-security?

Rui Zhang[1], Goichiro Hanaoka[1]*, Junji Shikata[2], and Hideki Imai[1]

[1] Institute of Industrial Science, University of Tokyo, Japan
{zhang,hanaoka}@imailab.iis.u-tokyo.ac.jp, imai@iis.u-tokyo.ac.jp
[2] Graduate School of Environment and Information Science,
Yokohama National University, Japan
shikata@ynu.ac.jp

Abstract. In a practical system, a message is often encrypted more than once by different encryptions, here called multiple encryption, to enhance its security. Additionally, new features may be achieved by multiple encrypting a message, such as the key-insulated cryptosystems and anonymous channels. Intuitively, a multiple encryption should remain "secure", whenever there is one component cipher unbreakable in it. In NESSIE's latest Portfolio of recommended cryptographic primitives (Feb. 2003), it is suggested to use multiple encryption with component ciphers based on different assumptions to acquire long term security. However, in this paper we show this needs careful discussion, especially, this may *not* be true according to adaptive chosen ciphertext attack (CCA), even with all component ciphers CCA-secure. We define an extended model of (standard) CCA called *chosen ciphertext attack for multiple encryption* (ME-CCA) emulating partial breaking of assumptions, and give constructions of multiple encryption satisfying ME-CCA-security. We further relax CCA by introducing *weak* ME-CCA (ME-wCCA) and study the relations among these definitions, proving ME-wCCA-security can be acquired by combining IND-CCA-secure component ciphers together. We then apply these results to key-insulated cryptosystem.

1 Introduction

A practical cryptosystem often encrypts a message several times with independent secret keys or even distinct encryption schemes based on different assumptions to enhance the confidentiality of message. We call such cryptosystems multiple encryption, specifically double encryption and triple encryption for two times and three times multiple encryptions respectively. In this paper, we investigate the security notion of multiple encryption against partial breaking of underlying assumptions as well as key exposure.

* The second author is supported by a Research Fellowship from Japan Society for the Promotion of Science (JSPS).

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 360–374, 2004.

WHY MULTIPLE ENCRYPTION. It is widely believed that multiple encryption provides better security because even if underlying assumptions of some component ciphers are broken or some of the secret keys are compromised, the confidentiality can still be maintained by the remaining encryptions. Historically, sudden emergence of efficient attacks against the elliptic curve cryptosystem on supersingular curves [23, 14] and on prime-field anomalous curves [28, 33, 27] have already reminded us the necessity to do this. Especially, it is suggested by NESSIE ([25], pp. 5, line 7-11) on asymmetric encryption scheme to *"use double encryption using ACE-KEM and RSA-KEM with different OEMs gives a good range of security, based on various different assumptions", "if very long term security is important"*. Furthermore, *"Triple encryption that also uses a public-key scheme not based on number-theoretical assumptions might increase the security against future breakthrough"*. However, it seems that this needs more careful discussions.

On the other hand, multiple encryption can bring favorable additional new features to a scheme. Combination of ordinary threshold encryptions may yield new threshold encryption with various access structures. Many practical applications achieving sender anonymity via practical open network, like Mix-net [7, 19], onion routing [7] and key-insulated cryptosystems [11] are all practical examples of multiple encryption.

CONTRADICTION TO THE INTUITION. In this paper, we show that even if it consists of only *independently* selected semantically secure against adaptive chosen ciphertext attack (IND-CCA) secure components, a multiple encryption is not necessarily secure against chosen ciphertext attack (CCA) with with partial component ciphers broken. This contradicts our intuition at the first sight, but such "natural" constructions of multiple encryption can be shown easily to lose the CCA-security. Meanwhile, this result may imply CCA-security is too strong because practical schemes with "pretty good" security could be considered insecure in the sense. Then we propose a generic construction of multiple encryption scheme achieving CCA-security exactly. On the other hand, we relax security definition based on the "natural" constructions emphasizing practical usability, and investigate the relations among security notions for multiple encryption. Finally as a byproduct, we give the first generic construction of CCA-secure key-insulated cryptosystem.

1.1 Related Work

MULTIPLE ENCRYPTION AND RELATED PRIMITIVES. Multiple encryption has been used in practical schemes, for instance Triple DES. NESSIE [25] has also lately announced its recommendation to use (public key) multiple encryption with encryptions under diverse assumptions to ensure long term security. Another example is the key-insulated cryptosystem, proposed by Dodis, Katz, Xu and Yung [11], whose generic construction is actually multiple encryption of messages under a number of keys from cover free family [21].

Another important category of applications using multiple encryption are those practical implementations of anonymous channel in open network, such as, the Mix-net [19] and onion routing [7]. In these settings, several agents are appointed to transmit data from the sender to the receiver without revealing identity of the sender. Typical design of such protocols is to encrypt data under multiple public keys of these agents, which decrypt the data one layer after another until eventually reach the destination. It is essential to perform these decryption correctly, e.g., [1] has shown some practical attacks against some carelessly designed Mix-net protocols [20,18], which if translated in our language, are insecure multiple encryption.

A related notion to multiple encryption is the threshold cryptosystem [8, 32], which maintains secrecy of decryption key even if part of the secret key servers storing key shares are compromised. However, all known constructions are based on particular number theoretic assumption and can be employed to only a restrictive range of applications.

SECURITY NOTIONS. Standard definitions of public key encryption scheme are founded gradually in literature, e.g. [17, 12, 26, 4, 13]. *Semantic security,* first defined by Goldwasser and Micali [17], later refined by Goldreich [16, 15] and Watanabe, Shikata and Imai [34], captures the computational approximation of Shannon's information-theoretic security [29], regulating that it should be infeasible for any PPT (Probabilistic Polynomial Time) adversary to obtain any partial information about the plaintext of a given ciphertext. Another rather technical definition, *indistinguishability,* defines that given a ciphertext an adversary cannot distinguish which plaintext is encrypted from two plaintexts. Indistinguishability is proven to be equivalent to semantic security in several attack models, namely chosen plaintext attack (CPA), (non-adaptive) chosen-ciphertext attack (CCA1) and adaptive chosen-ciphertext attack (CCA2) [17, 16, 34, 15]. Another intricate notion, *non-malleability,* defined by Dolev, Dwork and Naor [12, 13] formulates that the adversary should not be able to create a ciphertext of a different message that is meaningfully related to the original ciphertext and non-malleability implies indistinguishability in all above three attack models. Independently in [4] and [13], indistinguishability and non-malleability are proven to be equivalent under (adaptive) chosen-ciphertext attack (herefter CCA).

CCA-security is crucial in analyzing security of protocols. Mainly it allows the adversary can make arbitrary decryption queries on any ciphertext other than the target message. However, Shoup first argues CCA-security is too stringent for practical schemes and suggests "benign malleability" in the proposal for ISO public key encryption standard [31], as a relaxation for CCA model. An, Dodis and Rabin [3] give similar discussion under the name "generalized-CCA" (gCCA). In these two relaxed definitions, a relation function checks and rejects "obvious" decryption queries decrypted to the target message. Canetti, Krawczyk and Nielsen recently propose another relaxation, RCCA (Replayable CCA), which is strictly weaker than gCCA in most of cases [6].

PREVIOUS WORK ON MULTIPLE ENCRYPTIONS AND RELATIONS. Multiple encryption was addressed by Shannon as early as [29] under the name "product cipher", and in [9, 24, 2] in context of symmetric key cryptosystems. Massay and Maurer [22] have also studied the problem under the name "cascade cipher" . However, all above work lacks considerations for CCA-security and is not adequate, for applying their underlying notions to public key setting straightforwardly, even only to the sequential case.

In ongoing work of [10], Dodis and Katz, independently of our work, propose another generic construction of CCA-secure multiple encryption. The security of their scheme can be proven in the standard model and can be generated to threshold settings. The difference lies in that first their scheme needs CCA-secure components while we only require component ciphers to be CPA secure. Besides, threshold setting seems not fit for our main goal "to enhance security of single component cipher". So far, they have presented their work in Rump Session in Crypto'03, Aug. 2003, while an earlier version [36] of our work was publicly announced in SCIS'03, Jan. 2003.

1.2 Our Contributions

Our contributions lie in following aspects:

MODEL AND SECURITY DEFINITION OF MULTIPLE ENCRYPTION. We give the first formal model regarding public key multiple encryption. To the best of our knowledge, no previous work has strict formalization including CCA-security on this respect, and actually our model can be extended to both public key and symmetric key based cryptosystems. Our model consorts the modular design: combining "secure" component ciphers to have a "secure" multiple encryption. As a theoretical extension of traditional security definitions, we give the corresponding security definitions on multiple encryption based on indistinguishability and non-malleability against different attacks, especially chosen ciphertext attack (ME-CCA). Without loss of generality, breaking underlying assumptions of component ciphers can be esuriently modelled as the secret key is leaked to the adversary. Also some analyses here can be applied to symmetric key schemes.

VULNERABILITY OF NATURAL MULTIPLE ENCRYPTION. We demonstrate generic attacks against some "natural" constructions of multiple encryption schemes with each component IND-CCA-secure, by an adversary that breaks the indistinguishability of the scheme with only accesses to the Decryption Oracle and the Key Exposure Oracle. In fact, such adversary even breaks the onewayness. This suggests the necessity that multiple encryption should be treated as a separate primitive from single encryption.

SECURE CONSTRUCTION OF MULTIPLE ENCRYPTION. We build multiple encryption schemes satisfying "strong" security, e.g. CCA from those satisfying only "weak" security, e.g., CPA. Though this task can be achieved using general zero-knowledge proof or one-time signature, considering efficiency of practical schemes, we design a scheme that is provably secure in the random oracle model.

RE-DEFINING SECURITY OF MULTIPLE ENCRYPTION. IND-CCA-security has been treated as standard definition for single encryption, which is shown modular design can be achieved for cryptographical protocols in the UC framework [5]. However, our analysis shows CCA-security may be too stringent since even IND-CCA-secure components would result in a CCA insecure multiple encryption for most of "natural" constructions. We argue the CCA-security definition is too strong for defining the multiple encryptions. As a reasonable relaxation, we give a new security definition named *weak chosen ciphertext attack for multiple encryption* (ME-wCCA) that is sufficient in most of interesting cases.

SECURITY NOTIONS OF MULTIPLE ENCRYPTION. We study the relations among different security definitions for multiple encryption. We believe a good analysis of these relations will help protocol designer more than simply give a specific construction based on concrete mathematical assumptions. Security definitions, namely indistinguishability and non-malleability, are formulated under different attack models. We show indistinguishability and non-malleability are still equivalent under ME-CCA, which corresponds to previous results: A multiple encryption degenerates to an ordinary public key cryptosystem, if there is only one component cipher in it. Similar relation holds for the relaxed definitions.

APPLICATION TO KEY INSULATED ENCRYPTION. We reconsider the chosen ciphertext security of key-insulated encryption. It is only previously known in [11] that a generic construction exists provably secure against CPA attack. In this paper, we show that their scheme is in fact provably secure in the relaxed wCCA model, which reasonably supports the correctness and practical usability of their scheme. We further give a generic construction meeting exact CCA-security (in the random oracle model). We point out this is the first generic construction of CCA-secure key-insulated cryptosystem ever reported.

2 Multiple Encryption

Informally a multiple encryption is to encrypt a message by multiple cryptosystems. A multiple encryption scheme $\mathcal{ME}$ is generated by component ciphers.

Specification Multiple encryption is a cryptosystem composed by separate component ciphers, each of which may be independent. Suppose $\{\mathcal{E}_i\}_{1\le i\le n}$ is a set of *compatible* component ciphers, where for $\mathcal{E}_i$,

- $\mathsf{Enc\text{-}Gen}_i$ a probabilistic key-generation algorithm, with the input (1^k) and the internal coin flipping produces a public-secret key pair (pk_i, sk_i);
- Enc_i an encryption algorithm, with an input message $m_i \in \mathcal{M}_i$ and the public key pk_i, with the internal coin flipping, outputs a ciphertext $c_i \in \mathcal{C}_i$;
- Dec_i a decryption algorithm, which is a deterministic algorithm, with the input ciphertext c_i and the secret key sk_i, outputs a message m_i or "$\perp$".

A multiple encryption is a 3-tuple algorithm (MEnc-Gen,MEnc,MDec), where each algorithm may be combined from a number of public key cryptosystems with a unifilar connecting order. MEnc-Gen invokes every $\mathsf{Enc\text{-}Gen}_i$, and writes their outputs to a key list with public keys $PK = (pk_1, ..., pk_n)$ and secret keys $SK = (sk_1, ..., sk_n)$. MEnc with an input message M from message space $\mathcal{M}$ and PK, performs encryption MEnc on M by invoking a list of component encryption algorithms, eventually outputs a ciphertext $C \in \mathcal{C}$. The decryption algorithm MDec takes (C, SK) as input and outputs M, or "$\perp$" if C is invalid. We also denote in brief the encryption algorithm as MEnc(M ; COIN) (or MEnc(M)), and the decryption algorithm as MDec(C) in clear context, where COIN stands for the randomness used the multiple encryption. Essentially, we have two typical constructions: *parallel construction,* e.g., the generic construction given in [11], which the message is first split into shares by secret sharing then encrypted separately; *sequential construction,* e.g., the cascade cipher studied in [22], the message is encrypted by one component cipher then encrypted by another, and eventually forms the ciphertext. By combining these two constructions, we get a hybrid construction, which we refer to hereafter as "natural" construction.

3 Chosen Ciphertext Security for Multiple Encryption

Partially breaking of underlying assumptions (key exposure) is usually not considered in the security of a normal public key encryption scheme, such as IND-CCA, whereas a multiple encryption should remain secure even when most of the underlying assumptions are broken. Since this gap cannot merge sometimes, modifications should be performed to the (standard) CCA-security definition in order to catch this act. We here introduce an additional oracle into standard CCA game to emulate this scenario: a Key Exposure Oracle that upon the adaptive request of the adversary, reveals secret keys of the component ciphers to the adversary. Note that more has been considered in our model than mere key exposure and the situations are more complicated.

ORACLE ACCESS RULES. There are three oracles in our model: An Encryption Oracle $\mathcal{EO}$, which upon calling with input (M_0, M_1), returns C_b, the encryption of M_b, where $b \in \{0, 1\}$ decided by internal coin flipping. A Decryption Oracle $\mathcal{DE}$, upon decryption query C, outputs M = MDec(C), if $C \neq C_b$; otherwise, "$\perp$". A Key Exposure Oracle, upon calling with i as one index of entire n component ciphers, $1 \leq i \leq n$, returns the corresponding secret key sk_i. The adversary can access three oracles in any order at any time of its choice, but it can only query $\mathcal{EO}$ once and $\mathcal{KE}$ at most $n - 1$ times.

Definition 1 (IND-ME-CCA). *Assume any PPT adversary play the following game with a multiple encryption $\mathcal{ME}$. First key generation algorithm* MEnc-Gen *is run. The public key $PK = \{pk_i \mid i = 1, \ldots, n\}$ is then given to an Encryption Oracle $\mathcal{EO}$ and the adversary. The secret key $SK = \{sk_i | i = 1, \ldots, n\}$ is given to a Decryption Oracle $\mathcal{DO}$ and a Key Exposure Oracle $\mathcal{KE}$. The adversary chooses to access the three oracles in any order and at any time. According to the*

timing of access to $\mathcal{EO}$, the adversary's strategy is divided into two algorithms $(\mathcal{A}_{\mathsf{find}}, \mathcal{A}_{\mathsf{guess}})$, where $\mathcal{A}_{\mathsf{find}}$ tries to find (M_0, M_1) to submit to $\mathcal{EO}$ which returns C_b, and $\mathcal{A}_{\mathsf{guess}}$ tries to output a guess on b. If the difference of the success probability of the adversary $\mathcal{A}$ compared to random guess in the IND-ME-CCA *game is negligible:*

$$\Pr\left[b = \tilde{b} \middle| \begin{array}{c}(PK, SK) \leftarrow \mathsf{MEnc\text{-}Gen}(1^k), (M_0, M_1, \alpha) \leftarrow \mathcal{A}_{\mathsf{find}}^{\mathcal{KE}, \mathcal{DO}}(PK), \\ b \stackrel{R}{\leftarrow} \{0,1\}, C_b \leftarrow \mathsf{MEnc}(M_b), \tilde{b} \leftarrow \mathcal{A}_{\mathsf{guess}}^{\mathcal{KE}, \mathcal{DO}}(C_b, \alpha)\end{array}\right] \leq \frac{1}{2} + \mathsf{neg}(k)$$

then we call this $\mathcal{ME}$ IND-ME-CCA*-secure.*

Non-malleability of multiple encryption against CCA (NM-ME-CCA) is similar to IND-ME-CCA except that the adversary succeeds by outputting a new ciphertext with is "meaningfully" related to the challenge ciphertext. That is, suppose R is a prescribed relation, then the adversary wins, if the adversary could output a different ciphertext C' from the challenge ciphertext C_b, with two plaintexts decrypted from C' and C_b satisfying R (R outputs TRUE).

Definition 2 (NM-ME-CCA). *Denote $\mathbb{M}, \mathbb{C}$ as sets of plaintexts and ciphertexts being empty initially, respectively. According to the above access rules for the three oracles, if any PPT adversary in the following game has success probability negligibly close to 1/2, we call the multiple encryption scheme* NM-ME-CCA*-secure.*

$$\Pr\left[b = 1 \middle| \begin{array}{c}(PK, SK) \leftarrow \mathsf{MEnc\text{-}Gen}(1^k), (M_0, M_1, \alpha) \leftarrow \mathcal{A}_1^{\mathcal{KE}, \mathcal{DO}}(PK), \\ C_b \leftarrow \mathsf{MEnc}(M_1), (R, \mathbb{C}) \leftarrow \mathcal{A}_2^{\mathcal{KE}, \mathcal{DO}}(C_b, \alpha), \\ \mathbb{M} \leftarrow \mathsf{MDec}(\mathbb{C}), (C_b \notin \mathbb{C}) \wedge (\perp \notin \mathbb{M}) \wedge R(M_b, \mathbb{M})\end{array}\right] \leq \frac{1}{2} + \mathsf{neg}(k)$$

These definitions are also applicable to chosen plaintext attack CPA by letting $\mathcal{DO}$ always output an empty string on any decryption query, which results in the definition of *chosen plaintext attack for multiple encryption* ME-CPA. Analogously, we can define IND-ME-CPA, NM-ME-CPA. By fixing the number of component ciphers $n = 1$ in the dedition of IND-ME-CCA (or NM-ME-CCA), we obtain definition of the standard IND-CCA (or NM-CCA).

4 Insecurity of Natural Constructions

Given each component IND-CCA-secure, let's consider the following problem: Is the above "natural" construction IND-ME-CCA-secure? Rather disappointing, the answer is negative. All "natural" constructions seem insecure without further treatments.

BASIC ANALYSIS. At the first glance, one may think all multiple encryption schemes from such construction should be secure, since each component is chosen independently from each other and satisfies strong security notion IND-CCA, then all outputs will be indistinguishable from random sequence. However, this reasoning is fallacious. The flaw is in that this does not consider the case that

the adversary can make use of $\mathcal{DO}$. In this case $\mathcal{DO}$ can be very helpful because every ciphertext different from the original can be decrypted and returned according to the definition of CCA attack. Then all the adversary needs to do is to modify the challenge ciphertext to a "new" one but decrypt to the same message, and submit it to the Decryption Oracle $\mathcal{DO}$. In the (standard) CCA setting, the adversary cannot do this easily because the secret key is kept privately. However, in ME-CCA setting, partial key can be exposed by the Key Exposure Oracle $\mathcal{KE}$, moreover, since every component is semantically secure, as it must be probabilistic, where there exist at least two valid ciphertexts $C_0, C_1 \in \mathcal{C}$ with $\mathsf{MDec}(C_0) = \mathsf{MDec}(C_1) = M$, where $M \in \mathcal{M}$ is any valid plaintext. Furthermore, we have the following theorem (The proof can be found in the full version of this paper [35].).

Theorem 1. There exists *insecure* multiple encryption in the sense of IND-ME-CCA, even if it contains only independent IND-CCA-secure component ciphers.

DISCUSSION. The theorem shows only the case of indistinguishability under ME-CCA attack. We briefly explain the case of *onewayness* against chosen ciphertext attack for multiple encryption, denoted as OW-ME-CCA. Onewayness can be informally described as: given ciphertext *C,* output the plaintext *M.* It is a strictly weaker notion than indistinguishability. However, the proof of Theorem 1 tells us that not only IND-ME-CCA, but also onewayness may *not* be maintained in ME-CCA model, even if all the components are CCA-secure. On the other hand, we can see such natural schemes are malleable because the adversary can easily produce a "new" ciphertext with a proper key exposure query and simulates the Encryption Oracle. NM-ME-CCA-security better explains why the adversary can launch that attack: it actually has produced a ciphertext with relation that it contains the same plaintext to the challenge ciphertext. NM-ME-CCA-security is not trivially obtainable in such situations, either.

5 A Generic Construction for Secure Multiple Encryption

We have shown that the simple modular design without further treatment of multiple encryption is not sufficient to yield ME-CCA-security. Then two questions arise naturally: First, does a ME-CCA-secure multiple encryption exit? Second, whether a generic construction with ME-CCA-security can be combined from component ciphers with weaker security, e.g., *onewayness against chosen plaintext attack* (OW-CPA) security? We answer both questions by giving a generic construction combining component ciphers of weak security (OW-CPA) to ME-CCA-secure multiple encryption.

For the "natural" constructions, ME-CCA-security is hard to achieve with simple connections of component ciphers because partial exposure of the secret keys will always cause malleability of ciphertexts. This prompts us the necessity to check the randomness used in encryption to ensure the validity of all parts of a ciphertext before outputting the plaintext. Suppose all randomness used in

the encryption can be verified during decryption, then the Decryption Oracle in fact does not help the adversary: If the adversary can pass the randomness verification, with overwhelming probability, it has already known all the randomness used. This can further be achieved by embedding all randomness into the plaintext, then consistence of all randomness can be verified in the decryption phase, i.e., the adversary must be forced to have known the corresponding plaintext when it submits a valid ciphertext query. Then a multiple encryption will be secure if an adversary cannot break all underlying component ciphers.

5.1 Secure Construction of Multiple Encryption

ME-CCA constructions based on any public key encryption components with OW-CPA security that is satisfied by most practical public key encryption schemes. Recall $\mathcal{E}_i$ is the i-th component cipher of the multiple encryption, $\mathsf{Enc}_i(m_i, pk_i; \mathtt{COIN}_i)$ and $\mathsf{Dec}_i(c_i, sk_i)$ are the encryption algorithm and decryption algorithm for $\mathcal{E}_i$ (in short $\mathsf{Enc}_i(m_i; \mathtt{COIN}_i)$ and $\mathsf{Dec}_i(c_i)$, respectively), where pk_i is the public key and sk_i is the secret key of $\mathcal{E}_i$ (see section 2). We further design the following construction. Denote $H_i : \{0,1\}^* \to \{0,1\}^{k_i}$ (k_i is the length of necessary random coin for $\mathcal{E}_i$) and $G_i : \{0,1\}^* \to \{0,1\}^{l_i}$ (l_i is the length of c_{i2}) as random functions. For parallel multiple, one can consider the following construction:

Key-Generation $\mathsf{MGen\text{-}Enc}(1^k)$**:** $(pk_i, sk_i) \leftarrow \mathsf{Gen\text{-}Enc}_i$, for $1 \le i \le n$; $PK = (pk_1, ..., pk_n)$, $SK = (sk_1, ..., sk_n)$.

Encryption $\mathsf{MEnc}(M, PK)$**:** $(m_1, ..., m_n) \stackrel{\mathsf{AONT}}{\leftarrow} \mathcal{T}(M)$. $r_i \in_R \{0,1\}^*$, for $1 \le i \le n$. For i-th component cipher: $c_{i1} \leftarrow \mathsf{Enc}_i(r_i; H_i(M, r_1, ..., r_n))$, $c_{i2} \leftarrow G_i(r_i) \oplus m_i$, $c_i = (c_{i1}, c_{i2})$, $1 \le i \le n$. Outputs $C = (c_1, ..., c_n)$ as ciphertext.

Decryption $\mathsf{MDec}(C, SK)$**:** $r_i \leftarrow \mathsf{Dec}_i(\bar{c}_{i1})$, $\bar{m}_i = G(\bar{r}_i) \oplus \bar{c}_{i2}$, $1 \le i \le n$. Outputs $\bar{M} \leftarrow \mathcal{I}(\bar{m}_1, ..., \bar{m}_n)$ as plaintext if $\bar{c}_{i1} = \mathsf{Enc}_i(\bar{r}_i; H_i(\bar{M}, \bar{r}_1, ..., \bar{r}_n))$, otherwise "$\perp$".

We prove the following theorem holds for above construction, whose proof can be found in the full version of this paper [35]. Based on the same idea, one can design a secure construction for sequential multiple encryption, of which an example can be found in [35].

Theorem 2. *Multiple encryptions from above constructions are secure* IND-ME-CCA-secwre *in the random oracle model.*

DISCUSSION. One complementary remark should be addressed on the *uniformity* of underlying primitives. What we have considered so far is mainly non-deterministic component ciphers. For deterministic primitive public key encryption, e.g., RSA, above construction is not sufficient, however, it can be modified to fit this transform. Furthermore, if all the component ciphers are deterministic, the task is easier: just connect them together and set proper padding schemes as pre-procession of the message, like OAEP+ [30], and form the whole multiple

encryption with parallel construction with compatible input domain, or sequential connecting one after another. AONT can be even replaced by OAEP+. This construction should also be secure because if the encryption primitive is deterministic, an adversary cannot re-encrypt the corresponding parts of a ciphertext into valid new part to produce another ciphertext even if it seizes corresponding secret keys. We shall give formal analysis regarding the deterministic encryption primitive in the forthcoming work.

6 New Security Definitions for Multiple Encryption

It seems contradictive to our intuition that though component ciphers are independent, even onewayness may lose with just simple connection of independently chosen ciphers. However, if we follow the CCA-security, it is doomed to appear completely insecure. From another aspect, it suggests that CCA-security may be somehow excessively strong. In the real world, it is unreasonable that $\mathcal{DO}$ helps such obvious attacks. A well-known example states that a new cipher S' constructed from a CCA-secure cipher S, where a harmless bit is appended to the ciphertext of S and is discarded during decryption, is no longer secure in the sense of CCA. In fact such attack to S' should be easily judged and have "no significant difference" in most of interesting cases. When $\mathcal{DO}$ encounters such queries, it should easily determine whether this is really a "new" ciphertext, by just looking at the ciphertext.

6.1 Relaxing Security Definition Regarding Multiple Encryption

CCA-security might be too strong and is not always necessary, as pointed out in [31, 3, 6], among which, Shoup's "benign malleability" [31] and An, Dodis and Rabin's "gCCA" [3] are basically equivalent: a relation function $\mathcal{RF}$ helps the Decryption Oracle against obvious attacks. In gCCA definition, the relation function performs as follows: if $\mathcal{RF}(c, c') = \mathsf{TRUE} \Rightarrow \mathsf{Dec}(c) = \mathsf{Dec}(c')$. The opposite direction does not hold, otherwise, the relation function can be used as an oracle breaking the indistinguishability. There must be $\exists\ (c, c')$, such that $\mathcal{RF}(c, c') = \mathsf{FALSE}$, with $\mathsf{Dec}(c) = \mathsf{Dec}(c')$ (refer [3] for more details). Canetti, Krawczyk and Nielsen [6] recently propose another relaxation, called "replayable chosen ciphertext attack" (RCCA), with most of cases strictly weaker than gCCA.

To rule out the definitional limitation of CCA-security in multiple encryption setting, we also introduce a relaxed definition called *"weak chosen ciphertext attack for multiple encryption"* (ME-wCCA). In the definition of wCCA, there is a relation function $\mathcal{RF}^*$ is computed by invoking $\mathcal{RF}_i$ $(1 \leq i \leq n)$ during the decryption process inside $\mathcal{DO}$, with initial value of each $\mathcal{RF}_i$ set to FALSE, where $\mathcal{RF}_i$ is the relation function defined according to gCCA-security for i-th component cipher $\mathcal{E}_i$. $\mathcal{RF}_i(c_i, c_i') = \mathsf{TRUE} \Rightarrow \mathsf{Dec}(c_i) = \mathsf{Dec}(c_i')$. Whenever $\mathcal{RF}_i = \mathsf{TRUE}$ for some i, $\mathcal{RF}^*$ halts and returns TRUE to $\mathcal{DO}$ immediately. Once receiving TRUE, $\mathcal{DO}$ outputs "$\perp$" to the adversary. Informally, if $\mathcal{RF}^*$ finds a part (may be the intermediate decryption result) of the query ciphertext

looks "the same" as the corresponding part of the challenge ciphertext, it tells the Decryption Oracle to reject this decryption query. Since the rules for oracle access is the same, the definition of IND-ME-CCA only needs to be modified a little to adapt to IND-ME-wCCA.

We stress that ME-wCCA-security is a reasonable relaxation for CCA-security. This notion is basically an extension of gCCA-security. By restricting a multiple encryption to only one component cipher, IND-ME-wCCA becomes IND-gCCA.

Definition 3 (IND-ME-wCCA). *In this game, every thing is the same except the operation of the Decryption Oracle $\mathcal{DO}$. The Decryption Oracle $\mathcal{DO}$ is equipped with a Relation Function $\mathcal{RF}^*$ inside, which is computable in polynomial time. The scheme is secure if any probabilistic polynomial time adversary has success negligibly close to 1/2.*

$$\Pr\left[b=\tilde{b}\left|\begin{array}{c}(PK,SK)\leftarrow \mathsf{MEnc\text{-}Gen}(1^k),(M_0,M_1,\alpha)\leftarrow \mathcal{A}_{\mathsf{find}}^{\mathcal{KE},\mathcal{DO}^{\neg\mathcal{RF}^*}}(PK),\\ b\overset{R}{\leftarrow}\{0,1\},C_b\leftarrow \mathsf{Enc}(M_b),\tilde{b}\leftarrow \mathcal{A}_{\mathsf{guess}}^{\mathcal{KE},\mathcal{DO}^{\neg\mathcal{RF}^*}}(C_b,\alpha)\end{array}\right.\right]\leq\frac{1}{2}+\mathsf{neg}(k)$$

The following lemma shows that IND-ME-wCCA-secure multiple encryption can be acquired from IND-gCCA-secure component ciphers (for proof see [35]).

Lemma 1. *A multiple encryption scheme $\mathcal{ME}$ is* IND-ME-wCCA*-secure w.r.t. $\mathcal{RF}^*$ by any of three basic constructions, if each component cipher $\mathcal{E}_i$ is* IND-gCCA*-secure w.r.t relation function $\mathcal{RF}_i$, $1\leq i\leq n$. $\mathcal{RF}^*$ is defined as $\mathcal{RF}^*(C, C')=\mathsf{TRUE}$, such that $\mathcal{RF}_i(c_i,c_i')=\mathsf{TRUE}$ for some i, $1\leq i\leq n$, where c_i, c_i' are two ciphertexts of $\mathcal{E}_i$, and C, C' are the corresponding ciphertexts for $\mathcal{ME}$.*

Since IND-CCA always implies IND-gCCA, we have the following theorem:

Theorem 3. *If all component ciphers are* IND-CCA*-secure and chosen independently according to above "natural" constructions, then the resulting multiple encryption is* IND-ME-wCCA*-secure.*

In fact, each attack per theorem 1 can construct a new ciphertext with the same plaintext. Since non-malleability is an arduous goal for multiple encryption, we define relaxed gNM-ME-CCA similar to IND-ME-wCCA. Informally, the definition limits that the adversary does not win as long as it outputs with a new ciphertext with the equivalence relation regulated by the relation function to the challenge ciphertext, where the relation function is defined analogously to that of IND-ME-wCCA.

Definition 4 (gNM-ME-CCA). *A multiple encryption scheme is generalized-non-malleable against* ME-CCA *attack if for any PPT adversary, which is assisted by Decryption Oracle $\mathcal{DO}$, and a Key Exposure Oracle $\mathcal{KE}$, it cannot produce a new ciphertext with relation other than what the Relation Function $\mathcal{RF}^*$ specifies with non-negligible probability, where $\mathcal{RF}^*$ is defined identical to* ME-wCCA*. Denote $\mathbb{M},\mathbb{C}$ as sets of plaintexts and ciphertexts being empty initially, respectively.*

$$\Pr\left[b=1\left|\begin{array}{c}(PK,SK)\leftarrow \mathsf{MEnc\text{-}Gen}(1^k),(M_0,M_1,\alpha)\leftarrow \mathcal{A}_1^{\mathcal{KE},\mathcal{DO}}(PK),\\ C_b\leftarrow \mathsf{MEnc}(M_1),(R,\mathbb{C})\leftarrow \mathcal{A}_2^{\mathcal{KE},\mathcal{DO}}(C_b,\alpha,M_0,M_1),\\ \mathbb{M}\leftarrow \mathsf{MDec}(\mathbb{C}),(C_b\notin\mathbb{C})\wedge(\perp\notin\mathbb{M})\wedge R(M_b,\mathbb{M})\wedge(R\neq\mathcal{RF}^*)\end{array}\right.\right]\leq\frac{1}{2}+\mathsf{neg}(k)$$

gNM-ME-CCA is a strictly weaker notion than NM-ME-CCA-security (cf. IND-ME-wCCA to IND-ME-CCA).

7 Relations among Security Definitions

In this section, we discuss the relations among security definitions of multiple encryptions. The good news is that in multiple encryption scenario indistinguishability and non-malleability are still equivalent under ME-CCA attacks (IND-ME-wCCA is equivalent to gNM-ME-CCA). The proofs of these theorems are left to the full version of this paper [35].

Theorem 4. IND-ME-CCA $\Leftrightarrow$ NM-ME-CCA

Theorem 5. IND-ME-wCCA $\Leftrightarrow$ gNM-ME-CCA

Theorem 6. IND-ME-wCCA $\Rightarrow$ IND-ME-CPA, IND-ME-CPA $\not\Rightarrow$ IND-ME-wCCA.

8 Applications to Key-Insulated Cryptosystem

The key-insulated cryptosystem is proposed by [11] to protect cryptosystems against partial key exposure. In such system, encryption is done in an insecure user device. Additionally, there is a physically secure server that stores a master key. With the help of this server, user keys are updated periodically so that compromise of user keys in some periods does not affect the system in other periods. In [11], a generic construction is proposed based on arbitrary semantically secure public key encryption against *chosen plaintext attack.* Recall that the authors of [11] do not claim their generic construction CCA-secure.

At the first look, because of the property of cover-free family even if the secret keys are compromised in t periods, at most $t-1$ secret keys of a period other than these t are known to the adversary. Since the message is split into shares by AONT, we know it is computationally infeasible to break the indistinguishability even after viewing part of the sub-messages generated by AONT. However, an adversary actually can bypass the hard task and just needs to try to modify the challenge ciphertext using known secret keys in order to get help from the Decryption Oracle $\mathcal{DO}$. In fact, it can obtain any secret key sk_j by sending adaptive query to the Key Exposure Oracle $\mathcal{KE}$ for sk_j in some period i with $j \in S_i$. Then it can decrypt $c_j = \mathsf{Enc}_j(m_j)$, and re-encrypt it. It can always succeed to produce $c'_j = \mathsf{Enc}_j(m_j)$ with $c'_j \neq c_j$, since according to the system settings, all component ciphers are semantically secure. Now the adversary can replace c_j with c'_j and submit this "new" ciphertext C' to $\mathcal{DO}$, which will return the corresponding message M. This attack works for any period i.

The original generic construction of [11] does not satisfy chosen ciphertext attack security, actually if every component cipher is chosen IND-CCA-secure, this generic construction is actually IND-ME-wCCA-secure (Theorem 3). We note that this scheme still provides very practical security.

8.1 CCA-Secure Key-Insulated Cryptosystem

The feasibility of constructing a CCA-secure key-insulated cryptosystem (parallel multiple encryption) has already been shown in section 5.1. We are only fascinated at whether given IND-CCA-secure ciphers as building blocks, a parallel construction can be transformed to a CCA-secure key-insulated cryptosystem with minimum modification. Recall $\mathtt{coin}_i$ is the auxiliary randomness input for encryption component $\mathcal{E}_i$. Let $\mathtt{coin}_i = h(\mathtt{r}||Index_i)$, where $\mathtt{r}$ is a random number, $Index_i$ is the description of i-th component and h is a random function. The Encryption is $C = \mathsf{MEnc}(M||r; (\mathtt{coin}_1, ..., \mathtt{coin}_n))$, especially for IND-CCA component $\mathcal{E}_i$, $\mathsf{Enc}_i(m_i; \mathtt{coin}_i)$ where m_i is generated from AONT with input $M||r$. Decryption process becomes: for a ciphertext C', $M'||\mathtt{r}' = \mathsf{MDec}(C')$, output M' only if $c'_i = \mathsf{Enc}_i(m_i; h(\mathtt{r}'||Index_i))$ is well formed, for every $1 \leq i \leq n$. Whenever it is detected that a ciphertext has used invalid randomness, the Decryption Oracle rejects this query immediately.

It is easy to see this scheme satisfies the security definition of [11] under CCA attack. The proof is easy and will be omitted here. We point out this is actually the *first* generic construction of key-insulated cryptosystem enjoying CCA-security (Another generic construction for CCA-secure key-insulated cryptosystem will be given by Dodis and Katz in their upcoming work, whose security can be proven in the standard model.). In fact, this transform turns IND-ME-CPA secure multiple encryptions into IND-ME-CCA-secure ones.

Acknowledgement

The authors would thank Masayuki Abe, Yevgeniy Dodis, Yumiko Hanaoka, Jonathan Katz, Kazukuni Kobara and Moti Yung for fruitful discussions. The authors would also like to thank anonymous referees for invaluable comments.

References

[1] M. Abe and H. Imai. Flaws in some robust optimistic mix-nets. In *ACISP'03,* volume 2727 of *LNCS,* pages 39 – 50. Springer-Verlag, 2003.

[2] B. Aiello, M. Bellare, G. Di Crescenzo, and R. Venkatesan. Security amplification by composition: the case of doubly-iterated, ideal ciphers. In *Crypto'98,* volume 1462 of *LNCS,* pages 390–407. Springer-Verlag, 1998.

[3] J.H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In *Eurocrypt'02,* volume 2332 of *LNCS,* pages 83–107, Springer-Verlag, 2002.

[4] M. Bellare, A. Desai, D. Pointcheval, and P. Rogway. Relations among notions of security for public-key encryption schemes. In *Crypto'98,* volume 1462 of *LNCS.* Springer-Verlag, 1998.

[5] R. Canetti. Composable security: A new paradigm for cryptographic protocols. In *42nd FOCS,* pages 136–145, 2001.

[6] R. Canetti, H. Krawczyk, and J. Nielsen. Relaxing chosen-ciphertext security. In *Crypto'03.* Full version available: `http://eprint.iacr.org/2003/174/`, 2003.

[7] D. Chaum. Untraceable electronic mail, return address, and digitalpseudonyms. *Communication of the ACM,* 24:84–88, 1981.
[8] Y. Desmedt and Y. Frankel. Threshold cryptosystems. In *Crypto '89,* volume 435 of *LNCS,* pages 307–315. Springer-Verlag, 1989.
[9] W. Diffie and M.E. Hellman. Exhaustive cryptananlysis of NBS data encryption standard. *IEEE Computer Magazine,* 10(6):74–84, June 1977.
[10] Y. Dodis and J. Katz. On the chosen ciphertext security of multiple encryption. In *Rump session of Crypto '03, manuscript available from the authors,* 2003.
[11] Y. Dodis, J. Katz, S. Xu, and M. Yung. Key-insulated public key cryptosystems. In *Eurocrypt'02,* volume 2332 of *LNCS,* pages 65–82. Springer-Verlag, 2002.
[12] D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. In *23rd STOC,* pages 542–552. ACM, 1991.
[13] D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. In *SIAM Journal of Computing,* volume 30. ACM, 2000.
[14] G. Frey and H.G. Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Mathematics of Computation,* 62(206):865–874, 1994.
[15] O. Goldreich. *Foundations of Cryptography,* volume 2 (third posted version). Aavailable at: http://www.wisdom.weizmann.ac.il/~oded/PSBookFrag/enc.ps.
[16] O. Goldreich. *Foundations of Cryptography,* volume 1. Cambridge University Press: New York, 2001.
[17] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Science,* (28):270–299, 1984.
[18] P. Golle, S. Zhong, D. Boneh, M. Jakobsson, and A. Juels. Optimistic mixing for exit-polls. In *Asiacrypt'02,* volume 2501, pages 451–465. Springer-Verlag, 2002.
[19] M. Jakobsson. A practical mix. In *Eurocrypt'98,* volume 1403 of *LNCS,* pages 448–461. Springer-Verlag, 1998.
[20] M. Juels and M. Jakobsson. An optimally robust hybrid mix network. In *20th annual ACM Symposium on Principles of Distributed Computation,* 2001.
[21] R. Kumar, S. Rajagopalan, and A. Sahai. Coding constructions for blacklisting problems. In *Crypto'99,* volume 1666 of *LNCS,* pages 609–623. Springer-Verlag, 1999.
[22] U.M. Maurer and J.L. Massey. Cascade ciphers: The importance of being first. *Journal of Cryptology: the journal of the International Association for Cryptologic Research,* 6(1):55–61, 1993.
[23] A. Menezes, T. Okamoto, and S. Vanstone. Reducing elliptic curve logarithms to lgarithms in a finite field. *IEEE Trans. on Information Theory,* 39:1639–1646, 1993.
[24] R. Merkle and M. Hellman. On the security of multiple encryption. *Communications of the ACM,* 24(7):465–467, 1981.
[25] NESSIE. NESSIE Portfolio of recommended cryptographic primitives (Latest version: Feb. 2003). Available at: https://www.cosic.esat.kuleuven.ac.be/nessie/ deliverables/decision-final.pdf.
[26] C. Rackoff and D. Simon. Noninteractive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Crypto'91,* volume 576 of *LNCS,* pages 433–444. Springer-Verlag, 1991.
[27] T. Satoh and K. Araki. Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. *Commentarii Mathematici Universitatis Sancti Pauli,* (47):81–92, 1998.

[28] I. Semaev. Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic *p*. *Mathematics of Computation,* (67):353–356, 1998.
[29] C. Shannon. Communication theory of secrecy systems. In *Bell System Technical Journal,* volume 28, 1949.
[30] V. Shoup. OAEP reconsidered. In *Crypto'01,* volume 2139 of *LNCS,* pages 239–259, 2001.
[31] V. Shoup. A proposal for an iso standard for public key encryption (version 2.1). Manuscript, 2001.
[32] V. Shoup and R. Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. *Journal of Cryptology,* 15(2):75–96, 2002.
[33] N. Smart. The discrete logarithm problems on elliptic curves of trace one. *Journal of Cryptology,* 12:193–196, 1999.
[34] Y. Watanabe, J. Shikata, and H. Imai. Equivalence between semantic security and indistinguishability against chosen ciphertext attacks. In *PKC 2003,* volume 2567 of *LNCS,* pages 71–84, 2003.
[35] R. Zhang, G. Hanaoka, J. Shikata, and H. Imai. Full version of this paper. Available at: http://eprint.iacr.org/2003/181/.
[36] R. Zhang, G. Hanaoka, J. Shikata, and H. Imai. On the security of multi-layered encryption or CCA-security+CCA-security=CCA-security? In *SCIS'03,* January, 2003.

QuasiModo: Efficient Certificate Validation and Revocation*

Farid F. Elwailly[1], Craig Gentry[2], and Zulfikar Ramzan[2]

[1] Lockheed Martin
farid.f.elwailly@lmco.com
[2] DoCoMo Communications Laboratories USA, Inc.
{cgentry, ramzan}@docomolabs-usa.com

Abstract. We present two new schemes for efficient certificate revocation. Our first scheme is a direct improvement on a well-known tree-based variant of the NOVOMODO system of Micali [11]. Our second scheme is a direct improvement on a tree-based variant of a multi-certificate revocation system by Aiello, Lodha, and Ostrovsky [1]. At the core of our schemes is a novel construct termed a QuasiModo tree, which is like a Merkle tree but contains a length-2 chain at the leaves and also directly utilizes interior nodes. This concept is of independent interest, and we believe such trees will have numerous other applications. The idea, while simple, immediately provides a strict improvement in the relevant time and communication complexities over previously published schemes.

1 Introduction

As we move to an increasingly online world, public-key cryptography will be prevalent. Underlying such use we must have a public-key infrastructure (PKI) that constitutes the policy, procedures, personnel, components, and facilities for binding public keys to identities or authorizations for the purposes of offering desired security services. Typically, a PKI includes a certificate authority (CA) that not only issues binding certificates but also manages them. When issuing a certificate, the CA obviously must check that a user's credentials are accurate, but even a legitimately issued certificate may need to be revoked. Handling revocation is one of the most challenging components of certificate management.

THE CERTIFICATE REVOCATION PROBLEM. While a certificate's validity may be limited by an expiration date, we may sometimes wish to revoke a certificate prior to this time. For example, a key holder may change his affiliation or position, or his private key may have been compromised. This problem is both fundamental and critical – the lack of an efficient solution will hinder the widespread use of PKI. Accordingly, we need an efficient mechanism for revoking a certificate.

One common approach is a *certificate revocation list* (CRL), which is a signed and time-stamped list issued by the CA specifying which certificates have been

* A very preliminary portion of this work was conducted when F. Elwailly and Z. Ramzan were at IP Dynamics, Inc.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 375–388, 2004.

revoked according to some identifier like a serial number. These CRLs must be distributed periodically, even if there are no changes, to prevent illegitimate reuse of stale certificates. CRLs are appealing because of their simplicity. However, their management may be unwieldy with respect to communication, search, and verification costs. An alternative approach, proposed by Kocher [7], is a Certificate Revocation Tree (CRT), which is a Merkle tree that associates each leaf with a revoked certificate. We describe Merkle trees in greater detail below.

Rather than posting full-fledged lists of revoked certificates, the CA may instead answer online queries about specific certificates. This approach is used in OCSP [13], but it has limitations. In particular, the CA must sign each response, which may be computationally infeasible given that it may have to handle numerous requests. A centralized CA creates a major scalability issue because all requests are routed through it. On the other hand, a decentralized CA may lower security since the precious signing key will be replicated on multiple servers, thereby creating multiple attack points.

THE NOVOMODO APPROACH. Micali [10, 11, 12] addressed these problems in an elegant scheme now called NOVOMODO. His scheme works with any standard certificate format such as X.509 and allows a CA to provide validity status of a certificate at any pre-specified time interval such as a day, an hour, etc. NOVOMODO uses a hash chain together with a single digital signature. The advantage is that the cost of the single signature is amortized over many validity proofs. Unfortunately, NOVOMODO requires verification time proportional to the number of periods that have passed between two queries, assuming that the verifier caches information from previous sessions. If, however, the verifier does not cache such information, verification time is proportional to the number of intervals that have passed since the certificate's creation. Even though hash functions require much less time to compute than traditional signatures, hash chain traversal costs may be prohibitively expensive for long chains. For example, benchmark tests conducted using the Crypto++ library showed that SHA-1 is about 5000-6000 times faster than RSA-1024 signing and about 200 times faster than verification. On the other hand, SHA-1 is only 500-600 times faster than ESIGN-1023 signing and about 200 times faster than verification. See [3] for further details. This data suggests that while cryptographic hash functions are faster than signatures, long hash chains are very undesirable, especially for some of the faster signature schemes like ESIGN [16]. Therefore, a natural extension to NOVOMODO that uses Merkle trees was pointed out by Gasko et al. [4] as well as by Naor and Nissim [14]. This variant has the nice property that validity proof size is logarithmic in the total number of update periods.

MULTI-CERTIFICATE REVOCATION. Aiello, Lodha, and Ostrovsky [1] discovered a clever extension to the NOVOMODO approach which allows the CA to provide validity status for a group of certificate owners with a single proof. The idea is to form a cover set $\mathcal{F}$ consisting of various subsets of the set of certificate owners, and construct a Merkle tree or a hash chain for each element of the cover. The cover is constructed so that for any arbitrary subset of revoked users, there are

elements in the cover whose union exactly constitutes the set of non-revoked users. Then, at a given interval, instead of providing validity information for each individual certificate owner, the CA instead finds elements from $\mathcal{F}$ whose union is the set of non-revoked users. The validity proof, which consists of various Merkle tree or hash chain values, is published just for these elements.

OUR CONTRIBUTION: THE QUASIMODO APPROACH. We propose an alternative to Merkle trees which we term QuasiModo trees. QuasiModo trees have two differences. First, their leaves are augmented with hash chains of length 2. Second, rather than starting validity proofs at the leaves, as is typically done in Merkle trees, QuasiModo trees are carefully numbered to allow proofs to start with alternate internal nodes of the tree. The idea, while simple, does not seem to have appeared previously. Yet, the result is a direct improvement in both the overall verification complexity, as well as the communication complexity, over previous tree-based schemes. Moreover, validity proofs are small enough to fit within a single packet – so the extra communication (compared to hash chains) required in practice is negligible. Table 1 summarizes the results of using QuasiModo trees as compared to Merkle trees. QuasiModo trees are of independent interest and may be used to improve other schemes involving Merkle trees. For example, they have recently been applied to the problem of secure billing in networks [5].

ORGANIZATION. The next section states various preliminaries. Section 3 describes the NOVOMODO scheme and section 4 explains the QuasiModo improvement to NOVOMODO. Section 5 discusses the multi-certificate revocation extension to NOVOMODO proposed by [1] and describes how to improve it using QuasiModo trees. Finally, section 6 analyzes the performance of QuasiModo trees as compared to Merkle trees, and provides a security proof for our schemes.

2 Preliminaries

MODEL AND NOTATION. We have a certificate authority $\mathcal{C}$ who issues public-key certificates, and two participants Alice $\mathcal{A}$ and Bob $\mathcal{B}$. $\mathcal{B}$ has a public key that $\mathcal{A}$ wishes to verify. We assume the existence of an open or closed PKI where both $\mathcal{C}$ and $\mathcal{B}$ have public-private key pairs. Let (Sk, Pk) denote a key pair where Sk is the private signing key for computing the signature on a message, and Pk is the public verification key corresponding to Sk. Subscripts denote which keys belong to specific individuals. So, the key pair for $\mathcal{C}$ is $(\mathsf{Pk}_{\mathcal{C}}, \mathsf{Sk}_{\mathcal{C}})$ and the key pair for $\mathcal{B}$ is $(\mathsf{Pk}_{\mathcal{B}}, \mathsf{Sk}_{\mathcal{B}})$. Let $\mathcal{DS} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Vf})$ denote a digital signature scheme that is secure against existential forgery under adaptive chosen message attack [6]. Here KG denotes the key generation algorithm, Sign(Sk, M) denotes the signing algorithm which outputs a signature σ on message M under signing key Sk (the signing algorithm may be randomized), and $\mathsf{Vf}(\mathsf{Pk}, M, \sigma) \in \{0, 1\}$ denotes the verification algorithm which evaluates to 1 if the signature σ on message M is correct with respect to the public key Pk. We remark that KG implicitly takes as input a security parameter specifying the lengths of the keys it should generate.

Let {0,1}* denote the set of all bit strings. Let H denote a *cryptographic compression function* that takes as input a b-bit payload and produces a v-bit output. In our constructions $b = 2v$ which can be achieved by all well-known compression function constructions through padding. H also utilizes a v-bit initialization vector or IV which we assume is fixed and publicly known. For simplicity, we do not view the IV as an actual hash function argument, so we may not always explicitly list it as an input. A practical example of such a cryptographic compression function is SHA-1 [15] whose output and IV size is 20-bytes, and whose payload size is 64-bytes. In any practical instantiation of our schemes we will not need to operate on data larger than the compression function payload size; however there are numerous standard techniques such as iterated hashing or Merkle-trees [9] for doing so. For convenience, we use the term hash function instead of compression function, where it is understood that a hash function can take arbitrary length strings $\{0,1\}^*$ and produce a fixed length output in $\{0,1\}^v$. The symbol $\mathcal{H}$ denotes such a function. We assume cryptographic compression functions and the hash functions built on top of them are *one way and collision resistant* (i.e., finding two distinct inputs $m_1 \neq m_2$ such that $\mathcal{H}(\mathsf{IV}, m_1) = \mathcal{H}(\mathsf{IV}, m_2)$ is difficult).

For a length-preserving function $f : \{0,1\}^n \rightarrow \{0,1\}^n$ and an integer $i \geq 1$, let f^i denote its i-fold composition: $f^i(x) = f(x)$ for $i = 1$ and $f^i(x) = f(f^{i-1}(x))$ for $i > 1$. We say f is a one-way function if, given $f(x)$, where x is randomly chosen, it is hard to find a z such that $f(z) = f(x)$, except with negligible probability. We say f is one way on its iterates if for any i, given $f^i(x)$, it is hard to find a z such that $f(z) = f^i(x)$, except with negligible probability. In practice, one often constructs a candidate function that is one way on its iterates by starting with a hash function $\mathcal{H}$ and padding part of the payload to make it length preserving. Finally, for a real number r, let $\lceil r \rceil$ denote the smallest integer greater than or equal to r. Similarly, $\lfloor r \rfloor$ denotes the largest integer less than or equal to r.

MERKLE TREES. We now describe *Merkle trees* [9]. Suppose that we have m values $x_1, \ldots, x_m$, each of which is in $\{0,1\}^n$. For simplicity, assume that m is a power of 2. Let $\mathcal{H} : \{0,1\}^{2n} \rightarrow \{0,1\}^n$ be a cryptographic hash function. The Merkle tree associated with $x_1, \ldots, x_m$ under hash function $\mathcal{H}$ is a balanced binary tree in which each node is associated with a specific value $\mathsf{Value}(v)$. There are m leaves, and for each leaf ℓ_i, $\mathsf{Value}(\ell_i) = x_i$, $1 \leq i \leq m$. For an interior vertex v, let $\mathsf{C}_0(v)$ and $\mathsf{C}_1(v)$ denote its left and right children. Let $\circ$ denote the concatenation operation. Then, $\mathsf{Value}(v) = \mathcal{H}(\mathsf{IV}, \mathsf{Value}(\mathsf{C}_0(v)) \circ \mathsf{Value}(\mathsf{C}_1(v)))$. Merkle trees may be used to digest data in digital signatures, where the signed digest corresponds to the value associated with the root. If the underlying compression function is collision resistant, then it is hard to find two different messages whose Merkle root value is identical [2, 8]. We will also make use of the notion of the *co-nodes* for a given vertex in a Merkle tree. For a vertex v, $\mathsf{CoNodes}(v)$ is the set of siblings of the vertices on the path from v to the root. More formally, if we let $\mathsf{Sib}(v)$ and $\mathsf{Parent}(v)$ denote v's sibling and parent respectively, then:

$$\mathsf{CoNodes}(v) = \begin{cases} \emptyset & \text{if } v \text{ is the root} \\ \{\mathsf{Sib}(v)\} \bigcup \mathsf{CoNodes}(\mathsf{Parent}(v)) & \text{otherwise.} \end{cases} \quad (1)$$

Finally, for a set of co-nodes, we abuse notation by letting $\mathsf{Value}(\mathsf{CoNodes}(v))$ denote the values associated with the co-nodes of a vertex v. The analogous notion of co-nodes exists for any arbitrary tree. Given the values of a vertex and its co-nodes, we can calculate the root value of the tree. In particular, let the value associated with a vertex be v and let the values of its co-nodes be $v_1, \ldots, v_\ell$. Then, the root value is h_ℓ where $h_1 = \mathcal{H}(v \circ v_1)$ and $h_i = \mathcal{H}([h_{i-1}, v_i])$, $2 \leq i \leq \ell$, where $[h_i, v_i]$ equals $v_i \circ h_i$ if v_i is a left child or $h_i \circ v_i$ if v_i is a right child.

3 NOVOMODO

We now describe the NOVOMODO scheme of Micali [10, 11, 12]. The scheme can be broken up into three phases: a set up phase in which the CA $\mathcal{C}$ issues a certificate to a user Bob $\mathcal{B}$, an update phase in which $\mathcal{C}$ provides an efficient proof of revocation or validity, and a verification phase where a user Alice $\mathcal{A}$ determines the status of $\mathcal{B}$'s certificate.

SET UP. Let f be a function that is one way on its iterates. Let $\mathcal{D}$ denote traditional certificate data (e.g., $\mathcal{B}$'s public key, a serial number, a string that serves as $\mathcal{B}$'s identity, an issue date, and an expiration date). Let p denote the number of periods in the certificate scheme. The CA $\mathcal{C}$ associates with the certificate data $\mathcal{D}$ two numbers y_p and N computed as follows. $\mathcal{C}$ picks values y_0 and N_0 at random from $\{0,1\}^n$. He sets $y_p = f^p(y_0)$ and $N_1 = f(N_0)$. We refer to y_p as the validity target and N_1 as the revocation target for reasons that will shortly become clear. The certificate consists of $(\langle \mathcal{D}, y_p, N_1 \rangle, \mathsf{Sign}(\mathsf{Sk}_\mathcal{C}, \langle \mathcal{D}, y_p, N_1 \rangle))$.

PERIODIC CERTIFICATE UPDATES. The directory is updated each period (for example, if $p = 365$, then the update interval might be daily for certificates that are valid for one year). At period i, if the certificate is valid, then $\mathcal{C}$ sends out $y_{p-i} = f^{p-i}(y_0)$. If the certificate has been revoked, $\mathcal{C}$ sends out N_0.

VERIFYING CERTIFICATE STATUS. Suppose $\mathcal{A}$ wants to verify the status of a certificate at period i. We assume $\mathcal{A}$ performs the standard checks; e.g., the certificate has not expired and $\mathcal{C}$'s signature on the certificate is valid. Now, if $\mathcal{C}$ claims the certificate has been revoked, then $\mathcal{A}$ takes the value N_0 sent by $\mathcal{C}$ and checks if $N_1 = f(N_0)$. Note that she knows N_1 since it is in the certificate. Similarly, if $\mathcal{C}$ claims the certificate has not been revoked, then $\mathcal{A}$ takes the value y_{p-i} sent by $\mathcal{C}$ and checks if $f^i(y_{p-i}) = y_p$. Again, note that $\mathcal{A}$ knows y_p.

NOVOMODO WITH MERKLE TREES. One undesirable property of NOVOMODO is that the verification time is linear in the size of the interval between consecutive validity checks made by $\mathcal{A}$ assuming $\mathcal{A}$ always caches responses from previous queries. For example, if the update period is every 3 hours and certificates are valid for a year, then $\mathcal{A}$ may have to make up to several thousand hash function calls when verifying a certificate. To address this concern, the following

use of Merkle trees in NOVOMODO has been suggested [1, 4, 14]. The CA $\mathcal{C}$ creates a Merkle tree with $2p$ leaves $\ell_1, \ldots, \ell_{2p}$, each of which is assigned a secret pseudorandom value, and signs the root.[3] The leaves are numbered left to right from 1 to $2p$, and at time period i, if the certificate is valid, $\mathcal{C}$ sends out $\mathsf{Value}(\ell_{2i})$ and $\mathsf{Value}(\mathsf{CoNodes}(\ell_{2i}))$.

4 QuasiModo Trees for Single Certificate Revocation

Having described NOVOMODO, we describe our QuasiModo approach. At a high level, QuasiModo replaces the NOVOMODO Merkle trees with QuasiModo trees. These trees yield a performance improvement over using Merkle trees.

QUASIMODO TREES. QuasiModo trees bear some similarity to the Merkle trees used in NOVOMODO, except that we first append length-2 hash chains to the bottom of the tree, and we next carefully number every other interior vertex so they can be efficiently used directly in validation proofs. The power of using such trees is that a subset of the *internal* nodes can be directly utilized in the certificate revocation scheme and we do not always have to use the leaves as is done in the normal Merkle case. The upshot is a sizeable improvement in both the verification complexity and communication complexity.

We start with m randomly chosen values, $x_1, \ldots, x_m$; note that these values can be pseudorandomly generated from a single sufficiently large random seed. For simplicity, suppose that $m = 2^k$ for some integer $k > 0$. We set up a tree as follows. The bottom layer has m vertices which are *only-children* (i.e., they have no siblings). Next, we construct a balanced binary tree of depth $k+1$ which resides on top of the bottom-level m vertices. We assign values to each of the vertices as follows. The bottom-level m vertices take on the values $x_1, \ldots, x_m$ respectively. For the layer that is directly on top of the bottom layer, we assign the n-bit value $f(x_i)$ to the i^{th} such vertex, where $f : \{0,1\}^n \rightarrow \{0,1\}^n$ is a one-way function. That is, if ℓ'_i is such a vertex, then $\mathsf{Value}(\ell'_i) = f(x_i)$, for $1 \leq i \leq m$. For any interior node v that is above the bottom two layers $\mathsf{Value}(v) = \mathcal{H}(\mathsf{IV}, \mathsf{Value}(\mathsf{C}_0(v)) \circ \mathsf{Value}(\mathsf{C}_1(v)))$. In practice, we would typically construct f by appropriately padding $\mathcal{H}$; so, from now on, we only refer to $\mathcal{H}$.

Another way to precisely characterize the same tree is as follows. There are $3m-1$ vertices. These are respectively: $\ell_1, \ldots, \ell_m$, $\ell'_1, \ldots, \ell'_m$, and $v_1, \ldots, v_{m-1}$. The values are assigned as follows. $\mathsf{Value}(\ell_i) = x_i$ and $\mathsf{Value}(\ell'_i) = f(x_i)$, for $1 \leq i \leq m$. Next, let $\lambda(i) = 2(i - m/2) + 1$ and let $\rho(i) = 2(i - m/2) + 2$. For $i \in \{m/2, m/2+1, \ldots, m-1\}$, we have $\mathsf{Value}(v_i) = \mathcal{H}(\mathsf{Value}(\ell'_{\lambda(i)}) \circ \mathsf{Value}(\ell'_{\rho(i)}))$. Finally, for $i \in \{1, \ldots, m/2-1\}$, we have $\mathsf{Value}(v_i) = \mathcal{H}(\mathsf{Value}(v_{2i}) \circ \mathsf{Value}(v_{2i+1}))$. This constitutes the assignment of values to the vertices. Now, we describe the directed edges. There is a directed edge from ℓ_i to ℓ'_i for $1 \leq i \leq m$. For $i \in \{m/2, m/2+1, \ldots, m-1\}$, we have a directed edge from $\ell'_{\lambda(i)}$ to v_i and a directed edge from $\ell'_{\rho(i)}$ to v_i. Finally, for $i \in \{1, \ldots, m/2-1\}$, we have a directed edge

[3] Though it does not seem to have been observed previously in [1, 4, 14], the values ℓ_{2i}, $1 \leq i \leq p$, can be made public without compromising security of the scheme.

from v_{2i} to v_i, and a directed edge from v_{2i+1} to v_i. At a high level, we put a directed edge from a vertex u to a vertex w if $\mathsf{Value}(u)$ was explicitly used to calculate $\mathsf{Value}(w)$.

Next, we apply the following two-coloring to the nodes in the tree. If a vertex is a left child or has no siblings (as in the case of the ℓ_i vertices), we color it grey. All other vertices, including the root, are colored white. Finally, the grey nodes are numbered breadth first (but where the edge directions are ignored). That is, we start at the top of the tree, and work our way down to each consecutive level, numbering each grey node sequentially from left to right. At first this idea of numbering the grey vertices may seem somewhat unnatural, but it turns out to be convenient since the i^{th} grey vertex value is involved in the validation proof at period i. We refer to the i^{th} grey vertex by $\mathsf{gv}(i)$. Figure 1 illustrates a QuasiModo tree that can accommodate a revocation scheme with 7 periods and a Merkle tree that accommodates 8 periods.

In general, a QuasiModo tree accommodating $p = 2^k - 1$ periods requires $\frac{3p+1}{2}$ vertices. A Merkle tree accommodating $p = 2^k$ periods requires $4p - 1$ vertices. A QuasiModo tree is thus approximately $\frac{8}{3} - \frac{14}{9p+3}$ times smaller than the corresponding Merkle tree. Note that we may naturally extend the notion of a QuasiModo tree to an ℓ-chained QuasiModo tree in which each internal vertex is replaced with a hash chain of length ℓ. This extension provides a middle ground between the tradeoffs achieved from QuasiModo trees and regular hash chains.

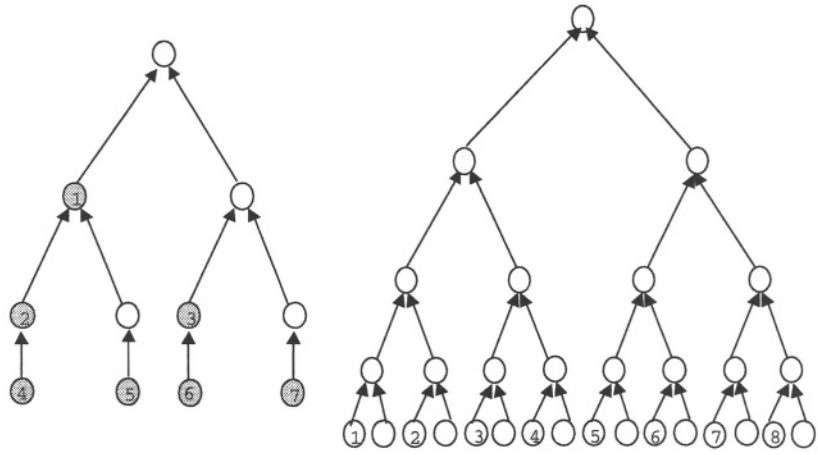


Fig. 1. On the left we have an 11-vertex QuasiModo tree, which can be used for 7 periods; the value of each interior node is the hash of the concatenation of the values of its children. Every grey vertex is numbered sequentially top-down left-to-right. On the right, we have a 31-vertex Merkle tree, which can be used for 8 periods. By using interior nodes and a hash chain at the end, we can get a more compact tree – resulting in shorter proofs, shorter verification time and lower communication complexity.

SET UP. As in NOVOMODO, let $\mathcal{D}$ denote traditional certificate data. The CA $\mathcal{C}$ associates with the certificate data $\mathcal{D}$ two numbers y_r and N_1 computed as follows. $\mathcal{C}$ constructs a QuasiModo tree and sets y_r to be value assigned to the root of that tree. He sets $N_1 = f(N_0)$ like he did for NOVOMODO. The certificate consists of $(\langle \mathcal{D}, y_r, N_1 \rangle, \mathsf{Sign}(\mathsf{Sk}_{\mathcal{C}}, \langle \mathcal{D}, y_r, N_1 \rangle))$.

PERIODIC CERTIFICATE UPDATES. The directory is updated each period. At period i, if the certificate is valid, $\mathcal{C}$ sends out $\langle \mathsf{Value}(\mathsf{gv}(i)), \mathsf{Value}(\mathsf{CoNodes}(\mathsf{gv}(i))) \rangle$. If the certificate has been revoked, he sends out N_0. Note that if $\mathcal{A}$ received co-node values from previous validity checks, it is not necessary for $\mathcal{C}$ to send every value in $\mathsf{Value}(\mathsf{CoNodes}(\mathsf{gv}(i)))$.

VERIFYING CERTIFICATE STATUS. Suppose that $\mathcal{A}$ wants to verify the status of a certificate at period i. We assume she first performs all the standard checks; e.g., the certificate has not expired and $\mathcal{C}$'s signature is correct. Now, if $\mathcal{C}$ claims the certificate has been revoked, then $\mathcal{A}$ takes the value N_0 sent by $\mathcal{C}$ and checks if indeed $N_1 = f(N_0)$. If $\mathcal{C}$ claims the certificate has not been revoked, then $\mathcal{A}$ takes the values $\mathsf{Value}(\mathsf{gv}(i))$ and $\mathsf{Value}(\mathsf{CoNodes}(\mathsf{gv}(i)))$ uses them to compute the QuasiModo tree root. Note that this step requires at most $\lfloor \log_2 i \rfloor + 1$ hash computations for QuasiModo trees as opposed to $\lceil \log_2 p \rceil + 1$ for Merkle trees. If the computed root matches the value y_r, then the certificate is valid. Alternatively, if $\mathcal{A}$ has already verified a certificate for a previous period j (and has stored the proof), and some of the vertex values associated with period i are in a subtree rooted at a vertex associated with the certificate for period j, then $\mathcal{A}$ only needs to use the co-nodes to compute up to that subtree root.

5 QuasiModo Trees for Multi-certificate Revocation

We now propose the use of QuasiModo trees to improve a scheme of Aiello, Lodha, and Ostrovsky (ALO) [1]. We first describe the generalized scheme, and then give examples of how to instantiate it. To describe the scheme, we must consider the notion of a *complement cover family*. Let U denote the universe; in our setting, it will be the set of all certificate holders (regardless of whether the certificate has been prematurely revoked). Let $R \subseteq U$; in our setting, R will denote the set of certificate holders whose certificates have been revoked prior to expiration. Let $\bar{R} = U - R$. That is, $\bar{R}$ will be the set of certificate holders whose certificates are currently not revoked. Now, let $\mathcal{S}$ be a set whose elements are subsets of U. We say that $\mathcal{S}$ is a complement cover of R if $\bigcup_{W \in \mathcal{S}} = \bar{R}$. We can extend this notion to the universe as follows. Let $\mathcal{F}$ be a set whose elements are subsets of U. We say that $\mathcal{F}$ is a complement cover family of U if and only if, for every subset R of U, $\mathcal{F}$ contains a complement cover of R. That is, for every subset R of U, there is a subset $\mathcal{S}$ of $\mathcal{F}$ such that $\mathcal{S}$ is a complement cover of R. The set of all singletons is a simple example of a complement cover family. That is, $\mathcal{F} = \{\{u_1\}, \ldots, \{u_N\}\}$ where $U = \{u_1, \ldots, u_N\}$. Indeed, it is very easy to see that the singleton cover must be contained in any complement cover family for

the universe U. At another extreme, the power set, or set of all subsets of a set, is also trivially seen to be a complement cover family.

At a high level in the ALO [1] scheme, the CA first constructs a complement cover family for the universe of certificate holders. Next, he assigns a Merkle tree to each element of the complement cover family. For a given certificate owner $\mathcal{B}$, let $\mathcal{F}(\mathcal{B})$ denote the set of elements of $\mathcal{F}$ to which the user belongs. The validation targets the CA incorporates, in its user certificate, are the roots of the Merkle trees corresponding to the elements of $\mathcal{F}(\mathcal{B})$. Now, to provide a validation proof at period i for a group of users, the CA first determines the set of revoked users R. Then, he computes the complement cover of R contained in $\mathcal{F}$ – call it $\mathcal{S}$. Note that such a complement cover $\mathcal{S}$ exists since $\mathcal{F}$ is a complement cover family for the universe U. The CA produces the i^{th} leaf and its co-nodes in the associated Merkle tree for each element of $\mathcal{S}$. To check the validity of $\mathcal{B}$'s certificate in period i, a verifier $\mathcal{A}$ checks that the CA has revealed the i^{th} leaf for at least one element of $\mathcal{S}$ in $\mathcal{F}(\mathcal{B})$. We can replace these Merkle trees with QuasiModo trees, and we now describe how to do so.

SET UP. Let U denote the universe of all certificate holders. Then the CA $\mathcal{C}$ constructs a complement cover family $\mathcal{F}$. Let p denote the number of periods. For each element of $\mathcal{F}$, the CA $\mathcal{C}$ constructs an independent QuasiModo tree that allows for p periods. We let $\mathcal{D}$ denote traditional certificate data. The CA $\mathcal{C}$ associates with the certificate data $\mathcal{D}$ a set of validation targets and a single revocation target as follows. $\mathcal{C}$ picks a value N_0 at random from $\{0,1\}^n$. He sets $N_1 = f(N_0)$ – where N_1 represents the revocation target, $\mathcal{C}$ constructs a set of validity targets for the certificate owner $\mathcal{B}$ as follows. He computes $\mathcal{F}(\mathcal{B})$, which is the set consisting of elements of $\mathcal{F}$ for which $\mathcal{B}$ is a member. Suppose that there are κ elements of $\mathcal{F}(\mathcal{B})$ – call them $\mathcal{F}_1, \ldots, \mathcal{F}_\kappa$. Let $r_1, \ldots, r_\kappa$ denote the values of the roots of the QuasiModo trees associated with $\mathcal{F}_1, \ldots, \mathcal{F}_\kappa$. The certificate consists of $(\langle \mathcal{D}, r_1, \ldots, r_\kappa, N_1 \rangle, \mathsf{Sign}(\mathsf{Sk}_\mathcal{C}, \langle \mathcal{D}, r_1, \ldots, r_\kappa, N_1 \rangle))$. We remark that for specific complement cover constructions, one can reduce the number of root values r_i that are included in the augmented certificate data.

PERIODIC CERTIFICATE UPDATES. The directory is updated each period. At period i, if a given certificate is revoked, then $\mathcal{C}$ sends out the pre-image of the revocation target (i.e., the value N_0 value associated with each certificate); if the certificate is valid, then $\mathcal{C}$ does the following. It first determines the set R of revoked holders. It computes the element $\mathcal{S} \in \mathcal{F}$ such that $\mathcal{S}$ is a complement cover for R. For each element of $\mathcal{S}$, $\mathcal{C}$ sends out the value $\mathsf{Value}(\mathsf{gv}(i))$ associated with the tree corresponding to that element, together with $\mathsf{Value}(\mathsf{CoNodes}(\mathsf{gv}(i)))$.

VERIFYING CERTIFICATE STATUS. Suppose that $\mathcal{A}$ wants to check the status of $\mathcal{B}$'s certificate at period i. She first checks the expiration date and that the signature by the CA $\mathcal{C}$ is valid. If $\mathcal{C}$ claims the certificate has been revoked, then $\mathcal{A}$ takes the value N_0 sent by $\mathcal{C}$ and checks if indeed $N_1 = f(N_0)$. If $\mathcal{C}$ claims the certificate has not been revoked, then $\mathcal{A}$ takes the values $\mathsf{Value}(\mathsf{gv}(i))$ and $\mathsf{Value}(\mathsf{CoNodes}(\mathsf{gv}(i)))$ associated with the element of the complement cover that is in $\mathcal{F}(\mathcal{B})$. $\mathcal{A}$ computes the QuasiModo tree root value. If the computed

root value matches one contained in the certificate, then the certificate is valid. Alternatively, if $\mathcal{A}$ has already verified a certificate for a previous period j (and has stored the relevant verification information), and a vertex associated with the proof in period i is in a subtree rooted at a vertex associated with the certificate for period j, then $\mathcal{A}$ only needs to use the co-nodes to compute up to that subtree root.

BINARY TREE HIERARCHY. For completeness, we review a specific complement cover family construction known as the binary tree hierarchy. Assume, for simplicity, that the number of certificate holders is 2^k for some integer $k \geq 0$. We create a binary tree with 2^k leaves and assign to every vertex a subset of the universe of certificate holders. At each leaf, we assign the singleton set corresponding to a single certificate holder. At each internal node, we assign the subset corresponding to the union of the subsets of the nodes of its children. The complement cover family $\mathcal{F}$ consists of the sets assigned to all the vertices. It is clear that $\mathcal{F}$ forms a complement cover family; the following steps yield a minimal-size complement cover of any subset $R \subseteq U$:

1. "Mark" every leaf vertex corresponding to an element of $\bar{R}$;
2. "Mark" every interior vertex on the path from the marked leaf to the root;
3. Determine the non-marked vertices whose parents are marked;
4. Consider the subsets associated with these vertices.

6 Performance and Security Analysis

Our QuasiModo single-certificate and multi-certificate revocation systems are quite efficient in terms of both computation and communication. We compare the performance to their Merkle tree analogues. Our analysis applies to both single-certificate revocation as in NOVOMODO and multi-certificate revocation as in ALO [1]. Table 1 summarizes the results.

COMPLEXITY WITHOUT CACHING. Suppose we have p periods where $p = 2^k - 1$ for some integer $k > 0$. To refresh a certificate at period p_t, $\mathcal{C}$ sends $\mathsf{Value}(\mathsf{gv}(p_t))$ and $\mathsf{Value}(\mathsf{CoNodes}(\mathsf{gv}(p_t)))$. The number of co-nodes to be sent is equal to the depth of this vertex which is $\lfloor \log_2(p_t) \rfloor + 1$. Therefore, the total proof size is $\lfloor \log_2(p_t) \rfloor + 2$ since we need to send the value at vertex p_t itself as part of the proof. To verify, the receiver computes at most $\log_2(p_t) + 1$ hashes, assuming he has not cached any previous values; if he has saved some information from a previous period, then the number of hashes is smaller. In particular, if the verifier caches the value of a vertex at level L of the QuasiModo tree on the path from grey vertex p_t to the root, then he need only compute $\lfloor \log p_t \rfloor - L$ hashes.

For the tree-based version of NovoModo suggested by [4, 14], there are $2p$ leaves, and hence a binary tree of depth $\log_2 p + 1$. However, since this scheme only uses the leaves, the proof size at period p_t is *always* $\lceil \log_2 p \rceil + 2$ and the number of hashes to verify the proof is always $\lceil \log_2 p \rceil + 1$. However, since $p_t \leq p$, we have that $\lfloor \log_2 p_t \rfloor \leq \lceil \log_2 p \rceil$. So, the QuasiModo scheme provides a strict

improvement. Not only are fewer hash function computations required, but also fewer cache look-ups are required to retrieve proof vertex values.

COMPLEXITY WITH CACHING. We compare the bandwidth consumption of QuasiModo tree schemes with Merkle tree schemes assuming that the verifier checks the certificate status at each update period and caches all received results.[4] For $p = 2^k - 1$ periods the corresponding QuasiModo tree has $\frac{3p+1}{2}$ vertices; so the total number of proof node values transmitted is $\frac{3p-1}{2}$ since the root is not counted. For p transactions, the amortized proof size is $\frac{3}{2} - \frac{1}{2p}$ hash values per transaction, and assuming caching $\mathcal{C}$ always sends exactly 2 values for non-leaf vertices and 1 value for leaf vertices. For a Merkle-tree with $p = 2^k$ periods, there are $4p - 1$ vertices ($2p$ leaves and $2p - 1$ internal nodes). Again, ignoring the root value, the total number of proof node values transmitted is $4p - 2$. Thus, the amortized proof size of p transactions is $4 - \frac{2}{p}$. Therefore, the improvement factor is $\frac{8}{3} - \frac{4}{9p-3}$ which approaches $2\frac{2}{3}$ as p gets large. In practice, however, the effects may be more pronounced since the proof sizes in the Merkle setting will vary with each iteration – going up to $\lceil \log_2 p \rceil + 1$ hash values – whereas for QuasiModo trees the size will always be one or two hash values. This variance exhibited by Merkle trees may create performance issues.

We now compare the time complexity of verifying QuasiModo proofs versus Merkle-tree proofs. For p periods, the amortized proof size in a QuasiModo tree is $\frac{3}{2} - \frac{1}{2p}$, and we only require p total calls to a cryptographic compression function for verification at each step assuming that these values fit in the compression function payload, which is the case for practical examples such as SHA-1 [15]. For a Merkle tree the total number of compression-function calls during proof verification is equal to the number of internal (non-leaf) vertices since each internal vertex results from a single compression function call applied to the concatenation of the values associated with its children. Therefore, the number of total compression function calls is $2p - 1$. Consequently, the improvement factor from using QuasiModo trees is $2 - \frac{1}{p}$ which approaches 2 as p gets large.

A potential drawback of the QuasiModo approach is that achieving constant-time verification requires the verifier to cache many of the values it receives. In the worst case, for a QuasiModo tree with p periods, the verifier may have to cache up to $\frac{p+1}{2}$ vertex values (corresponding to the values of the vertices one level from the bottom). This might not be a problem for reasonable parameter values. For example, suppose that a given verifier deals with 100 certificates concurrently, each of which permits 1023 periods (approximately a six-month certificate with update periods every four hours). Then, in the worse case, he needs to keep track of $(100 \cdot \frac{1023+1}{2})$ hash values, which requires under a megabyte of storage assuming we use the SHA-1 hash function with a full 20-byte tag.

[4] In practice there are likely to be many gaps in certificate status checks, but we examine this always-check always-cache case since it lends itself to a cleaner analysis. This portion of the analysis does not apply to our multi-certificate revocation scheme because there may always be gaps. Note, however, that our tree-based constructions are especially advantageous when there are gaps between checks.

HASH CHAINS VERSUS HASH TREES. In a chain-based approach the computation cost may be high since it is linear in the gap size between two verification steps. Trees reduce this to a logarithmic cost. Of course, we make the very reasonable assumption that roughly $\mathcal{O}(\log p)$ processor cache look-ups require less time than $\mathcal{O}(p)$ cryptographic hash function computations. Alternatively, QuasiModo proofs may potentially be short enough to be loaded directly into data registers when reading the incoming proof packet from the CA $\mathcal{C}$. However, one ostensible reason to prefer hash chains is that the proof size is smaller – involving the transmission of just a single hash function value. While the communication requirements of chains, in theory, are smaller, this may not translate into an actual performance improvement in practice since transmission time is typically proportional to the number of packets sent (assuming that they are reasonably sized) rather than the number of bits sent. The average TCP packet, for example, holds a payload on the order of 536 bytes (after removing 20-bytes each for the TCP and IP packet headers) and TCP packet sizes up to approximately 1500 bytes (the maximum ethernet packet size) are reasonable – especially if we perform path maximum transmission unit detection to prevent fragmentation. With packet sizes that are much larger than 20 bytes, we may find room for a few extra hash values without requiring the transmission of any extra packets. In particular we can fit 26 hash values (resp. 70+ hash values) in an *average sized* (resp. *larger sized*) TCP packet with room to spare. These values would permit over 16 *million* (resp. 256 *quintillion* $= 256 \times 10^{18}$) intervals – far more than we may ever require in any practical application. So, in all practical instances, QuasiModo proofs, like NovoModo proofs, would fit into a single packet. Yet, QuasiModo proofs take far less time to verify.

Metric	QuasiModo trees	Merkle trees
Tree Size	$\frac{3p+1}{2}$	$4p-1$
Proof Size (NC)	$\lfloor \log_2(p-r) \rfloor + 2$	$\lceil \log_2 p \rceil + 2$
Verification Time (NC)	$\lfloor \log_2(p-r) \rfloor + 1$	$\lceil \log_2 p \rceil + 1$
Amortized Proof Size (C)	$\frac{3}{2} - \frac{1}{2p}$	$4 - \frac{2}{p}$
Amortized Verification Time (C)	1	$2 - \frac{1}{p}$
Max. Proof Size (C)	2	$\lceil \log_2 p \rceil + 2$
Max. Proof Verification Time (C)	1	$\lceil \log_2 p \rceil + 1$
Min. Proof Size (C)	1	2
Min. Proof Verification Time (C)	1	1

Table 1. Comparing QuasiModo trees to Merkle trees for p periods. Here r denotes the number of periods remaining. Sizes are measured with respect to hash function output size (e.g., 20-bytes). Running times are measured in terms of the number of hash computations. Here *(C)* denotes that the verifier performs validation checks at each interval and caches all values it receives from the CA. We use *(NC)* when the verifier does not cache at all, but does check at each interval.

SECURITY ANALYSIS. Since a QuasiModo tree is essentially a type of hash tree, it is very straightforward to see the security of our scheme. For completeness, however, we sketch the proof of the following security theorem.

Theorem 1. *Assuming that $\mathcal{H}$ is a one-way collision-resistant hash function and that $\mathcal{DS}$ is a secure signature scheme, neither a proof of revocation nor a proof of validity can be forged.*

Proof. (Sketch) We first consider the slightly more involved case of the validity proof. First observe that assuming the security of $\mathcal{DS}$, no adversary can forge the certificate, except with negligible probability. Therefore, an adversary must use an existing certificate and come up with proof of validity that hashes to at least one validity target. Suppose that t update periods have already passed, and an adversary is trying to forge a validity proof for update period $t + \Delta$. Denote the adversary's spurious validity proof by $\mathsf{Value}'(\mathsf{gv}(t+\Delta)), \mathsf{Value}'(\mathsf{CoNodes}(\mathsf{gv}(t+\Delta)))$, where $\mathsf{Value}'(\mathsf{gv}(t+\Delta))$, and $\mathsf{Value}'(\mathsf{CoNodes}(\mathsf{gv}(t+\Delta)))$ denote spurious values for the co-nodes in the CA's QuasiModo tree. Let r denote the root of the tree, which is already known to a verifier since it is part of the certificate. For a verifier to accept the proof, the spurious values must hash to r. For notational simplicity, let $v = \mathsf{Value}'(\mathsf{gv}(t+\Delta))$ and let $r'_1, \ldots r'_\ell$ denote the values of the co-nodes ordered along the siblings of the vertices on the path from the vertex to the root. First note that if ℓ is greater than the depth of the original tree, then the expiration period would be reached (it would also imply that the adversary inverted $\mathcal{H}$ at a random point, which we assume to be infeasible, except with negligible probability). So, let us suppose ℓ is bounded by the depth of the original tree. Now, let $r_1, \ldots, r_\ell$ denote the *actual values* corresponding to what the CA generated in the actual QuasiModo tree. If for all $i \in \{1, \ldots, \ell\}$, it holds that $r_i = r'_i$, then it follows that the adversary correctly computed a pre-image of $\mathcal{H}$ since the CA never revealed all the r_i. This event only happens with negligible probability since $\mathcal{H}$ is a one-way collision-resistant cryptographic hash function.

So, suppose that the r_i and r'_i are not all equal; we show how to construct a hash function collision. Because the r'_i verifiably hash to the root, it follows that the root value can be calculated as h'_ℓ where $h'_1 = \mathcal{H}(\mathsf{Value}'(\mathsf{gv}(t+\Delta))) \circ r'_1)$, and $h'_i = \mathcal{H}([h'_{i-1}, r'_i])$ for $i \in \{1, \ldots, \ell\}$. Likewise, the same root value can be calculated as h_ℓ where $h_1 = \mathcal{H}(\mathsf{Value}(\mathsf{gv}(t+\Delta))) \circ r_1)$ and $h_i = \mathcal{H}([h_{i-1}, r_i])$. Because both calculations yield the same committed root value, it follows that $h_\ell = \mathsf{Value}(r) = h'_\ell$. Now since the r_i and r'_i are distinct, but $h_\ell = h'_\ell$, there must be some index $j \in \{1, \ldots, \ell\}$ for which $h'_j = h_j$, but $(h_{j-1}, r_j) \neq (h'_{j-1}, r'_j)$. In that case, $h'_j = \mathcal{H}([h'_{j-1}, r'_j]) = \mathcal{H}([h_{j-1}, r_j]) = h_j$, which is a collision since the inputs to $\mathcal{H}$ are distinct. We have therefore violated the collision-resistance property of $\mathcal{H}$, which can only happen with negligible probability.

We now consider the revocation target. A forgery yields a pre-image of the revocation target. Since the CA constructed the revocation target by applying $\mathcal{H}$ to a random value, that means the adversary can invert $\mathcal{H}$ at a random point, which happens with negligible probability by the one-wayness of $\mathcal{H}$.

References

[1] W. Aiello, S. Lodha, and R. Ostrovsky. Fast Digital Identity Revocation. In *Proc. of CRYPTO '98.*
[2] I. Damgård. A Design Principle for Hash Functions. In *Proc. of CRYPTO '89.*
[3] W. Dei. Crypto++ library v5.1.
[4] I. Gassko, P. S. Gemmell, and P. MacKenzie. Efficient and Fresh Certification. In *Proc. of PKC 2000.*
[5] C. Gentry and Z. Ramzan. Microcredits for Verifiable Foreign Service Provider Metering. In *Proc. of Financial Cryptography 2004 (to Appear).*
[6] *S.* Goldwasser, S. Micali, and R. L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SIAM Journal on Computing,* 17(2):281–308, 1988.
[7] P. Kocher. On Certificate Revocation and Validation. In *Proc. of Financial Cryptography '98.*
[8] R. Merkle. One-way Hash Functions and DES. In *Proc. of CRYPTO '89.*
[9] R. Merkle. Protocols for Public-Key Cryptography. In *Proc. of IEEE Symposium on Security and Privacy '80.*
[10] S. Micali. Efficient Certificate Revocation. In *Proc. of RSA Data Security Conference '97.*
[11] S. Micali. NOVOMODO: Scalable Certificate Validation and Simplified PKI Management. In *Proc. of PKI Research Workshop '02.*
[12] S. Micali. Efficient Certificate Revocation. LCS/TM 542b, Massachusetts Institute of Technology, 1996.
[13] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. In *Internet RFC 2560,* June.
[14] M. Naor and K. Nissim. Certificate Revocation and Certificate Update. In *Proc. of USENIX Security '98.*
[15] National Institute of Standards. FIPS 180-1: Secure Hash Standard. 1995.
[16] T. Okamoto, E. Fujisaki, and H. Morita. TSH-ESIGN: Efficient Digital Signature Scheme Using Trisection Size Hash. *Contribution to IEEE P1363 '98.*

A Acknowledgements

We thank Alejandro Hevia, Ravi Jain and Toshiro Kawahara for helpful discussions and feedback on earlier manuscript drafts.

A Distributed Online Certificate Status Protocol with a Single Public Key

Satoshi Koga[1] and Kouichi Sakurai[2]

[1] Graduate School of Information Science and Electrical Engineering, Kyushu University
6-10-1 Hakozaki, Higashi-ku, Fukuoka, 812-8581, Japan
satoshi@itslab.csce.kyushu-u.ac.jp
[2] Faculty of Information Science and Electrical Engineering, Kyushu University
6-10-1 Hakozaki, Higashi-ku, Fukuoka City, 812-8581, Japan
sakurai@csce.kyushu-u.ac.jp

Abstract. The *Public Key Infrastructure* (PKI) technology is very important to support secure global electronic commerce and digital communications on networks. The *Online Certificate Status Protocol* (OCSP) is the standard protocol for retrieving certificate revocation information in PKI. To minimize the damages caused by OCSP responder's private key exposure, a distributed OCSP composed of multiple responders is needed. This paper presents a new distributed OCSP with a single public key by using *key-insulated signature scheme* [6]. In proposed distributed OCSP, each responder has the different private key, but corresponding public key remains fixed, so the client simply obtains and stores one certificate and can verify any responses by using a single public key.

Keywords: Public Key Infrastructure, Certificate Revocation, Online Certificate Status Protocol, Distributed OCSP, Key-Insulated Signature Scheme

1 Introduction

1.1 Background and Motivation

Recently, the Internet has been spread all over the world and it has used to be an infrastructure of electronic commerce. However a lot of threats exist on networks, for example wiretapping, alteration of data, and impersonation. It is important to support secure digital transactions and communications throughout existing networks. Confidentiality, integrity, authentication, and non-repudiation are all security requirements to prevent these threats. These requirements can be supported by a variety of different key management architectures. One of these architectures is a *Public Key Infrastructure* (PKI). A PKI is the basis of security infrastructure whose services are implemented and provided using public key techniques. Most of the protocols for secure e-mail, web service, virtual private networks, and authentication systems make use of PKIs.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 389–401, 2004.

In a PKI, a trusted third party called *Certification Authority* (CA) issues a certificate digitally signed by using its private signing key. A certificate is used to bind an entity's identity information with the corresponding public key. Nevertheless, certificates are revoked in case of breaking that binding before its expiration date. If user's private key is compromised or the user's personal information is changed, the user makes a request to the CA for revoking own certificate. The CA is the ultimate owner of certificate status and has the responsibility of publishing to the users that the certificate has been invalid. Thus, users do not simply check the expiration data on the certificate, but also check whether the certificate has been revoked or not. The validation of certificates status information is the current issues in PKI.

A certificate revocation can be implemented in several ways. The most well-known method is to periodically publish a *Certificate Revocation List* (CRL) [9,7]. A CRL is a digitally signed list of revoked certificates and usually issued by a CA for the certificates it manages. In case of validating user's certificate, the verifier obtains the CRL stored in repository and should verify its validity and CA's digital signature. And the verifier should confirm whether user's certificate is contained in the CRL or not. The main advantage of the CRL systems is its simplicity, however, there are several problems pointed out [1,23]. Especially, the main disadvantage of the CRL systems is its high communication costs between the user and the repository stored on CRLs. It is said that a certificate revocation rate around 10 percent per year is reasonable [20]. Therefore, the size of CRL will be quite long if the CA has many clients. That is, the validation performance is likely to be slow, since the verifier has to download the CRLs from each CA (or CA's repository) in a certification chain and verify each CRLs. This fact is critical problem if the client is the mobile terminal with restricted processing capacities, memory limitations, and network bandwidth. In order to reduce the size of CRLs, several modifications have been suggested. *Delta CRL* [9] is small CRL that includes information about the certificates that have been revoked since the issuance of a complete revocation list called *Base CRL.* And *CRL Distribution Points* was defined in [9]. CRL Distribution Points allow revocation information within a single domain to be divided into the multiple CRLs.

In order to reduce the communication costs, there are some alternative methods to CRL-based systems. The *Certificate Revocation Tree* (CRT) was proposed by Kocher [12]. CRTs are based on Merkle Hash Trees [14], in which the tree itself represents all certificate revocation information. Naor and Nissim proposed the *Authenticated Directory* [19], which improves the reduction in communication cost by balancing the hash tree. They introduced using a 2-3 tree, in which every node has two or three children. In [10,11], the binary hash tree is extended to k-ary hash tree in which any node has at most k children. Micali proposed the revocation system using hash chains [15,16], taking into account both user's and CA's efficiency.

If the client needs very timely information of certificate status, an online certificate status service is required. The standard online revocation system is the *Online Certificate Status Protocol* (OCSP) defined in [18]. The OCSP pro-

vide the up-to-date response to certificate status queries and enable to reduce the communication costs in comparison with the CRL, because the client only request to return the status of certificate instead of obtaining the CRLs. The certificate status is returned by a trusted entity referred to as an OCSP responder. The response indicates the status of the certificate returning the value *good, revoked,* and *unknown.* Additionally, the OCSP responder signs each response it produces by using its private key. The CRL is published the data on all of revoked certificates, for example those data are issuer's name and its serial number. Since any client can obtain the CRL, this fact will be leading the privacy concerns. On the other hand, the OCSP responder simply returns the status of requested certificate and does not expose information about all revoked certificates. In mobile environment, the method of using the OCSP appears to be a good choice, because the client can retrieve timely certificate's status with a moderate resource usage. As the online protocol that are more extensive than OCSP, several mechanisms that build and validate certification path instead of end users are suggested [13,22]. This paper only focuses on the certificate status checking mechanism.

In OCSP, the communication costs will be reduced, however, it substantially increases computation costs since a digital signature is a computationally complex operation. Consequently, it becomes highly vulnerable to *denial-of-service* (DoS) attacks, if the responder is centralized [16]. Another threat is the leakage of responder's private key. In case of compromising responder's private key, the attacker can generate the forged response that the revoked certificate is valid. As well as CA's private key, responder's private key exposures affect the serious impact for the client. So the countermeasure against those threats is important to provide the online certificate status service securely.

1.2 Related Work

To reduce the risk of DoS attacks, OCSP responders may pre-produce signed responses specifying the status of certificates at a specified time [18]. However, the use of pre-produced responses allows replay attacks in which an old response is replayed prior to its expiration date but after the certificate has been revoked. To avoid the replay attacks, the responder needs to generate pre-produced responses within a short period of time. But this consumes a lot of processing and this fact causes DoS attacks. In [17], the modification over OCSP using hash chain is suggested to reduce the computational load of the OCSP responder.

As well as CA's private keys, responder's private key must be stored very carefully. There are some approaches to protect the private key from attackers. A Hardware Security Module (HSM) may reduce the risk of key compromise. An attacker requires penetration or theft of the HSM to retrieve responder's private key. To evaluate the security of HSM objectively, the security requirements for cryptographic modules are specified in [21]. Another approach is to manage a share of responder's private key on different servers by using a threshold cryptography [4]. A proactive signature [3] is the enhanced threshold solution

by periodic refreshment of shares. These approaches can be effective, but key exposures caused by operation mistakes appear to be unavoidable.

1.3 Our Contributions

As mentioned above, it is difficult to avoid all of threats completely. If the OCSP responder is centralized, the entire system is affected by DoS attacks and compromising responder's private key. That is, the entire service is not available in those cases. Therefore, minimizing damages caused by responder's private key exposure and DoS attacks is extremely important to employ the OCSP system.

A distributed OCSP (D-OCSP) model composed of multiple responders mitigates these damages. In case that each responder has the same private key, compromising any responder compromises the entire system [16]. On the other hand, if each responder has the different private key, compromising a responder cannot affect the others. Hence, this paper examines the D-OCSP that each responder has the different private key. In the general D-OCSP model, the CA issues each responder's certificate. However, the client's load becomes heavy in this model. Every time clients receive the response from the responder, they need to obtain responder's certificate. Moreover, when clients utilize the different responder, they need to get its certificate.

This paper presents a new D-OCSP with a single public key by using *key-insulated signature scheme* (KIS) based on the difficulty of the discrete logarithm problem [6]. The KIS is one of the methods for mitigating the damage caused by private key exposures. Using a KIS-enabled responder, compromise of responder's private key only affects at short time period. We focus on the property that all signatures can be verified by using fixed public key in KIS. This paper takes a different approach from KIS-enabled responder. The multiple private keys are generated using key update algorithm in KIS and assigned to the separate responders, respectively. Thus each responder has the different private key, but corresponding public key remains fixed and the client can verify any responses by using a single public key. Once the client obtained responder's certificate, she simply stores it and can utilize during its validity. Thereby, communication costs are more efficient in comparison with the general model. In our model, the client needs to check the validation of responder's private key as well as the traditional certificate. Our proposed D-OCSP applies the Micali's revocation system [16] and the client checks the validation of responder's private key efficiently than using like the CRL.

The rest of this paper is organized as follows. In Section 2, we explain the traditional D-OCSP, in which the CA issues responder's certificate with a short life-time, and discuss the problems of traditional D-OCSP. In Section 3, we describe the proposed D-OCSP, including the validation of responder's private key and decentralizing processes of responders. Section 4 details the viewpoints of security and performance of our D-OCSP. Concluding remarks are made in Section 5.

2 Distributed OCSP

2.1 Model

In a distributed OCSP (D-OCSP), there are three entities, as shown in Fig1.

1. **Certification Authority (CA)**
   A Certification Authority (CA) is a trusted third party that has the responsibility of publishing the certificate revocation information. Compromise of CA's private key will have disastrous for the entire system, so the CA is isolated form the Internet in order to avoid unauthorized accesses.
2. **Responders**
   A responder is a trusted entity that sends the certificate status information to clients.
3. **Clients**
   Clients trust the CA's public key certificate and request the certificate status information to responders.

In this section, we explain the general D-OCSP model using responder's certificates. If each responder has the same private key, the compromising of any responder compromises the entire system [16]. Thus, we examine the D-OCSP that each responder has the different key-pair (PK_i, SK_i). The CA issues each responder's public key certificate digitally signed by its own private key. As well as the traditional public key certificate, the client needs to check revocation information of responder's certificates. There are some ways of checking those information [7]. The simplest method is to use the CRL issued by CA. Another way is to use the responder's certificate with a short lifetime. Using short-lived certificates, clients don't have to check the validation of responder's certificate. In this way, D-OCSP composed of n-responders is shown in Figure 1.

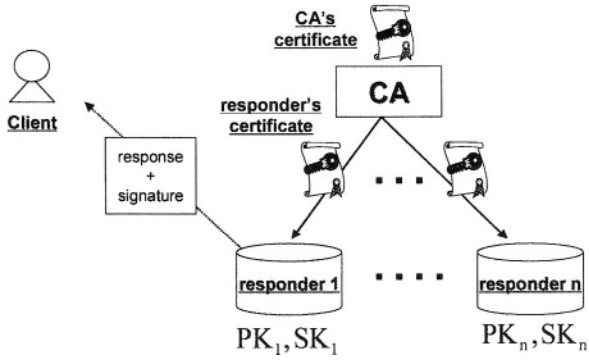


Fig. 1. A Distributed OCSP Model

2.2 Verification Processes

In case that the client receives the response from responder i, she should verify that response as follows.

1. The client obtains the certificate of responder's by online or offline.
2. The client verifies the digital signature contained responder's certificate by using CA's public key.
3. The client verifies the digital signature contained the response by using responder's public key.

(Problems)

1. *Client Efficiency*
   Every time the client receives the response, she should obtain the responder's certificates, since responder's certificate should be updated frequently. Therefore, the communication costs between the client and responders are not efficient. Even if the CA issues the long-lived responder's certificate, the client needs to download the different responder's certificate in case of receiving responses sent by the different responder. So the memory space of the client will be increasing.
2. *CA Efficiency*
   The CA needs to issue responder's certificates frequently. Thereby, the CA needs to produce a digital signature and the computational costs are increasing.

3 Proposed Method

This paper proposes a new D-OCSP with a single public key. In detail, we use a *key-insulated signature scheme* (KIS) [6] and responder's private keys are generated at once. And the client can verify any responses by using a single public key. Before suggesting the decentralization method, we explain the KIS in detail.

A lot of the digital signature schemes have been proposed, but they provide no security guarantees in case of private key exposures. To minimize the damage caused by the leakage of private keys, the notion of key-insulated security was introduced in [5] and a KIS is formalized in [6]. As in a standard signature scheme, the user begins by registering a single public key that remains fixed for the lifetime of the protocol, while the corresponding private key can be changed frequently. A master secret key is stored on physically secure device. The lifetime of the protocol is divided into distinct periods 1, ..., N. At the beginning of period i, the user interacts with the secure device to derive a temporary private key SK_i. Even if SK_i is exposed, an attacker cannot forge signatures for any other time periods. Moreover, in a strong (t, N)-key-insulated scheme, an attacker cannot forge signature for any of remaining $N-t$ periods even if she obtains the private keys for up to t periods. Using a KIS-enabled responder, responses are signed

using responder's private key SK_i at time period i. In that case, the attacker can forge the responses only during period i, if SK_i is compromised. That is, compromise of responder's private key only affects those responses at the point of compromise.

We focus on the property that all signatures can be verified by using fixed public key. This paper takes a different approach from KIS-enabled responder. Suppose the total number of responders is n in our D-OCSP, n private keys are generated using key update algorithm in KIS and assigned to the separate n responders, respectively. Thus each responder has the different private key, but corresponding public key remains fixed. Thus, verifiers can verify responses sent by any responders using a single public key. The details of these processes are described in Section 3.2.

Besides a key-insulated model, alternate approaches have been proposed. The first such example is a *forward-secure signature scheme* (FSS) [2]. This scheme can prevent compromise of private keys at the previous time periods, even if an attacker exposes the current private key. However, once the attacker exposes the current private key, she can easily derive the private keys of the future periods. Like a proposed model, a D-OCSP model using FSS has the advantage that the client can verify any responses using a single public key, but this model cannot minimize the impact caused by compromising responder's private keys. Another approach is a *intrusion-resilient signature scheme* (IRS) proposed in [8]. This scheme adds key-insulation to a proactive refresh capability which may be performed more frequently than key updates. IRS can be tolerant multiple corruptions of both the user and the physically secure device. Any signatures are secure if both of devices are compromised, as long as the compromises are not simultaneous. Compared to FSS and KIS, IRS has a high security. In our method, however, a master secret key stored on physically secure device is only used during private key generations. Thus master key is deleted after that generations are finished. Taking into account the computation costs, this paper examines the decentralizing method of CA using KIS.

3.1 Validation of Responder's Private Key

In this section, we examine the validation method of responder's private key. The client needs to check that a responder's certificate has not been revoked. There are some ways of checking those information [7]. The simplest method is that the client checks the offline verification using like a CRL issued by the CA. While, the CA may choose not to specify any method of revocation checking for responder's certificate. In that case, responder's certificate with a very short lifetime should be issued. In the traditional D-OCSP mentioned in Section 2, the client doesn't have to check the validation of responder's certificate. However, the D-OCSP using responder's certificate with a short lifetime has disadvantages. The first problem is that communication costs are inefficient, since the client should obtain the responder's certificate in case of receiving the response. Moreover, CA's computational costs become high because of updating responder's certificate frequently.

In our model, each responder has the different private key, but corresponding public key remains fixed. As well as the traditional model, if this private key is compromised, this private key needs to be revoked and the CA publishes all user that this private key is invalid. We utilize Micali's revocation system proposed in [16]. Micali's revocation system uses the hash-chain and is efficient as to computational costs. Our model uses a one-way hash function H satisfying the following properties, as well as Micali's system.

(One-way hash function)

1. H is at least 10,000 faster to compute than a digital signature scheme.
2. H produces 20-byte outputs.
3. H is hard to invert, given Y, finding X such that $H(X) = Y$ is practically impossible.

(Issuance of Responder's Certificate)

1. Let T be the total number of time-periods. For example, T is 365 if each responder's certificate expires 365 days after issuance. The CA produces T hash value using H as follows.

$$X_T \xrightarrow{h} X_{T-1} \xrightarrow{h} X_{T-2} \xrightarrow{h} \cdots \xrightarrow{h} X_1$$

Let n be the total number of responders. The CA repeatedly produces n hash-chain as different input value $X_{T,i}$. $X_{t,i}$ denotes the hash value at time priod t for validation of responder j. These hash values are stored on the CA.

$$\begin{array}{c} X_{T,1} \xrightarrow{h} X_{T-1,1} \xrightarrow{h} X_{T-2,1} \xrightarrow{h} \cdots \xrightarrow{h} X_{1,1} \\ X_{T,2} \xrightarrow{h} X_{T-1,2} \xrightarrow{h} X_{T-2,2} \xrightarrow{h} \cdots \xrightarrow{h} X_{1,2} \\ \vdots \\ X_{T,n} \xrightarrow{h} X_{T-1,n} \xrightarrow{h} X_{T-2,n} \xrightarrow{h} \cdots \xrightarrow{h} X_{1,n} \end{array}$$

2. The CA issues responder's certificate C_{res} by using own private key. SN is the serial number of certificate and V represents the validity period. I and S denote issuer and subject of certificate, respectively.

$$C_{res} = Sig_{SK_{CA}}(PK_{res}, SN, I, S, V, X_{1,1}, ..., X_{1,n})$$

(Validation of Responder's Private Key)

1. The CA delivers the hash value $X_{t,i}$ to responder i, if responder i's private key SK_i is valid at t period.
2. When responder i returns the response to the client at period t, she also delivers the hash value $X_{t,i}$ to the client.

$$\langle i, t, X_{t,i}, R, Sig_{SK_i}(R) \rangle$$

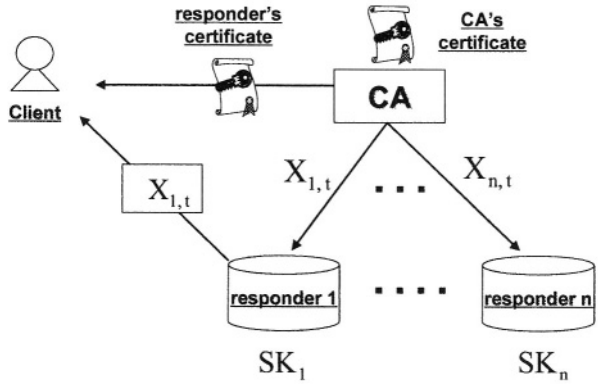


Fig. 2. Proposed D-OCSP

3. When the client receives the response by responder i, she verifies the digital signature by using responder's public key PK_{res}. Then the client can check the validation of responder's private key using hash value $X_{t,i}$ and $X_{1,i}$ contained responder's certificate. In detail, the client checks the following equation. If that equation is satisfied, the client can certify that SK_i is valid.

$$X_{1,i} = H^{t-1}(X_{t,i})$$

In this way, the client can verify the validation of the responder's private key. The responder's certificate is not revoked during its validity unless all of responder's private keys are revoked.

3.2 Decentralizing Method of Responder

We describe the decentralizing process using KIS based on the difficulty of discrete logarithm problem [6]. Let $R_1, ..., R_n$ be responders in our model. Using the following processes, a D-OCSP composed of $R_1, ..., R_n$ is constructed (Figure2).

Step1: generation of responder's private keys

1. *Key pair generation*
   Let p and q be prime numbers such that $p = 2q + 1$ and let g, h be a element of order q in the group $\mathbb{Z}_p$. A responder's public key PK_{res} is generated by choosing $x, y \in_R \mathbb{Z}_q$ and setting $v = g^x h^y$. SK^* denotes the master key to be used generating of responder's private keys. During the generation processes, SK^* is stored on the CA.

$$x_0^*, y_0^*, ..., x_t^*, y_t^* \leftarrow \mathbb{Z}_q$$
$$v_i^* = g^{x_i^*} h^{y_i^*}$$
$$SK^* = (x_1^*, y_1^*, ..., x_t^*, y_t^*)$$
$$PK_{res} = (g, h, v_0^*, ..., v_t^*)$$

2. *Responder's private key generation*
   A partial key SK'_i is generated as follows. SK'_i is used to derive R_i's private key.

$$x'_i = \Sigma_{k=1}^{t} x_k^*(i^k - (i-1)^k)$$
$$y'_i = \Sigma_{k=1}^{t} y_k^*(i^k - (i-1)^k)$$
$$SK'_i = (x'_i, y'_i)$$

   By using partial keys derived above, n private keys are generated. Once all private keys is derived, SK'_i and SK^* are deleted.

$$x_i = x_{i-1} + x'_i$$
$$y_i = y_{i-1} + y'_i$$
$$SK_i = (x_i, y_i)$$

   The CA delivers the private key SK_i to R_i securely. Thus, each responder has the different private key.
3. *Issuance of responder's certificate*
   As mentioned section 3.1, the CA issues the responder's certificate C_{res} as follows.

$$C_{res} = Sig_{SK_{CA}}(PK_{res}, SN, I, S, V, X_0^1, ..., X_0^n)$$

Step2: Signature and verification algorithm

1. *Signature algorithm*
   When R_i returns the response M to the client, she generates a digital signature $\langle i, (w, a, b)\rangle$ by using SK_i asfollows.

$$r_1, r_2 \leftarrow \mathbb{Z}_q$$
$$w = g^{r_1} h^{r_2}$$
$$\tau = H(i, M, w)$$
$$a = r_1 - \tau x_i$$
$$b = r_2 - \tau y_i$$

2. *Verification algorithm*
   The client can verify R_i's signature by using PK_{res} as follows.

$$v_i = \prod_{k=0}^{t} (v_i^*)^{i^k}$$
$$\tau = H(i, M, w)$$
$$w = g^a h^b v_i^{\tau}$$

4 Evaluations

1. *Security*
   Suppose that an attacker steals R_i's private key SK_i and hash value X_t^i at time period t. In this case, she cannot derive any other responder's private keys unless she obtain SK^* (SK^* is deleted after generating responder's private keys). And if an attacker can get the hash value X_t^i, she cannot

derive the hash value $X_{t+1,i}(H(X_{t+1,i}) = X_{t,i})$ because H is a one-way function. Therefore, an attacker cannot cheat that SK_i is valid after period $t+1$ and our model can minimize the damage caused by responder's private key exposures.

2. *Communication costs*
   In the traditional D-OCSP, the client should get the responder's certificate in case of receiving the response from the responder. On the other hand, our model can mitigate the communication costs, because the responder's certificate is only one. The client stores responder's certificate and need not to obtain it by online or offline during the certificate's validity.
3. *Validation of responders*
   In our model, validation of responder's private key is performed by using hash-chain, without using CRL. As mentioned above, hash computation is much faster than digital signature computations. In case of checking the status of responder's certificate, the client just computes t-times hash computations.
4. *CA Efficiency*
   The CA should store the hash value securely. The total size of those value amounts $20nT$-bytes. However, the CA does not have to store all hash values and only store $X_{1,i}$ ($20n$-bytes), since hash computations is very fast. At period t, $X_{t,i}$ is derived by $T-t$ times hash computations. In the traditional D-OCSP, the CA should issue the responder's certificate with a short lifetime. In our model, the CA can issue long-lived responder's certificate, because the client can validate the responder's private key. Thus our model is more efficient than traditional model.

Table 1 shows the comparison between our model using KIS and the traditional D-OCSP using DSA. As the comparison items, we consider the total size of responses, the verification cost of the client (validation of responder's certificate and verification cost), and signing cost of responder. Let $size(C_{res})$ be the size of responder's certificate. (For example, the size of traditional public key certificate is about 800-byte.) Let q, t be the parameter of digital signature scheme. We consider that $q = 160$ and $t \approx n$. The computational cost is represented as the number of multiplications over $\mathbb{Z}_p$ or $\mathbb{Z}_q$. Let $\mathrm{EX}_{\mathbb{Z}_p}$ be the number of multiplications required to compute an exponentiation. In our method, computational cost is less efficient than traditional D-OCSP, but the client may verify any responses by using a single public key. Additionally, the client just obtains the responder's certificate at a time.

5 Conclusions

In order to minimize the damage caused by responder's private key exposure and DoS attacks, the distributed OCSP model composed of the multiple responders is required in real world. This paper suggests the new distributed OCSP model using key-insulated signature scheme. In our model, the client needs to check

Table 1. Comparison between traditional D-OCSP and our D-OCSP

	Traditional (DSA)	Proposal (KIS)
size of resposes	$2q + size(C_{res})$	$3q + 160$
validation of certificate	nothing	t-hash computaions
signing verification cost	3+2EX$_{Z_p}\|q\|$	t + 2+3EX$_{Z_p}\|q\|$
signing cost	2+EX$_{Z_p}\|q\|$	2+2EX$_{Z_p}\|q\|$

the validation of responder's private key as well as the traditional certificate. Our proposed D-OCSP applies Micali's revocation system and the client check the validation of responder's private key efficiently than using like the CRL. In mobile environment, the client has the restricted processing capacity as well as the bandwidth. So our future work is to reduce the computation costs of clients.

References

1. A. Arnes, M. Just, S. J. Knapskog, S. Lloyd, and H. Meijer, *Selecting Revocation Solutions for PKI,* 5th Nordic Workshop on Secure IT Systems (NORDSEC 2000), 2000.
   http://www.pvv.ntnu.no/ andrearn/certrev/
2. M. Bellare, and S. K. Miner, *A Forward-Secure Digital Signature Scheme,* Advances in Cryptology - CRYPTO '99, LNCS 1666, pp.431-448, Springer-Verlag, 1999.
3. R. Canetti, R. Gennaro, A. Herzberg, and D. Naor, *Proactive Security: Long-term protection against break-ins,* RSA CryptoBytes, Volume 3, No. 1, 1997.
   http://www.rsasecurity.com/rsalabs/cryptobytes/
4. Y. Desmedt and Y. Frankel, *Threshold Cryptosystems,* Advances in Cryptology - CRYPTO '89, LNCS 435, pp.307-315, Springer-Verlag, 1990.
5. Y. Dodis, J. Katz, S. Xu and M. Yung, *Key-Insulated Public Key Cryptosystems,* Advances in Cryptology - EUROCRYPT 2002, LNCS 2332, pp.65-82, Springer-Verlag, 2002.
6. Y. Dodis, J. Katz, S. Xu, and M. Yung, *Strong Key-Insulated Signature Schemes,* Public Key Cryptography - PKC 2003, LNCS 2567, pp.130-144, Springer-Verlag, 2003.
7. R. Housley, W. Polk, W. Ford, and D. Solo, *Certificate and Certificate Revocation List (CRL) Profile,* IETF RFC3280, 2002.
   http://www.ietf.org/rfc/rfc3280.txt
8. G. Itkis, and L. Reyzin, *SiBIR: Signer-Base Intrusion-Resilient Signatures*CAdvances in Cryptology - CRYPTO 2002, LNCS 2442, pp.499-514, Springer-Verlag, 2002.
9. ITU/ISO Recommendation. X.509 Information Technology Open Sysytems Interconnection - The Directory: Authentication Frameworks, 2000.
10. H. Kikuchi, K. Abe, and S. Nakanishi, *Performance Evaluation of Certificate Revocation Using k-valued Hash Tree,* 2nd International Workshop on Information Security (ISW '99), LNCS 1729, pp.103-117, Springer-Verlag, 1999.
11. H. Kikuchi, K. Abe, and S. Nakanishi, *Certificate Revocation Protocol Using k-Ary Hash Tree,* IEICE TRANS. COMMUN., Vol. E84-B, No.8, 2001.

12. P. C. Kocher, *On Certificate Revocation and Validation,* Financial Cryptography (FC '98), LNCS 1465, pp.172-177, Springer-Verlag, 1998.
13. A. Malpani, R. Housley, and T. Freeman, *Simple Certificate Validation Protocol*CIETF Internet-Draft, 2003.
http://www.ietf.org/internet-drafts/draft-ietf-pkix-scvp-13.txt
14. R. C. Merkle, *A Certified Digital Signature,* Advances in Cryptology - CRYPTO '89, LNCS 435, pp.218-238, Springer-Verlag, 1990.
15. S. Micali, *Efficient Certificate Revocation,* Technical Memo MIT/LCS/TM-542b, Massachusetts Institute of Technology, 1996.
16. S. Micali, *NOVOMODO; Scalable Certificate Validation And Simplified PKI Management,* 1st Annual PKI Research Workshop, pp.15-25, 2002.
http://www.cs.dartmouth.edu/ pki02/
17. J. L. Munoz, J. Forne, O. Esparza, I. Bernable, and M. Soriano, *Using OCSP to Secure Certificate-Using Transactions in M-commerce,* Applied Cryptography and Network Security (ACNS 2003), LNCS 2846, pp.280-292, Springer-Verlag, 2003.
18. M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, *X.509 Internet Publik Key Infrastructure Online Certificate Status Protocol-OCSP,* IETF RFC2560, 1999.
http://www.ietf.org/rfc/rfc2560.txt
19. M. Naor, and K. Nissim, *Certificate Revocation and Certificate Update,* 7th USENIX Security Symposium, pp.217-228, 1998.
http://www.usenix.org/publications/library/proceedings/usenix98/
20. A. Nash, W. Duane, C. Joseph, and D. Brink, *PKI - Implementing and Managing E-Security,* Osborne Media Group, 2001.
21. National Institute of Standards and Technology (NIST), *Security Requirements for Cryptographic Modules,* FIPS 140-2, 2001.
http://csrc.nist.gov/publications/fips/
22. D. Pinkas, and R. Housley, *Delegated Path Validation and Delegated Path Discovery Protocol Requirements,* RFC3379, 2002.
http://www.ietf.org/rfc/rfc3379.txt
23. R. L. Rivest, *Can We Eliminate Certificate Revocation Lists ?,* Financial Cryptography (FC '98), LNCS 1465, pp.178-183, Springer-Verlag, 1998.

A First Approach to Provide Anonymity in Attribute Certificates*

Vicente Benjumea, Javier Lopez, Jose A. Montenegro, and Jose M. Troya

Computer Science Department
University of Malaga, Spain
{benjumea,jlm,monte,troya}@lcc.uma.es

Abstract. This paper focus on two security services for internet applications: authorization and anonymity. Traditional authorization solutions are not very helpful for many of the Internet applications; however, attribute certificates proposed by ITU-T seems to be well suited and provide adequate solution. On the other hand, special attention is paid to the fact that many of the operations and transactions that are part of Internet applications can be easily recorded and collected. Consequently, anonymity has become a desirable feature to be added in many cases. In this work we propose a solution to enhance the X.509 attribute certificate in such a way that it becomes a conditionally anonymous attribute certificate. Moreover, we present a protocol to obtain such certificates in a way that respects users' anonymity by using a fair blind signature scheme. We also show how to use such certificates and describe a few cases where problems could arise, identifying some open problems.

Keywords: Authorization, PMI, anonymity, pseudonym, credential, X.509 attribute certificates

1 Introduction

Identity certificates (or *public-key certificates*) provide the best solution to integrate the authentication service into most of those applications that are developed for the Internet and make use of digital signatures. The use of a wide-range authentication service based on identity certificates is not practical unless it is complemented by an efficient and trustworthy mean to manage and distribute all certificates in the system. This is provided by a *Public-Key Infrastructure* (PKI).

However, new applications, particularly in the area of e-commerce, need an authorization service to describe what the user is granted to. In this case, privileges to perform tasks should be considered. Thus, for instance, when a company needs to establish distinctions among their employees regarding privileges over

* This work has been partially supported by the Spanish Ministry of Science and Technology under the Project TIC2002-04500-C02-02

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 402–415, 2004.

resources, the authorization service becomes important. Different sets of privileges over resources (either hardware or software) will be assigned to different categories of employees. Also, in those distributed applications where company resources must be partially shared through the Internet with other associated companies, providers, or clients, the authorization service becomes an essential part.

Authorization is not a new problem, and different solutions have been used in the past. However, traditional solutions are not very helpful for many of the Internet applications. *Attribute Certificates,* proposed by the ITU-T (International Telecommunications Union) in the X.509 Recommendation [14], provide an appropriate solution. Additionally, the attribute certificates framework defined by ITU provides a foundation upon which a *Privilege Management Infrastructure* (PMI) can be built.

On the other hand, during last years users have paid special attention to the problem caused by the fact that many of the operations and transactions they carry out through the Internet can be easily recorded and collected. Thus, anonymity has become a desirable feature to be added in many cases.

Since early 80's many studies have been oriented towards the protection of users' privacy in electronic transactions [4,5,6,18]. Those studies have originated with new cryptographic primitives and protocols that have been applied to several specific applications oriented to solve some specific problems such as electronic cash [8], electronic voting [1,10,12], and others, and some proposals with a multi-purpose point of view that cope with organizations and credentials [6,7,9,15,16]. However, such a technology have not been transferred to general applications in the real world. To the best of our knowledge, only one system have been designed and implemented with a practical point of view [2,3]. However, even this system does not follow proposed standards such as X.509 attribute certificates.

It is our belief that one of the main steps to transfer such a technology to multi-purpose real world applications is the ability to apply them to open standard systems. Therefore, in this paper we show a first approach to provide anonymity in X.509 attribute certificates, transferring *fair blind signature* schemes to those standard certificates, and defining *Anonymous Attribute Certificates* in which the holder's identity can be conditionally traceable depending on certain conditions.

The structure of the paper is as follows. In section 2 we briefly argue the use of blind signatures as basic construction block for our solution. Section 3 describes the standard X.509 attribute certificates proposed by ITU-T, and how the framework that this type of attributes define is linked to PKIs. Section 4 describes, throughout three subsections the overview of the scheme, the adaptation of attribute certificates to support anonymity, and the protocol for a user to obtain an anonymous attribute certificate. Section 5 concludes the paper, presenting an interesting discussion about results and open issues.

2 Blind Signatures as a Basic Construction Block

It is widely known that *blind signature* protocols [5] provide a mean for a signer to sign a message sent by an entity. The signer is unable to know anything about the message, and can not link the signed message with its originator.

These schemes have been widely studied and applied to solve specific problems where anonymity is fundamental, such as electronic voting systems [1,10,12] and electronic cash [8]. However, these schemes present an open door for fraud, since perfect anonymity offers the best coverage for dishonest behaviour [19]. Therefore, these schemes must be used with the maximum of caution, by subjects under control, and where perfect anonymity is the only solution to the problem.

Other schemes have been developed to avoid that inconvenience. *Fair blind signature* protocols [18,11] try to close the gap between anonymity and fairness. In these schemes, the anonymity can be broken and the signed message can be linked (only under certain conditions) with the person who requested such a blind signature. In these cases, a *Trusted Third Party* (TTP) is needed in order to run the protocol, and the collusion of the TTP with the signer and the signed message is a necessary condition.

3 X.509 Attribute Certificates

One of the main advantages of an attribute certificate is that it can be used for various purposes. It may contain group membership, role, clearance, or any other form of authorization. A very essential feature is that the attribute certificate provides the means to transport authorization information in distributed applications. This is especially relevant because through attribute certificates authorization information becomes “mobile”, which is highly convenient for Internet applications.

The mobility feature of attributes have been used in applications since the publication of the 1997 ITU-T X.509 Recommendation [13]. However, it has been used in a very inefficient way. That recommendation introduced an ill-defined concept of attribute certificate. For this reason, most of actual applications do not use specific attribute certificates to carry authorization information. On the contrary, attributes of entities are carried inside identity certificates. The *subjectDirectoryAttributes* extension field is used for this purpose. This field conveys any desired directory attribute values for the subject of the certificate, and is defined as follows:

```
subjectDirectoryAttributes EXTENSION ::= {
    SYNTAX    AttributesSyntax
    IDENTIFIED BY id-ce-subjectDirectoryAttributes }
AttributesSyntax ::= SEQUENCE SIZE (1..MAX) OF Attribute
```

This solution does not make entity attributes independent from identity, what can cause problems. Firstly, this is not convenient in the frequent situations where the authority issuing the identity certificate is not the authority for the

assignment of privileges. Secondly, even in the situations where the authority is the same one, we must consider that life of identity certificates is relatively long when compared to the frequency of change of user privileges. Therefore, every time privileges change it is necessary to revoke the identity certificate, and it is already known that certificate revocation is a costly process.

Moreover, many applications deal with authorization issues like delegation (conveyance of privilege from one entity that holds a privilege to another entity) or substitution (one user is temporarily substituted by another user, and this one holds the privileges of the first one for a certain period of time). Identity certificates support neither delegation nor substitution.

The most recent ITU-T X.509 Recommendation of year 2000 provides an approach to these problems because it standardizes the concept of attribute certificate, and defines a framework that provides the basis upon which a PMI can be built. Precisely, the foundation of the PMI framework is the PKI framework defined by ITU. In fact, ITU attribute certificates seem to have been mainly proposed to be used in conjunction with identity certificates; that is, PKI and PMI infrastructures are linked by information contained in the identity and attribute certificates (figure 1).

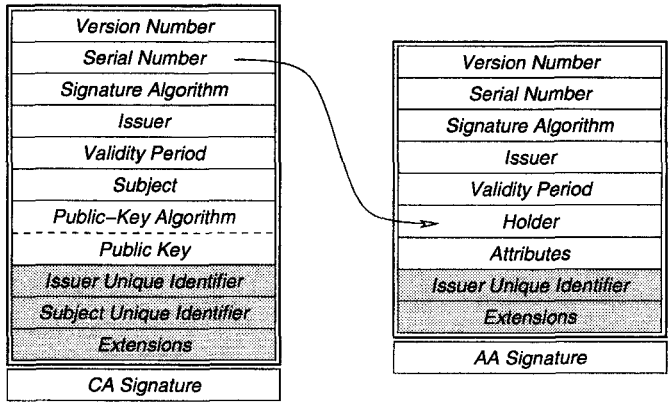


Fig. 1. Relation between identity and attribute certificates

Although linked, both infrastructures can be autonomous, and managed independently, what provides a real advantage. In the most recent recommendation, attribute certificates are conveniently described, including an extensibility mechanism and a set of specific extensions. A new type of authority for the assignment of privileges is also defined, the *Attribute Authority* (AA), while a special type of authority, the *Source of Authority* (SOA), is settled as the root of delegation chains. The recommendation defines a framework that provides a foundation upon which a PMI is built to contain a multiplicity of AAs and final users. Revocation procedures are also considered by defining the concept

of *Attribute Certificate Revocation Lists,* which are handled in the same way as *Certificate Revocation Lists,* published by *Certification Authorities* (CAs) in the PKI case.

As shown in figure 1, the field *holder* in the attribute certificate contains the *serial number* of the identity certificate. As mentioned in [17], it is also possible to bind the attribute certificate to any object by using the hash value of that object. For instance, the hash value of the public key, or the hash value of the identity certificate itself, can be used. All possibilities for the binding can be concluded from the ASN.1 specification of the field *holder,* where other related data structures are also specified:

```
Holder  ::= SEQUENCE {
    baseCertificateID   [0] IssuerSerial     OPTIONAL,
    entityName          [1] GeneralNames     OPTIONAL,
    objectDigestInfo    [2] ObjectDigestInfo OPTIONAL
}

GeneralNames ::= SEQUENCE SIZE (1..MAX) OF GeneralName

GeneralName ::= CHOICE {
    otherName                   [0] INSTANCE OF OTHER-NAME,
    rfc822Name                  [1] IA5String,
    dNSName                     [2] IA5String,
    x400Address                 [3] ORAddress,
    directoryName               [4] Name,
    ediPartyName                [5] EDIPartyName,
    uniformResourceIdentifier   [6] IA5String,
    iPAddress                   [7] OCTET STRING,
    registeredID                [8] OBJECT IDENTIFIER
}

ObjectDigestInfo    ::= SEQUENCE {
    digestedObjectType  ENUMERATED {
        publicKey               (0),
        publicKeyCert           (1),
        otherObjectTypes        (2)
   },
    otherObjectTypeID       OBJECT IDENTIFIER  OPTIONAL,
    digestAlgorithm         AlgorithmIdentifier,
    objectDigest            BIT STRING
}
```

As we will see in next section, the content of this specification is essential for the scheme that we have developed.

4 Introducing Anonymity into Attribute Certificates

4.1 Overview of the Scheme

Our scheme coexists with standards PMI and PKI. While a PKI provides support for users' identities, the AA issues certificates about attributes that the

users hold. Additionally, we suppose that some organizations provide services to users based on their respective attributes. We have introduced in the scheme a TTP which provides (in collusion with the AAs) the ability to disclose anonymous users' identities. Some of the AAs will have the special capacity to issue anonymous attribute certificates. Each of those AAs is in connection with several Attribute sub-Authorities, that will be in charge of verifying that a user fulfills the requirements needed to obtain an "anonymous" certificate containing a specific attribute (figure 2).

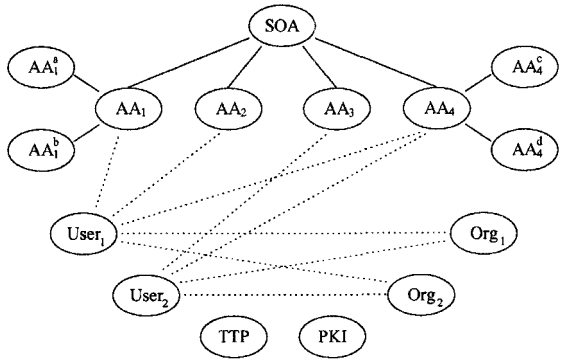


Fig. 2. System Overview

The role that the actors play in our solution can be roughly seen as follows. A user can anonymously acquire as many pseudonyms as he needs from the TTP, where the validity of the pseudonyms are limited in time. Every obtained pseudonym is composed by two related parts: one of them is public and the other one is private. In the following we will refer these parts as public pseudonym and private pseudonym respectively. The TTP keeps such a relationship until the end of the validity period. For each anonymous certificate that the user wants to get, he will collect all proofs needed to apply for a specific attribute (or set of attributes), and will send the proofs, together with his identity and his public pseudonym, to the Attribute sub-Authority in charge of verifying such proofs. If the set of proofs is complete, a special token related to the public pseudonym will be issued (by using a fair blind signature scheme), and a link stating the relationship between the user's identity and his public pseudonym will be stored.

This special token will be modified (again, using a fair blind signature scheme) by the user in order to hide its relationship with the public pseudonym and will reflect, since that moment, the relationship with the private part. This token, now associated with the private pseudonym, will be used by the user to anonymously apply for an anonymous attribute certificate to the AA. Note that if the anonymous user holds that token, then he fullfills the requirements needed to get the certificate containing a (set of) specific attributes.

Once the AA checks that everything is correct, it issues the certificate of the attributes that corresponds with the Attribute sub-Authority that issued such a token. As stated, these certificates are issued anonymously and are related with the user's private pseudonym. Therefore, nobody can link them with the real users' identity unless the TTP and the Attribute sub-Authority collude and some conditions are met. By definition, it is supposed that the TTP will remain trusted and will not reveal the link between both parts of the pseudonym unless a condition expressed in the certificate is fulfilled and such a condition is signed by the user and the AA.

The user will make use of the attribute certificate in order to enforce his privileges. As it is anonymous, it is not linked to any PKI. However it contains a public key and the user who knows the corresponding private key is considered the one who owns such an attribute.

4.2 Adapting Attribute Certificates to Support Anonymity

In section 3 we have mentioned that the field *holder* of the attribute certificate can contain the digest of any object. Thus, we will define an object, called *Pseudonym Structure* (figure 3), to support the conditionally anonymity of the owner and will link such an object with the attribute certificate by using this field. The pseudonym structure fields are the following ones:

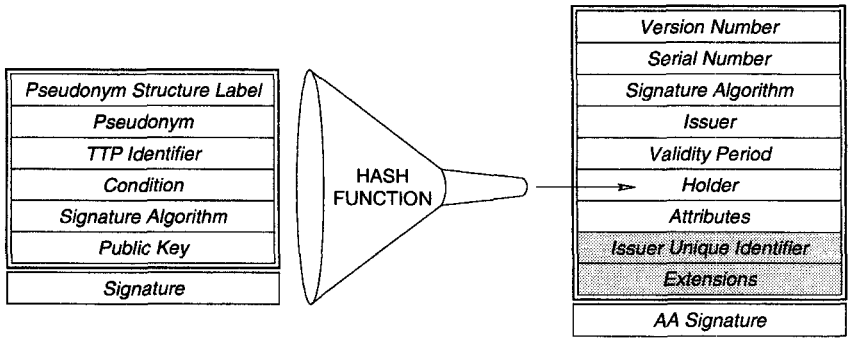


Fig. 3. Relation between pseudonym structure and attribute certificate

- *Pseudonym Structure Label:* A static field that allows us to interpret the object as a proper pseudonym structure.
- *Pseudonym:* The holder's private pseudonym, issued by the TTP specified in the next field.
- *TTP Identifier:* The issuer of the pseudonym, that keeps a record linking the private pseudonym with the public one.
- *Condition:* The condition under which, if fulfilled, both the TTP and the AA will collude and will reveal the user's identity.

- *Signature Algorithm:* Identifies the algorithm for signature and verification of documents using the public key stored in the next field.
- *Public Key:* The key used to authenticate the owner of the attribute certificate in such a way that the anonymous user who holds the corresponding private key will be the attribute owner. For a proper authentication procedure, the anonymous user should sign a challenge with that private key every time that authentication is needed.
- *Signature:* The anonymous user signs the pseudonym structure to prove that it is a valid structure and that he knows the corresponding private key. Moreover, the signature is the proof that the anonymous user accepts the condition stated above with respect to revealing his real identity.

We will define a *conditionally anonymous* X.509 attribute certificate as the attribute certificate itself together with the pseudonym structure, linked by mean of the *holder* field as stated before. The attribute certificate is signed by the attribute authority, what means that the AA agrees on the terms expressed in the linked pseudonym structure. Therefore, the user should know the authorization policy and the conditions under which an attribute certificate request is granted.

It is supposed that the TTP will not reveal pseudonym links unless the condition stated in the certificate is fulfilled. It is also supposed that the user will not transfer his anonymous attribute certificate by revealing the corresponding private key to any other user. This is probably the weakest requirement in our solution and it needs a further study, as discussed later.

4.3 Protocol to Obtain and Use an Attribute Certificate

In this subsection we will explain the protocol to obtain an attribute certificate, and how it can be used. This protocol uses as fundamental construction block the fair blind signature scheme presented in [18] under the name of *fair blind signatures with registration.* Most of the structure of Parts I and II of our protocol correspond with the aforementioned protocol, but the nomenclature has been adapted, and an abstraction of the protocol has been used to masquerade the underlying mathematics. Additionally, in Part II, some steps have been introduced to adapt it to our scheme.

The cryptographic nomenclature used in the protocol is shown in Table 1. Actors involved in the protocol can be seen as follows:

- Actors and terminology
  - U is the user. His certified public key, U_{publ}, is supported by an external PKI.
  - N is a user's anonymous asymmetric key with no PKI support.
  - P is a user's pseudonym. It has a public part P_{publ}, which is associated to the user, and a private part, P_{priv}.
  - The *Trusted Third Party* [*TTP*] provides pseudonyms to users and keeps a link between both parts (public and private) of the pseudonym.
  - The *Attribute Authority* [*AA*] provides attribute certificates, and its certified public key AA_{publ} is supported by an external PKI.

Nomenclature	Meaning
$A : act$	$A's$ action act
$A \rightarrow B : m$	m is sent from A to B
$m = (m_1, m_2)$	m is composed by m_1 and m_2
$c = E_z(m)$	m is encrypted with the symmetric key z
$m = D_z(c)$	c is decrypted with the symmetric key z
A_{publ}, A_{priv}	$A's$ asymmetric public and private keys
$c = E_A(m)$	m is encrypted with $A's$ asymmetric public key
$m = D_A(c)$	c is decrypted with $A's$ asymmetric private key
$h = H(m)$	$m's$ one way hash function
$s_m = \mathbb{S}_A(m)$	$m's$ message signature with $A's$ asymmetric private key $[\mathbb{S}_A(m) \Leftrightarrow E_{A_{priv}}(H(m))]$
$m_s = S_A(m)$	Signed message composed by the message m and its signature with $A's$ asymmetric private key $[S_A(m) \Leftrightarrow (m, \mathbb{S}_A(m))]$
$b = V_A^?(m_s)$	Verify the signed message m_s with $A's$ asymmetric public key $[V_A^?(m_s) \Leftrightarrow \left(H(m') \stackrel{?}{=} D_{A_{publ}}(\mathbb{S}_A(m))\right)] \ / \ [m_s \equiv (m', \mathbb{S}_A(m))]$
$z = NSK()$	Create new symmetric key z
$A = NAK()$	Create new asymmetric key pair for A

Table 1. Cryptographic protocol nomenclature

- The *Attribute sub Authorities* $[AA^i \ / \ \forall\, i \in Attributes]$ verify that a user fullfills the requirements needed to apply for an attribute certificate on $ATTR^i$. Their certified public key $AA^i{}_{publ}$ are supported by an external PKI.
- $ATTR^i$ is the attribute for which the *Attribute subAuthority* $[AA^i \ / \ \forall\, i \in Attributes]$ checks for requirement fulfillment, and for which the *Attribute Authority* $[AA]$ provides attribute certificates.
- $ATTR^i_U$ is the proof that the user U fulfills the requirements to apply for the attribute i.
- SP is a *Service Provider* that offers services to those users that have the attribute certificate $ATTR^i$.
- f_{publ} and f_{priv} are two flags that specify which part of the pseudonym is public and which one is private.
- *val-period* is the period in which the pseudonym remains valid.
- $fblind_X(m)$ represents that the message m is protected to be "fair blind" signed by X.
- $S_X^{P_{publ}}(fblind_X(m))$ is the fair blind signature of X over message m under the public pseudonym P_{publ}, as specified in [18].
- $S_X^{P_{priv}}(m)$ is the fair blind signature of X over message m under the private pseudonym P_{priv}, after transforming the public blind signature to the corresponding private clear form. It is composed by the message and the fair blind signature under P_{priv}.

The whole protocol is divided into the following parts:

Part I. Obtaining a Pseudonym. This part corresponds with the registration phase in the *fair blind signatures with registration* protocol from [18]. It deals

with the user's acquisition of a pseudonym. The user will request a pseudonym from the TTP that is able to produce valid pseudonyms. This TTP must be recognized by the entity that issues the attribute certificates. This TTP will create a new pseudonym, which consists of two parts, the public and the private parts, respectively. Both parts must be created in a related way that makes possible the fair blind signature. The *TTP* will store and keep such a linked pair, so that the relation could be disclosed if some conditions are met. Then, both parts will be signed (with a flag identifying its purpose and its validity period) and sent to the user who requested them. This part of the protocol is achieved in an anonymous way and the *TTP* does not know anything about the user who requests a pseudonym. This part will be run whenever a user needs a new pseudonym.

1. $U : z = NSK()$
2. $U \rightarrow TTP : E_{TTP}\,(z, Pseudonym\_Request)$
3. $TTP : New\_Pseudonym\,(P_{publ}, P_{priv})$
4. $TTP : STORE\,(val\_period, P_{publ} \leftrightarrow P_{priv})$
5. $TTP \rightarrow U : E_z\,(S_{TTP}\,(f_{publ}, val\_period, P_{publ}), S_{TTP}\,(f_{priv}, val\_period, P_{priv}))$

Part II. Obtaining a Fair Blind Signature. This part of the protocol corresponds with the phase of getting a signature in the *fair blind signatures with registration* protocol from [18]. In this phase, the user obtains a message signed by the Attribute subAuthority $[AA^i]$ in charge of verifying fulfillment of the requirements needed to get a certificate over the attribute i. The way in which the fair blind signature operates guarantees that the signer is unable to know what he is signing, and that the signature is done over a public pseudonym related with the user, but such a relationship will be removed by transforming the signature over the private pseudonym.

Therefore, in this phase, the goal of the user is to obtain a proof that reveals that its owner fullfills the requirements needed to get an attribute certificate on a specific attribute. However, nobody must be able to link such a proof with the user.

In our protocol, the proof that a user fulfills a set of requirements consists of a public key signed by the authority in charge of verifying such requirements. The owner of the signed public key remains anonymous; that is, nobody is able to establish a relationship with the user that created it. However, the signature has a link with a private pseudonym, but nobody knows who the owner is. At the moment of issuing the fair blind signature the authority operates over a public pseudonym that is able to relate with the user's identity.

Thus, in the second part of the protocol the user creates a new asymmetric key pair (this key pair will be associated with the attribute certificate). He prepares such a public key to be fair blind-signed by the authority in charge and sends it together with information about the TTP, his public pseudonym and the set of proofs that show that the user fulfills the needed requirements.

The authority checks that the pseudonym is valid and that the TTP is recognized, and then checks if the user fulfills the requirements needed in order to

get a certificate containing the attribute i. These requirements depend on the entity's policy.

If the requirements are met, this information is stored for its later use and the public key will be fair blind signed over the public pseudonym. Once the user gets that signature, he transforms it into a clear signature of the public key over the private pseudonym.

1. $U : N = NAK()$
2. $U \rightarrow AA^i : S_U\left(TTP, S_{TTP}\left(f_{publ}, val\_period, P_{publ}\right), ATTR_U^i, fblind_{AA^i}\left(N_{publ}\right)\right)$
3. AA^i : **IF** $\left(\neg V_{TTP}^{?}\left(S_{TTP}\left(f_{publ}, val\_period, P_{publ}\right)\right)\right.$
   $\left.\vee \neg fulfill\_req\left(U, TTP, P_{publ}, ATTR_U^i\right)\right)$ **THEN Abort**
4. $AA^i : STORE\left(U \leftrightarrow ATTR_U^i \leftrightarrow TTP \leftrightarrow S_{TTP}\left(f_{publ}, val\_period, P_{publ}\right)\right)$
5. $AA^i \rightarrow U : S_{AA^i}^{P_{publ}}\left(fblind_{AA^i}\left(N_{publ}\right)\right)$
6. $U : S_{AA^i}^{P_{priv}}\left(N_{publ}\right)$

Part III. Obtaining a Conditionally Traceable Attribute Certificate. In this part of the protocol, the user will use the anonymous proof obtained in the previous part in order to apply for a standard attribute certificate. Thus, the user creates a structure to hold the information about his pseudonym and signs it to state that such information is correct and that the owner (the one who knows the private key associated with the public key) agrees on the terms expressed in such a structure.

At that moment, the user sends the proof obtained in the previous part, that is, the fair blind signature of the public key linked with the private pseudonym, the proof that the private pseudonym is valid, and the structure previously created.

The AA will verify every signature and will check the terms expressed in such a structure, specially in the condition under which the user's real identity will be revealed. Therefore, provided that the terms are signed by the holder and by the authority, the TTP will reveal the link between the private pseudonym and the public one whenever the attribute certificate is presented to the TTP and *condition* is verified. Additionally, the AA will reveal the link between the public pseudonym and the user's identity.

When everything works correctly, the AA creates an attribute certificate for a validity period stating that the holder of the related structure possesses such a specified attribute, and sends it to the user. The holder of such a structure is the one who knows the private key associated with the public key in it.

1. $U : Pseud\_Inf = S_N\left(Label_{PI}, P_{priv}, TTP, Cond, Sig\_Alg, N_{publ}\right)$
2. $U \rightarrow AA : \left(S_{TTP}\left(f_{priv}, val\_period, P_{priv}\right), S_{AA^i}^{P_{priv}}\left(N_{publ}\right), Pseud\_Inf\right)$
3. AA : **IF** $\left(\neg V_{TTP}^{?}\left(S_{TTP}\left(f_{priv}, val\_period, P_{priv}\right)\right) \vee \neg V_{AA^i}^{?}\left(S_{AA^i}^{P_{priv}}\left(N_{publ}\right)\right)\right.$
   $\left.\vee \neg V_N^{?}\left(Pseud\_Inf\right) \vee \left(\neg Agree\_on\left(Cond\right)\right)\right)$ **THEN Abort**
4. $AA := Attr\_Cert S_{AA}\left(Vers, Serial, Sig\_Alg, AA, Val\_Period, H\left(Pseud\_Inf\right), ATTR^i\right)$
5. $AA \rightarrow U : Attr\_Cert$

Part IV. Using a Conditionally Traceable Attribute Certificate. In this part we show how the attribute certificate obtained in the previous part can be used. A user will send his anonymous attribute certificate plus the pseudonym information associated to any service provider, *SP*. This will verify that such a message is correct and that the certified attribute is enough to access to the service, sending a request to the anonymous user for the signature of a challenge in order to prove ownership. If the challenge is correctly signed then the service is granted to the user.

1. $U \rightarrow SP : (Attr\_Cert, Pseud\_Inf)$
2. $SP : \texttt{IF}\ \left(\neg V_{AA}^{?}\left(Attr\_Cert\right) \vee \left(H\left(Pseud\_Inf\right) \stackrel{?}{\neq} Holder\_Field\left(Attr\_Cert\right) \right) \right.$
   $\left. \vee\ \neg fulfill\_req\left(Service, Attr\_Cert, Pseud\_Inf\right) \right)\ \texttt{THEN Abort}$
3. $SP \rightarrow U : challenge$
4. $U \rightarrow SP : S_N\left(challenge\right)$
5. $SP : \texttt{IF}\ \left(\neg V_N^{?}\left(S_N\left(challenge\right)\right)\right)\ \texttt{THEN Abort}$
6. $SP \rightarrow U : Service\_granted$

If the user misuses his privileges obtained through an anonymous attribute certificate, then the service provider will collect all the proofs of that misuse, and will send them to the AA and the TTP requesting the revocation of the attribute certificate and revealing the user's identity (for an eventual prosecution). In these cases, it could be interesting that the challenge includes a timestamp and the transaction identification besides the random bits, in order to prove misuses where time is important.

5 Discussion and Future Work

New applications, particularly in the area of e-commerce, need an authorization service to describe privileges to perform tasks. Traditional authorization solutions are not very helpful for many of the Internet applications; however, attribute certificates proposed by ITU-T are well suited to solve this problem. On the other hand, during last years, users have paid special attention to the problem caused by the fact that many of the operations and transactions they carry out through the Internet can be easily recorded and collected. Thus, anonymity has become a desirable feature to be added in many cases.

We have presented a first approach to extend X.509 attribute certificates with anonymity capabilities, as well as a protocol to obtain certificates preserving user's anonymity by using a fair blind signature scheme.

The approach could be improved and adapted depending on the different scenarios where to be applied. We explain now how several improvements can be added to our scheme in order to have a better behavior.

In some applications when a user applies for a certificate, the system should provide a receipt of such a request in order to guarantee that the system will process it appropriately. Whenever the system replies to that request, it should

get a receipt in order to prove that its duty was achieved properly. In those systems a fair non-repudiation scheme [20] should be used.

Moreover, in order to improve the user's anonymity, an anonymous communication channel (such as a mixnet [4]) could be used in part I, III and IV of the protocol to masquerade the originator IP address. This scheme should be used in systems where user's anonymity is the most important requirement to the system and user's identity could be guessed using the IP address of the message originator.

In order to avoid the possibility that organizations create anonymous user profiles, a user can run the protocol several times to get the same attribute certificate under a different pseudonym. However, it would be interesting to get a pseudonym with one public part and many private ones, in such a way that it would be only necessary to re-run part III of the protocol in order to get the attribute certificate under different pseudonyms (all related to the same public part).

The solution that we propose in tgis work does not solve all problems that could arise in a multi-purpose anonymous attribute system. We believe that the main drawbacks in our actual solution are:

- The user's identity in part II of the protocols could be linked with the private pseudonym in part III of the protocols if, during the protocol run, such a user is the only one who has an unfinished open request and the AA colludes with the Attribute subAuthority. Thus, the interleaving of user's requests between part II and part III is very important in our protocol.
- Actual version of the protocol does not avoid that the anonymous user U_1 transfers the use of his anonymous attribute certificate to another anonymous user U_2 just by letting U_2 know the associate private key. U_1 and U_2 would share in this way the use and advantages of possessing that attribute, even if U_2 does not posses it. This is, of course, one of the most important areas where we will focus our further research.

References

1. J. Benaloh and D. Tuinstra. Receipt free secret-ballot elections. In *Proc. of 26th Symp. on Theory of Computing (STOC'94),* pages 544–553, New York, 1994.
2. J. Camenisch and E. V. Herreweghen. Design and implementation of the idemix anonymous credential system. In *Proc. of 9th ACM Conference on Computer and Communications Security (CCS),* Washington D.C., Nov. 2002. ACM, Academic Press.
3. J. Camenisch and A. Lysyanskaya. Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation. In *EUROCRYPT 2001,* volume 2045 of *Lecture Notes in Computer Science,* pages 93–118. Springer-Verlag, 2001.
4. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM,* 24(2):84–88, Feb. 1981.

5. D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. Rivest, and A. Sherman, editors, *Advances in Cryptology–Crypto'82,* pages 199–203, Santa Barbara, CA USA, Aug. 1983. Plenum Press.
6. D. Chaum. Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM,* 28(10):1030–1044, Oct. 1985.
7. D. Chaum and J. H. Evertse. A secure and privacy-protecting protocol for transmitting personal information between organizations. In A. M. Odlyzko, editor, *Advances in Cryptology - Crypto '86,* volume 263 of *Lecture Notes in Computer Science,* pages 118–170, Berlin, 1986. Springer-Verlag.
8. D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash (extended abstract). In *Advances in Cryptology–Crypto'88,* pages 319–327, 1989.
9. L. Chen. Access with pseudonyms. In E. Dawson and J. Golic, editors, *Cryptography: Policy and Algorithms,* volume 1029 of *Lecture Notes in Computer Science,* pages 232–243. Springer-Verlag, 1995.
10. L. Cranor and R. Cytron. Sensus: A security-conscious electronic polling system for the internet. In *Proceedings of the Hawaii International Conference on System Sciences,* Wailea, Hawaii, 1997.
11. C.-I. Fan and C.-L. Lei. A user efficient fair blind signature scheme for untraceable electronic cash. *Information Science and Engineering,* 18(1):47–58, Jan. 2002.
12. A. Fujioka, T. Okamoto, and K. Ohta. A practical secret voting scheme for large scale elections. In J. Seberry and Y. Zheng, editors, *ASIACRYPT'92,* volume 718 of *Lecture Notes in Computer Science,* pages 244–251. Springer-Verlag, 1992.
13. ITU-T Recommendation X.509. Information technology - open systems interconnection - the directory: Authentication framework. June 1997.
14. ITU-T Recommendation X.509. Information technology - open systems interconnection - the directory: Public-key and attribute certificate frameworks. Mar. 2000.
15. A. Lysyanskaya. Pseudonym systems. Master's thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, June 1999.
16. A. Lysyanskaya. *Signature Schemes and Applications to Cryptographic Protocol Design.* PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Sept. 2002.
17. S. Farrel and R. Housley. An Internet attribute certificates profile for authorization. Request for Comments 3281. Nework Working Group. Internet Engineering Task Force. April 2002.
18. M. A. Stadler, J. M. Piveteau, and J. L. Camenisch. Fair blind signatures. In L. C. Guillou and J. J. Quisquater, editors, *Advances in Cryptology–EUROCRYPT'95,* volume 921 of *Lecture Notes in Computer Science,* pages 209–219. Springer-Verlag, 1995.
19. S. von Solms and D. Naccache. On blind signatures and perfect crimes. *Computers & Security,* 11:581–583, 1992.
20. J. Zhou. Achieving fair nonrepudiation in electronic transactions. *Journal of Organizational Computing and Electronic Commerce,* 11(4):253–267, 2001.

A Nommiform Algorithm for the Hidden Number Problem in Subgroups

Igor E. Shparlinski[1] and Arne Winterhof[2]

[1] Department of Computing, Macquarie University
Sydney, NSW 2109, Australia
igor@ics.mq.edu.au
[2] Johann Radon Institute for Computational and Applied Mathematics
Austrian Academy of Sciences
Altenbergerstraße 69, 4040 Linz, Austria
arne.winterhof@oeaw.ac.at

Abstract. Boneh and Venkatesan have proposed a polynomial time algorithm in a non-uniform model for recovering a "hidden" element $\alpha \in \mathbb{F}_p$, where p is prime, from very short strings of the most significant bits of the residue of αt modulo p for several randomly chosen $t \in \mathbb{F}_p$. Here we modify the scheme and amplify the uniformity of distribution of the 'multipliers' t and thus extend this result to subgroups of $\mathbb{F}_p^*$, which are more relevant to practical usage. As in the work of Boneh and Venkatesan, our result can be applied to the bit security of Diffie–Hellman related encryption schemes starting with subgroups of very small size, including all cryptographically interesting subgroups.

Keywords: Hidden number problem, Diffie-Hellman key exchange, Lattice reduction, Exponential sums, Waring problem in finite fields, Nonuniform algorithm

1 Introduction

For a prime p, denote by $\mathbb{F}_p$ the field of p elements and always assume that it is represented by the set $\{0, 1, \ldots, p-1\}$.Accordingly, sometimes, where obvious, we treat elements of $\mathbb{F}_p$ as integer numbers in the above range.

For a real $\eta > 0$ and $t \in \mathbb{F}_p$ we denote by $\mathrm{MSB}_\eta(t)$ any integer which satisfies the inequality

$$|t - \mathrm{MSB}_\eta(t)| < p2^{-\eta-1}. \tag{1}$$

Roughly speaking, $\mathrm{MSB}_\eta(t)$ is an integer having about η most significant bits as t. However, this definition is more flexible and better suited to our purposes. In particular we remark that η in the inequality (1) need not be an integer.

Given a subgroup $\mathcal{G} \subseteq \mathbb{F}_p^*$ we consider the following *hidden number problem* over $\mathcal{G}$:

Recover a number $\alpha \in \mathbb{F}_p$ such that for k elements $t_1, \ldots, t_d \in \mathcal{G}$, chosen independently and uniformly at random from $\mathcal{G}$, we are given k pairs

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 416–424, 2004.

$$(t_h, \mathrm{MSB}_\eta(\alpha t_h)), \qquad h = 1, \ldots, d,$$

for some $\eta > 0$.

For $\mathcal{G} = \mathbb{F}_p^*$ this problem has been introduced and studied by Boneh and Venkatesan [3,4]. In [3] a polynomial time algorithm is designed which recovers a for some $\eta \sim (\log p)^{1/2}$ and $k = O(\log^{1/2} p)$. The algorithm of [3] has been extended in several directions. In particular, in [8] it is generalised to all sufficiently large subgroups $\mathcal{G} \subseteq \mathbb{F}_p^*$. This and other generalisations have led to a number of cryptographic applications, see [20,21,22]. Using bounds of exponential sums from [9,11] it has been shown that the algorithm of [3] works for subgroups $\mathcal{G} \subseteq \mathbb{F}_p^*$ of order $\#\mathcal{G} \ge p^{\nu+\varepsilon}$ where for any $\varepsilon > 0$ and sufficiently large p one can take

- $\nu = 1/3$ for all primes,
- $\nu = 0$ for almost all primes p.

Using a recent improvement of [5] of the bounds of exponential sums over small subgroups of $\mathbb{F}_p^*$ one can obtain the same result with $\nu = 0$ for all primes p and thus extend the results of [3,8] to subgroups of order $\#\mathcal{G} \ge p^\varepsilon$.

For $\mathcal{G} = \mathbb{F}_p^*$ in [4] an algorithm is constructed which works with much smaller values $\eta \sim \log\log p$, however this algorithm is non-uniform. This means that if the points $t_1, \ldots, t_k \in \mathcal{G}$ are known in advance, one can design (in exponential time) a certain data structure, that now given k values $\mathrm{MSB}_\eta(\alpha t_i)$, $i = 1, \ldots, k$, the hidden number α can be found in polynomial time. In the present paper we extend the algorithm of [4] to essentially arbitrary subgroups of $\mathbb{F}_p^*$. As in [4] we discuss possible applications of our algorithm to proving bit security results for several exponentiation based cryptographic schemes.

As in [3,4], the method is based on some properties of lattices, but also makes use of exponential sums, however not in such a direct way as in [8]. Namely, we introduce certain new arguments allowing to amplify the uniformity of distribution properties of small subgroups $\mathcal{G}$. This allows us to use the bound of exponential sums from [10] with elements of $\mathcal{G}$, which is very moderate in strength (and does not imply any uniformity of distribution properties of $\mathcal{G}$ which would be the crucial argument of the method of [8]). The bound of [10] has however the very important advantage over the bounds of [5,9,11] that it applies to subgroups of order

$$\#\mathcal{G} \ge \frac{\log p}{(\log\log p)^{1-\varepsilon}}.$$

It is interesting to note that our approach has links with the famous *Waring problem* which has been studied in number theory for several hundred years. In fact, the Waring problem in finite fields has been the main motivation of the bound of exponential sums of [10] which we use in this paper. For surveys of recent results on this problem see [6,10,25]. We also remark that a uniform algorithm, which is also based on a similar use of the bound of [10] and which improves the results of [8], has recently been proposed in [23].

Throughout the paper $\log x$ always denotes the binary logarithm of $x > 0$ and the constants in the 'O'-symbols may occasionally, where obvious, depend on a small positive parameter ε and are absolute otherwise. We always assume that p is a prime number with $p \geq 5$, thus the expression $\log\log p$ is defined (and positive).

Acknowledgements: The first author was supported in part by ARC grant DP0211459. The second author was supported in part by DSTA grant R-394-000-011-422, by the Austrian Academy of Sciences, and by FWF grant S8313.

2 Exponential Sums and Distribution of Short Sums of Elements of Subgroups

For a complex z we put $\mathbf{e}_p(z) = \exp(2\pi i z/p)$.

Let $T = \#\mathcal{G}$, $T|(p-1)$, be the cardinality of a subgroup $\mathcal{G} \subseteq \mathbb{F}_p^*$. If we put $n = (p-1)/T$ then each element $r \in \mathcal{G}$ has exactly n representations $r = x^n$ with $x \in \mathbb{F}_p^*$. Therefore, for any $\lambda \in \mathbb{F}_p$,

$$\sum_{r \in \mathcal{G}} \mathbf{e}_p(\lambda r) = \frac{T}{p-1} \sum_{x \in \mathbb{F}_p^*} \mathbf{e}_p(\lambda x^n).$$

Now by Theorem 1 of [10] we have the following bound, see also [6,11].

Lemma 1. *For any $1 > \varepsilon > 0$ there exists a constant $c(\varepsilon) > 0$ such that for any subgroup $\mathcal{G} \subseteq \mathbb{F}_p^*$ of order*

$$T \geq \frac{\log p}{(\log\log p)^{1-\varepsilon}}$$

the bound

$$\max_{\gcd(\lambda,p)=1} \left| \sum_{r \in \mathcal{G}} \mathbf{e}_p(\lambda r) \right| \leq T \left(1 - \frac{c(\varepsilon)}{(\log p)^{1+\varepsilon}}\right)$$

holds.

For an integer $k \geq 1$, a subgroup $\mathcal{G} \subseteq \mathbb{F}_p^*$ and $t \in \mathbb{F}_p$ we denote by $N_k(\mathcal{G}, t)$ the number of solutions of the equation

$$r_1 + \ldots + r_k \equiv t \pmod p, \qquad r_1, \ldots, r_k \in \mathcal{G}.$$

Recalling the relation between the set of nth powers, where $n = (p-1)/T$, we see that studying the above congruence is equivalent to studying the congruence

$$x_1^n + \ldots + x_k^n \equiv t \pmod p, \qquad x_1, \ldots, x_k \in \mathbb{F}_p^*.$$

The problem of finding the smallest possible value of k for which the congruence (or in more traditional settings the corresponding equation over $\mathbb{Z}$) has a solution for any t is known as the Waring problem. However for our purposes just a

solvability is not enough. Rather we need an asymptotic formula for the number of solutions.

We show that Lemma 1 can be used to prove that for reasonably small k, $N_k(\mathcal{G}, t)$ is close to its expected value.

Lemma 2. *For any $1 > \varepsilon > 0$ there exists a constant $C(\varepsilon) > 0$ such that for any subgroup $\mathcal{G} \subseteq \mathbb{F}_p^*$, of order*

$$T \geq \frac{\log p}{(\log \log p)^{1-\varepsilon}}$$

the bound

$$\max_{t \in \mathbb{F}_p} \left| N_k(\mathcal{G}, t) - \frac{T^k}{p} \right| \leq \frac{T^k}{p^2}$$

holds for any integer $k \geq C(\varepsilon)(\log p)^{2+\varepsilon}$.

Proof. The well-known identity (see for example [14, Chapter 5.1])

$$\sum_{\lambda=0}^{p-1} \mathbf{e}_p(\lambda u) = \begin{cases} 0, & \text{if } u \not\equiv 0 \pmod p, \\ p, & \text{if } u \equiv 0 \pmod p, \end{cases}$$

implies that

$$\begin{aligned} N_k(\mathcal{G}, a) &= \sum_{r_1, \ldots, r_k \in \mathcal{G}} \frac{1}{p} \sum_{\lambda=0}^{p-1} \mathbf{e}_p\left(\lambda(r_1 + \ldots + r_k - t)\right) \\ &= \frac{1}{p} \sum_{\lambda=0}^{p-1} \mathbf{e}_p(-\lambda t) \left(\sum_{r \in \mathcal{G}} \mathbf{e}_p(\lambda r) \right)^k. \end{aligned}$$

Separating the term T^k/p, corresponding to $\lambda = 0$, and applying Lemma 1 to other terms, we obtain

$$\max_{t \in \mathbb{F}_p} \left| N_k(\mathcal{G}, t) - \frac{T^k}{p} \right| \leq T^k \left(1 - \frac{c(\varepsilon)}{(\log p)^{1+\varepsilon}}\right)^k = T^k \exp\left(O\left(k(\log p)^{-1-\varepsilon}\right)\right)$$

and the desired result follows. □

3 Rounding in Lattices

Let $\mathcal{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_s)^T \in \mathbb{R}^{s \times s}$ be a nonsingular $s \times s$ matrix over the set of real numbers $\mathbb{R}$ with rows $\mathbf{b}_1, \ldots, \mathbf{b}_s$. The set of vectors

$$\mathcal{L} = \left\{ \sum_{i=1}^{s} n_i \mathbf{b}_i \mid n_i \in \mathbb{Z} \right\},$$

spanned by the rows of $\mathcal{B}$, is called an s-dimensional full rank lattice associated with the matrix $\mathcal{B}$. The set $\{\mathbf{b}_1, \ldots, \mathbf{b}_s\}$ is called a *basis* of $\mathcal{L}$.

One of the most fundamental problems in this area is the *closest vector problem*. This problem can be defined with respect to any vector norm $\|w\|$ as follows: given a basis of a lattice $\mathcal{L}$ in $\mathbb{R}^s$ and a target vector $\mathbf{u} \in \mathbb{R}^s$, find a lattice vector $\mathbf{v} \in \mathcal{L}$ with

$$\|\mathbf{u} - \mathbf{v}\| = \operatorname{dist}(\mathbf{u}, \mathcal{L})$$

where

$$\operatorname{dist}(\mathbf{u}, \mathcal{L}) = \min\{\|\mathbf{u} - \mathbf{z}\| \mid \mathbf{z} \in \mathcal{L}\}.$$

It is well known that the closest vector problem in the Euclidean norm is NP-hard (see [16,17] for references). However, its approximate version [2] admits a polynomial time algorithm which goes back to the lattice basis reduction algorithm of Lenstra, Lenstra and Lovász [12], see also [1] for more recent developments.

However, it has been noticed in [4] that for some special class of lattices a simple rounding technique gives an exact solution to the closest vector problem. Here we summarise several results from [4] which underlie this technique and its applications to the hidden number problem.

For our purposes the L_1-norm is most relevant thus from now on we always assume that $\|\mathbf{w}\| = \sum_{i=1}^{s} |w_i|$ is the L_1-norm of $\mathbf{w} = (w_1, \ldots, w_s) \in \mathbb{R}^s$, in particular $\operatorname{dist}(\mathbf{u}, \mathcal{L})$ is always assumed to be defined with respect to the L_1-norm.

Given a target vector $\mathbf{u} \in \mathbb{R}^s$, using standard linear algebra tools, we find its representation in the basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_s\}$

$$\mathbf{u} = \sum_{i=1}^{s} w_i \mathbf{b}_i$$

and then put

$$\lfloor \mathbf{u} \rceil = \sum_{i=1}^{s} \lfloor w_i \rceil \, \mathbf{b}_i$$

where for $w \in \mathbb{R}$, $\lfloor w \rceil$ denotes the closest integer (in the case of $2w \in \mathbb{Z}$ we put $\lfloor w \rceil = \lfloor w \rfloor$). Clearly, $\lfloor \mathbf{u} \rceil \in \mathcal{L}$ but certainly it is not the closest (or even just a close) vector.

Now, for a matrix $\mathcal{C} \in \mathbb{R}^{s \times s}$ with columns $\mathbf{c}_1^T, \ldots, \mathbf{c}_s^T$, we introduce the following measure

$$\rho(\mathcal{C}) = \max_{1 \le j \le s} \|\mathbf{c}_j\|.$$

The following statement, which is essentially [4, Lemma 2.1], gives a sufficient condition under which $\lfloor \mathbf{u} \rceil$ is a solution to the closest vector problem for $\mathbf{u}$.

Lemma 3. *If*

$$\rho\left(\mathcal{B}^{-1}\right) < \frac{1}{2\,\mathrm{dist}\,(\mathbf{u}, \mathcal{L})}$$

then

$$\|\mathbf{u} - \lfloor \mathbf{u} \rceil\| = \mathrm{dist}\,(\mathbf{u}, \mathcal{L}).$$

We consider the lattice $\mathcal{L}(t_1, \ldots, t_d)$ spanned by the rows of the matrix

$$\mathcal{B}(t_1, \ldots, t_d) = \begin{pmatrix} p & 0 & \ldots & 0 & 0 \\ 0 & p & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & 0 & \ldots & p & 0 \\ t_1 & t_2 & \ldots & t_d & 1 \end{pmatrix}.$$

The next statement follows from [4, Theorem 2.2].

Lemma 4. *Let p be a prime and $d > 4 + \log p + \log\log p$. Let $t_1, \ldots, t_d \in \{0, 1, \ldots, p-1\}$ be integers chosen uniformly and independently at random. Then with probability at least 1/2 there exists a basis of the lattice $\mathcal{L}(t_1, \ldots, t_d)$ spanned by rows of a certain matrix $\mathcal{C}$ with entries of polynomial size $(\log p)^{O(1)}$ and with*

$$\rho(\mathcal{C}^{-1}) < \frac{3d \log p}{p}.$$

4 Nonuniform Algorithm

For an integer w we denote by $\lfloor w \rfloor_p$ the remainder of w on division by p.

Assume that for $\alpha \in \mathbb{F}_p^*$ and a subgroup $\mathcal{G} \subseteq \mathbb{F}_p^*$ of order T, generated by $g \in \mathbb{F}_p^*$, we are given an *oracle* $\mathcal{HNP}_\mu$ such that for every $x \in \{0, 1, \ldots, T-1\}$, it returns $\mathrm{MSB}_\mu\left(\lfloor \alpha g^x \rfloor_p\right)$.

Theorem 1. *For any $1 > \varepsilon > 0$ there exists a constant $a(\varepsilon) > 0$ such that, for $\mu = a(\varepsilon) \log\log p$, for any $g \in \mathbb{F}_p^*$ of order*

$$T \geq \frac{\log p}{(\log\log p)^{1-\varepsilon}}$$

after taking a polynomial number $(\log p)^{O(1)}$ of advice bits depending only on p and $\mathcal{G}$ but independent on α, one can design a deterministic algorithm which makes $O\left((\log p)^{3+\varepsilon}\right)$ calls of the oracle $\mathcal{HNP}_\mu$ and then recovers α in polynomial time.

Proof. Put

$$d = 5 + \lfloor \log p + \log\log p \rfloor, \qquad k = \lceil C(\varepsilon)(\log p)^{2+\varepsilon} \rceil,$$

where $C(\varepsilon)$ is given by Lemma 2.

The advice bits which we request describe:

- the values of $t_1, \ldots, t_d \in \mathbb{F}_p$ for which the lattice $\mathcal{L}(t_1, \ldots, t_d)$ is spanned by a matrix $\mathcal{C}$ with
$$\rho(\mathcal{C}^{-1}) < \frac{3d \log p}{p},$$
which exist by Lemma 4, and the above matrix $\mathcal{C}$;
- the exponents x_{hj}, $h = 1, \ldots, d$, $j = 1, \ldots, k$ with
$$t_h \equiv \sum_{j=1}^{k} g^{x_{hj}} \pmod p, \qquad h = 1, \ldots, d,$$
which exist by Lemma 2.

We call the oracle with $x = 0$ getting an approximation $u_0 = \mathrm{MSB}_\mu(\alpha)$. Now we call the oracle $\mathcal{HNP}_\mu$ for the dk integers

$$r_{hj} = g^{x_{hj}} \in \mathcal{G}, \qquad j = 1, \ldots, k, \ h = 1, \ldots, d,$$

and get integers u_{hj} with

$$\left|\lfloor \alpha r_{hj} \rfloor_p - u_{hj}\right| < p/2^{\mu+1}, \qquad h = 1, \ldots, d, \ j = 1, \ldots, k.$$

For $h = 1, 2, \ldots, d$ we put

$$v_h = \sum_{j=1}^{k} \lfloor \alpha r_{hj} \rfloor_p, \qquad t_h = \left\lfloor \sum_{j=1}^{k} r_{hj} \right\rfloor_p, \qquad u_h = \sum_{j=1}^{k} u_{hj},$$

where all the additions are over $\mathbb{Z}$.

Note that for sufficiently large p,

$$|v_h - u_h| < kp/2^{\mu+1} \le p/2^{\eta+1},$$

where

$$\eta = \mu - \log k \ge \log\left(3d(d+1)\log p\right).$$

for an appropriate value of $a(\varepsilon)$ and sufficiently large p.

Letting $\mathbf{u} = (u_1, \ldots, u_d, u_0)$, we obtain

$$\mathrm{dist}\ (\mathbf{u}, \mathcal{L}(t_1, \ldots, t_d)) \le (d+1)p/2^{\eta+1}.$$

Therefore,

$$\rho(\mathcal{C}^{-1}) < \frac{3d \log p}{p} \le \frac{2^\eta}{(d+1)p} \le \frac{1}{2\,\mathrm{dist}\ (\mathbf{u}, \mathcal{L}(t_1, \ldots, t_d))}$$

and the result follows by Lemma 3. □

5 Application to Diffie-Hellman Related Schemes

Our result applies to the establishing bit security of the same exponentiation based cryptographic schemes as those of [4]. Such schemes include, but are not limited to, the Okamoto conference sharing scheme and a certain modification of the ElGamal scheme, see [4] for more details.

The main distinction between our result and that of [4] is that we do not need anymore assume that the generating element is a primitive root, which is a rather impractical assumption. Indeed, in practical applications of the Diffie-Hellman and other related schemes, one would probably choose a subgroup of $\mathbb{F}_p^*$ of prime order T. Moreover, it is quite reasonable to choose T of size about $\exp\left(c(\log p)^{1/3}(\log\log p)^{2/3}\right)$ for some constant $c > 0$, in order to balance time complexities of the number field sieve based attacks and Pollard's rho-method based attacks, see [7,15,18,19,24]. Thus our result closes the gap between the settings of [4] and settings more relevant to practical usage of the above schemes.

It also seems to be plausible that one can obtain similar, albeit slightly weaker, results for other cryptographically interesting subgroups in finite fields and rings, for which relevant bounds of character sums are available. For example, such bounds are known for XTR subgroups, see [13].

References

1. M. Ajtai, R. Kumar, and D. Sivakumar, 'A sieve algorithm for the shortest vector problem', *Proc. 33rd ACM Symp. on Theory of Comput.,* Crete, Greece, 2001, 601–610.
2. L. Babai, 'On Lovasz' lattice reduction and the nearest lattice point problem', *Combinatorica,* **6** (1986), 1–13.
3. D. Boneh and R. Venkatesan, 'Hardness of computing the most significant bits of secret keys in Diffie–Hellman and related schemes', *Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **1109** (1996), 129–142.
4. D. Boneh and R. Venkatesan, 'Rounding in lattices and its cryptographic applications', *Proc. 8th Annual ACM-SIAM Symp. on Discr. Algorithms,* SIAM, 1997, 675–681.
5. J. Bourgain and S. V. Konyagin, 'Estimates for the number of sums and products and for exponential sums over subgroups in fields of prime order', *Comptes Rendus Mathematique,* **337** (2003), 75–80.
6. T. Cochrane, C. Pinner, and J. Rosenhouse, 'Bounds on exponential sums and the polynomial Waring's problem mod p', *Proc. Lond. Math. Soc.,* **67** (2003), 319–336.
7. R. Crandall and C. Pomerance, *Prime numbers: A Computational perspective,* Springer-Verlag, Berlin, 2001.
8. M. I. González Vasco and I. E. Shparlinski, 'On the security of Diffie–Hellman bits', *Proc. Workshop on Cryptography and Computational Number Theory, Singapore 1999,* Birkhäuser, 2001, 257–268.
9. D. R. Heath-Brown and S. V. Konyagin, 'New bounds for Gauss sums derived from kth powers, and for Heilbronn's exponential sum', *Quart. J. Math.,* **51** (2000), 221–235.

10. S. V. Konyagin, 'On estimates of Gaussian sums and the Waring problem modulo a prime', *Trudy Matem. Inst. Acad. Nauk USSR,* Moscow, **198** (1992), 111–124 (in Russian); translation in *Proc. Steklov Inst. Math.,* **1** (1994), 105–117.
11. S. V. Konyagin and I. Shparlinski, *Character sums with exponential functions and their applications,* Cambridge Univ. Press, Cambridge, (1999).
12. A. K. Lenstra, H. W. Lenstra, and L. Lovász, 'Factoring polynomials with rational coefficients', *Mathematische Annalen,* **261** (1982), 515–534.
13. W.-C. W. Li, M. Näslund, and I. E. Shparlinski, 'The hidden number problem with the trace and bit security of XTR and LUC', *Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **2442** (2002), 433–448.
14. R. Lidl and H. Niederreiter, *Finite fields,* Cambridge University Press, Cambridge, (1997).
15. A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography,* CRC Press, Boca Raton, FL, 1996.
16. P. Q. Nguyen and J. Stern, 'Lattice reduction in cryptology: An update', *Lect. Notes Comp. Sci.,* Springer-Verlag, Berlin, **1838** (2000), 85–112.
17. P. Q. Nguyen and J. Stern, 'The two faces of lattices in cryptology', *Lect. Notes Comp. Sci.,* Springer-Verlag, Berlin, **2146** (2001), 146–180.
18. O. Schirokauer, 'Discrete logarithms and local units', *Philos. Trans. Roy. Soc. London, Ser. A,* **345** (1993), 409–423.
19. O. Schirokauer, D. Weber, and T. Denny, 'Discrete logarithms: The effectiveness of the index calculus method', *Lect. Notes in Comp. Sci.,* Springer-Verlag, Berlin, **1122** (1996), 337–362.
20. I. E. Shparlinski, 'Playing "Hide-and-Seek" in finite fields: Hidden number problem and its applications', *Proc. 7th Spanish Meeting on Cryptology and Information Security, Vol.1,* Univ. of Oviedo, 2002, 49–72.
21. I. E. Shparlinski, 'Exponential sums and lattice reduction: Applications to cryptography', *Finite Fields with Applications to Coding Theory, Cryptography and Related Areas,* Springer-Verlag, Berlin, 2002, 286–298.
22. I. E. Shparlinski, *Cryptographic applications of analytic number theory,* Birkhäuser, 2003.
23. I. E. Shparlinski and A. Winterhof, 'Hidden number problem in small subgroups', *Preprint,* 2003.
24. D. R. Stinson, *Cryptography: Theory and practice,* CRC Press, Boca Raton, FL, 2002.
25. A. Winterhof, 'On Waring's problem in finite fields', *Acta Arith.,* **87** (1998), 171–177.

Cryptographic Randomized Response Techniques

Andris Ambainis[1], Markus Jakobsson[2], and Helger Lipmaa[3]

[1] CS Division, University of California, Berkeley, CA 94720, USA
ambainis@cs.berkeley.edu
[2] RSA Laboratories, 174 Middlesex Turnpike, Bedford, MA 01730, USA
mjakobsson@rsasecurity.com
[3] Laboratory for Theoretical CS, Department of CS&E
Helsinki University of Technology, P.O.Box 5400, FIN-02015 HUT, Espoo, Finland
helger@tcs.hut.fi

Abstract. We develop cryptographically secure techniques to guarantee unconditional privacy for respondents to polls. Our constructions are efficient and practical, and are shown not to allow cheating respondents to affect the "tally" by more than their own vote—which will be given the exact same weight as that of other respondents. We demonstrate solutions to this problem based on both traditional cryptographic techniques and quantum cryptography.
Keywords: binary symmetric channel, oblivious transfer, polling, privacy, privacy-preserving data-mining, randomized response technique

1 Introduction

In some instances, privacy is a matter of keeping purchase information away from telemarketers, competitors, or other intruders. In other instances, privacy translates to security against traffic analysis, such as for web browsing; or to security of personal location information. In still other instances, which we study in this paper, privacy is a *precondition* to being able to obtain answers to important questions. Two concrete examples of instances of latter are *elections* and *surveys/polls*.

While the first of these examples is the one of the two that has received—by far—the most attention in the field of cryptography, there are important reasons to develop better privacy tools for polling. Surprisingly, the two examples (namely, elections and polls), while quite similar at a first sight, are very different in their requirements. Since it is typically the case that there is more funding available for providing privacy in elections than in surveys and polls, it follows that the tallying process in the former may involve more costly steps than that in the latter—whether the process is electronic (using, e.g., mix networks) or mechanic. Second, while in the case of the voting scheme, we have that users need to entrust their privacy with some set of authorities, it is often the case that there is less trust established between the parties in polls. Yet another reason to treat the two situations separately is that elections involve many more respondents than polls typically do, thereby allowing a unique opinion (e.g., vote) to be hidden among many more in the case of elections than in the case of polls. Finally, while elections require as exact tallying as is possible, *statistical truths* are both sufficient and desirable in polls. This allows the use of polling techniques that are very different from election techniques—in terms of their cost; how tallying is done; and how privacy is protected.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 425–138, 2004.

While not given much attention in cryptography, important work on polling has been done in statistics. In particular, the *randomized response technique* (RRT) was proposed by Warner [War65] in 1965, with the goal of being used in polls relating to sensitive issues, such as drug abuse, sexual preferences and shoplifting. The underlying idea behind Warner's proposal (alternative RRTs have been proposed since then) is for respondents to randomize each response according to a certain, and known, probability distribution. More precisely, they answer the question truthfully with some probability $p_{\mathsf{ct}} > 1/2$, while with a fixed and known probability $1 - p_{\mathsf{ct}}$ they lie. Thus, users can always claim that their answer—if it is of the "incriminating" type—was a lie. When evaluating all the answers of the poll, these lies become statistically insignificant given a large enough sample (where the size of the sample can be simply computed from the probability distribution governing lying.)

However, a pure RRT by itself is not well suited for all types of polls. E.g., it is believed that people are more likely to vote for somebody who leads the polls than somebody who is behind. Therefore, it could be politically valuable not to lie (as required by the protocol) in polls relating to one's political opinion, and therefore have one's "vote" assigned a greater weight. (This is the case since people with the opposite opinion—if honestly following the protocol—will sometimes cast a vote according to your opinion, but you would never cast a vote according to their opinion, assuming you are willing to cheat.) While the results of the poll remain meaningful if *everybody* cheats (i.e., tells the truth with a probability different from that specified by the protocol), this is *not* the case when only some people deviate from the desired behavior. Also, while one might say that the increased weight in the polls is gained at the price of the cheater's privacy, this is not necessarily the case if the cheater *claims* to have followed the protocol, and there is no evidence to the contrary.

To address the problem of cheating respondents in RRT, we propose the notion of *cryptographic randomized response technique* (CRRT), which is a modification of RRT that prevents cheating. We present three efficient protocols for CRRT; two of them using classic cryptographic methods (and being efficient for different values of p_{ct}), and one using quantum methods. Importantly, the quantum RRT protocol is implementable by using contemporary technology. We give rigorous proofs of security for one of the classical protocols and for the quantum protocol.

For all of our proposed solutions, the privacy of the respondent will be guaranteed information-theoretically (more precisely, statistically). This is appropriate to stimulate truthful feedback on topics that may affect the respondent for years, if not decades. All proposed solutions also *guarantee* that the respondents reply based on the desired probability distributions. Clearly, this requires that the respondent cannot determine the outcome of the protocol (as viewed by the interviewer) before the end of the protocol. Otherwise, he could simply halt the execution of the protocol to suppress answers in which the communicated opinion was a lie. We will therefore require protocols to offer privacy for the *interviewer* as well as for the respondent, meaning that the respondent cannot learn what the outcome of the protocol is, as seen by the interviewer. (One could relax this requirement slightly to allow the respondent to learn the outcome at the same time as the interviewer does, or afterward.)

While we believe that it is important to prevent the respondent from biasing the outcome by selective halting (corresponding to the protocol being *strongly secure*), we also describe simplified versions of our protocols in which this protection mechanism is not available. Such simplified versions (which we refer to as *weakly secure)* can still be useful in some situations. They may, for example, be used as the default scheme for a given application—where they would be replaced by their strongly secure relatives if too many interactions are halted prematurely. (The decision of when the shift would be performed should be based on standard statistical methods, and will not be covered herein.) The benefit of considering such dual modes is that the weakly secure versions typically are computationally less demanding than the strongly secure versions.

Finally, we also discuss cryptographic enhancements to two alternative RRT techniques. In the first, referred to as RRT-IQ, the respondent always gives the truthful answer to the question he is presented with. However, with a certain probability, he is presented with an Innocous Question instead of the intended question. A second alternative RRT technique is what is referred to as *polychotomous* RRT. In this version of RRT, the respondent is given more than two possible options per question.

Other Applications. Our first protocol uses a novel protocol for information-theoretically secure *verifiable oblivious transfer* that enables easier zero-knowledge proofs on the properties of the transferred values. The new verifiable oblivious transfer protocol may also be useful in other applications. While our main designated application is polling, our techniques have also several other applications, in particular in the privacy-preserving data-mining. They are also related to several fundamental cryptographic problems. For example, our protocols Wagner's technique are also efficient implementations of the *verifiable binary symmetric channel.* (See Section 3.)

New Verifiable Commitment Scheme. One of our RRT protocols uses a novel (and as far as we know, the first) two-round verifiable commitment scheme based on the (non-verifiable) commitment scheme by Naor and Pinkas [NP01]. Verifiable commitment schemes have a huge range of applications.[4]

Outline. We first review the details of the randomized response technique (Section 2), after which we review some related work in cryptography (Section 3). We then introduce the cryptographic building blocks of our protocols (Section 4). We then describe the functionality of our desired solution in terms of functional black boxes and protocol requirements (Section 5). In Section 6, we present our secure CRRT protocols. In Section 7 we describe cryptographic solutions to other variants of the standard RRT. The appendix contains additional information about the new oblivious transfer protocol and about the quantum RRT protocol.

2 Short Review of Randomized Response Technique

When polling on sensitive issues like sexual behavior or tax evasion, respondents often deny their stigmatizing behavior due to the natural concern about their privacy. In

[4] Slightly more efficient and recent verifiable commitment schemes that draw ideas from this paper were proposed by the third author in [Lip03b]. The new schemes can be seamlessly plugged into our first RRT protocol.

1965, Warner [War65] proposed the Randomized Response Technique (RRT) for organization of polls where an unbiased estimator (UE, defined in any standard statistics textbook) to the summatory information —the proportion of people belonging to a stigmatizing group A—can be recovered, while the privacy of every individual respondent is protected statistically. Since then, different variations of the RRT have been proposed in statistics, see [CM88] for a survey. These different variations provide, for example, smaller variance, smaller privacy breaches, optimality under different definitions of privacy, and ability to answer polychotomous questions. Next we will give a short overview of three types of RRT.

RRT-W. In Wagner's original method (RRT-W), the respondents provide a truthful answer to the question "Do you belong to a stigmatizing group A?" with a certain fixed and publicly known probability $p_{\mathsf{ct}} > 1/2$. With probability $1 - p_{\mathsf{ct}}$ they lie—i.e., answer the opposite question. Define π_A to be the true proportion of the population that belongs to A (or whose *type* is $t = 1$). Let p_{yes} be the proportion of "yes" responses in the poll. In RRT-W, the a *priori* probability of getting a "yes" response is $p_{\mathsf{yes}} = p_{\mathsf{ct}} \cdot \pi_A + (1 - p_{\mathsf{ct}})(1 - \pi_A)$. In the case of N players, L of which answer "yes", an UE of p_{yes} is $\widehat{p_{\mathsf{yes}}} = L/N$, the sample proportion of "yes" answers. From this, one can simply compute the unbiased estimator of π_A. This equals $\widehat{\pi_A} = \frac{\widehat{p_{\mathsf{yes}}} - (1-p_{\mathsf{ct}})}{2p_{\mathsf{ct}}-1} = \frac{p_{\mathsf{ct}}-1}{2p_{\mathsf{ct}}-1} + \frac{L}{N} \cdot \frac{1}{(2p_{\mathsf{ct}}-1)}$. Similarly, the variance $\mathrm{var}(\widehat{\pi_A})$ and its UE can be computed.

RRT-IQ. An alternative RRT is the *innocuous question method* (RRT-IQ), first analyzed in [GASH69]. When using RRT-IQ, the respondent answers the sensitive question with a probability p_{ct}, while with probability $1 - p_{\mathsf{ct}}$ to an unrelated and innocuous question, such as "Flip a coin. Did you get tails?". The RRT-IQ achieves the same goals as RRT-W but with less variance [CM88], which makes it more suitable for practical polling. Many other RRT-IQs are known, including some with unknown estimate of the the proportion of the population belonging to the innocuous group.

PRRT. The RRTs for dichotomous polling (where the answer is yes or no) can be generalized to *polychotomous RRT* (PRRT) where the respondent can belong to one of the m mutually exclusive groups $A_1, \ldots, A_m$, some of which are stigmatizing. A typical sensitive question of this kind is "When did you have your first child?", with answers "1—while not married", "2—within 9 months after the wedding" and "3—more than 9 months after the wedding". In many cultures, the answer 1 is stigmatizing, the answer 3 is innocuous, while the answer 2 is somewhere inbetween. The interviewer wants to know an UE for the proportion π_i of people who belong to the group A_i, $i \in [1, m]$. There are many possible PRRTs [CM88, Chapter 3]. One of the simplest is the following technique PRRT-BD by Bourke and Dalenius [CM88]: first fix the probabilities p_{ct} and $p_1, \ldots, p_m$, such that $p_{\mathsf{ct}} + \sum_{i \in [1,m]} p_i = 1$. A respondent either reveals her true type $t \in [1, m]$ with probability p_{ct}, or answers $i \in [1, m]$ with probability p_i. To recover an UE of $\boldsymbol{\pi} := (\pi_1, \ldots, \pi_m)^T$, define $\boldsymbol{p} := (p_1, \ldots, p_m)^T$ and $\boldsymbol{p}_{\mathsf{ans}} = (p_{\mathsf{ans}_1}, \ldots, p_{\mathsf{ans}_m})^T$, where p_{ans_i} is the proportion of people who answer i. Then $\boldsymbol{p}_{\mathsf{ans}} = p_{\mathsf{ct}} \cdot \boldsymbol{\pi} + \boldsymbol{p}$, and hence $\widehat{\boldsymbol{\pi}} = p_{\mathsf{ct}}^{-1} \cdot (\widehat{\boldsymbol{p}_{\mathsf{ans}}} - \boldsymbol{p})$.

3 Related Cryptographic Work

In [KANG99], Kikuchi et al. propose techniques with similar goals as ours. Unaware of the previous work on RRT, the authors reinvent this notion, and propose a protocol for performing the data exchange. However, their protocol is considerably less efficient than ours. Also, it does not offer strong security in our sense. This vulnerability makes their protocol unsuitable for their main application (voting), as well as polls where respondents may wish to bias their answer. Our protocols can be used in their framework.

The cryptographic RRT-W protocol can be seen as an implementation of an *verifiable BSC,* based on either verifiable oblivious transfer or more generally on a suitable commitment scheme. (Protocols for other RRTs implement even more complex channels.) Crépeau and Kilian have showed how to construct (nonverifiable) oblivious transfer protocols and commitment schemes from a (nonverifiable) BSC [CK88, Cré97], but their opposite reductions are less efficient.

There is a very close relationship between our protocols and protocols for oblivious transfer and for the fractional oblivious transfer [BR99]. While our goals are orthogonal to those of oblivious transfer, the techniques are hauntingly similar. In particular, one of our CRRT protocols uses a protocol for oblivious transfer as a building block. While in principle *any* such protocol can be used, it is clear that the properties of the building block will be inherited by the main protocol. Therefore, in order to provide unconditional guarantees of privacy for the respondents, we use a *verifiable* variant of the information theoretic protocol for oblivious transfer, namely that proposed by Naor and Pinkas [NP01]. We leave it as an open question whether the fractional oblivious transfer protocols of [BR99] (that essentially implement verifiable *erasure channel)* can be modified to work in our scenario (where we need to implement verifiable BSC in the case of RRT-W and related information channels without erasure in the case of other RRT protocols) or our protocols can be modified to work in their scenario; at least the first seems clearly not to be the case.

Furthermore, our work is related to the work on Private Information Retrieval (PIR) in that the goal of our interviewer is to retrieve some element from the respondent, without the latter learning what was retrieved. More specifically, if some ℓ out of n elements represent the respondent's opinion, and the remaining $n - \ell$ elements represent the opposite opinion, then the interviewer will learn the respondent's opinion with probability ℓ/n if he retrieves a random element. Of course, in order to guarantee the interviewer that the elements are correctly formed, additional mechanisms are required.

In privacy-preserving data-mining a related data randomization approach has been proposed: namely, the users input their data to the central database (e.g., a loyal customer inputs the name of the product he bought), and the database maintainer needs to do some statistical analysis on the database. However, the maintainer should not be able to recover individual items. Database randomization in the case when the maintainer is limited to the SUM function corresponds exactly to the RRT. For the same reasons as in the RRT, one should not be able to bias the data. Our protocols are also applicable in the privacy-preserving data-mining.

4 Cryptographic Building Blocks

Define $[a, b] := \{a, a+1, \ldots, b-1, b\}$. In the rest of this section we present some cryptographic building blocks that will be used in our CRRT protocols. Throughout this paper, assume that p is a large prime, and q, $q \mid (p-1)$, is another prime. Then $\mathbb{Z}_p^*$ has a unique subgroup G of order q. Let g and h be two generators of G, such that nobody knows their mutual discrete logarithms $\log_g h$ and $\log_h g$. We let k be the security parameter, in our setting we can take $k = q$. In the next two protocols (the Pedersen's commitment scheme and the Naor-Pinkas oblivious transfer protocol), the key K consists of public parameters, $K := (g; h)$.

Pedersen's Commitment Scheme. In this scheme [Ped91], a message $\mu \in \mathbb{Z}_q$ is committed by drawing a random $\rho \leftarrow_R \mathbb{Z}_q$, and setting $\mathsf{C}_K(\mu; \rho) := g^\mu h^\rho$. The commitment can be opened by sending μ and ρ to the verifier. This scheme is *homomorphic,* i.e., $\mathsf{C}_K(\mu; \rho)\mathsf{C}_K(\mu'; \rho') = \mathsf{C}_K(\mu + \mu'; \rho + \rho')$. Since it is also perfectly hiding and computationally binding, it can be used as a building block in efficient zero-knowledge arguments, such as protocols for arguing the knowledge of plaintext μ.

Verifiable 1-out-of-n Oblivious Transfer. In an $\binom{1}{n}$-oblivious transfer (OT) protocol, the sender $\mathcal{R}$ has private input $\mu = (\mu_1, \ldots, \mu_n) \subset M^n$ (and no private output) for some set M, while the chooser $\mathcal{I}$ has private input $\sigma \in [1, n]$ and private output μ_σ. The oblivious transfer (OT) protocol by Naor and Pinkas [NP01] guarantees information-theoretic privacy for $\mathcal{R}$, and computational privacy for $\mathcal{I}$. Intuitively, the in the Naor-Pinkas protocol, the sender oblivious-transfers one encryption key v_σ that is used to encrypt the actual database element μ_σ. The Naor and Pinkas [NP01] paper does not specify the encryption method, mentioning only that the encryption scheme must be semantically secure.

We propose to use Pedersen's commitment scheme instead of an encryption scheme. Let $K = (g; h)$ be the public key of the commitment scheme. The proposed variant of the Naor-Pinkas protocol works as follows:

1. $\mathcal{I}$ generates random $a, b \leftarrow \mathbb{Z}_q$ and sends $(A, B, C) \leftarrow (g^a, g^b, g^{ab-\sigma+1})$ to $\mathcal{R}$.
2. $\mathcal{R}$ performs the following, for $i \in [1, n]$: Generate random (r_i, s_i). Compute $w_i \leftarrow g^{r_i} A^{s_i}$, compute an encryption $y_i \leftarrow \mathsf{C}_K(\mu_i; v_i \mod q)$, where $v_i \leftarrow B^{r_i}(C \cdot g^{i-1})^{s_i}$. Send (w_i, y_i) to $\mathcal{I}$.
3. $\mathcal{I}$ computes $w_\sigma^b (= v_\sigma)$ and recovers $g^{\mu_\sigma} \leftarrow y_\sigma / h^{w_\sigma^b}$.

We denote this version of Naor-Pinkas protocol, where y_i is defined as $y_i = C_K(\mu_i, v_i)$, by $\binom{1}{n}\text{-}\mathsf{OT}_K(\mu; \sigma)$. As the end of this protocol, the verifier obtains commitments of all elements μ_i. Thus, the sender can argue in zero-knowledge for all $i \in [1, n]$ that the values μ_i satisfy some required conditions. We call such an OT protocol *verifiable.* (See [Lip03b] for a more precise definition.) $\mathcal{I}$ can "decrypt" y_σ with the "key" v_σ, given that the possible message space M is small enough for the exhaustive search on the set $\{g^x : x \in M\}$ to be practical. In the case of dichotomous RRT, $M = \{0, 1\}$.

We define the sender privacy of an oblivious transfer protocol as follows. The chooser $\mathcal{I}^*$ chooses σ and two different vectors, $\mu[1] = (\mu[1]_1, \ldots, \mu[1]_n) \in M^n$ and $\mu[2] = (\mu[1]_1, \ldots, \mu[1]_n) \in M^n$, such that $\mu[1]_\sigma = \mu[2]_\sigma$. Denote an $\mathcal{I}^*$ that has

made such choices by $\mathcal{I}^*(\mu[1], \mu[2])$. He submits both tuples to the responder, who flips a fair coin $b \leftarrow_R [1, 2]$. After that, the chooser and the responder execute the protocol $\binom{1}{n}\text{-}\mathsf{OT}_K(\mu[b]; \sigma)$. After receiving $\mu[b]_\sigma$, $\mathcal{I}^*$ guesses the value of b. Let $\mathsf{Adv}_k^{\mathsf{lor}}(\mathcal{I}^*, \mathcal{R})$ be the probability that $\mathcal{I}^*$ guesses the correct b, where probability is taken over the internal coin tosses of $\mathcal{I}^*$ and $\mathcal{R}$. We say that the oblivious transfer protocol is *ε-sender-private,* if for any unbounded algorithm $\mathcal{I}^*$, $\mathsf{Adv}_k^{\mathsf{lor}}(\mathcal{I}^*, \mathcal{R}) \leq \varepsilon$.

Theorem 1. *Let $\binom{1}{n}\text{-}\mathsf{OT}_K(\cdot;\cdot)$ be the described oblivious transfer protocol. (a) If a malicious $\mathcal{R}^*$ can guess the value of σ with advantage ε, then he can solve the Decisional Diffie Hellman (DDH) problem with the same probability and in approximately the same time. (b) This protocol is $(m-d)(m-1)/q \leq m(m-1)/q$-sender-private, where $d := q \mod m$ and $m := |M|$.*

The security proof is omitted from this extended abstract due to the space constraints.

Zero-Knowledge Arguments. We will use zero-knowledge arguments (and not proofs) of knowledge in our protocol, since they are at the very least statistically hiding and computationally convincing. This property is important in a setting where a verifier must not be able to extract additional information even if he is given infinite time.

Our first protocol uses only two very standard statistical zero-knowledge arguments. The first one is an argument that a given value y_i (Pedersen-)commits to a Boolean value $\mu_i \in \{0, 1\}$. One can use standard disjunctive proofs for this. We denote the (possibly parallelized) argument that this holds for $i \in [1, n]$ by $\mathsf{AKEncBool}(y_1, \ldots, y_n)$. The second argument of knowledge, $\mathsf{AKLin}(y_1, \ldots, y_{n+1}; a, b)$, is an argument that the prover knows some set of values μ_i, for which y_i is a commitment of μ_i, and such that $\sum_{i \leq n} \mu_i + a\mu_{n+1} = b$. This argument of knowledge can be constructed from Pedersen's commitment scheme by computing $y \leftarrow \prod_{i \leq n} y_i \cdot y_{n+1}^a$ and then arguing that the result y is a commitment to b. Note that such an argument of knowledge is secure only when accompanied by zero-knowledge arguments of knowledge of the values μ_i; for this purpose, we employ $\mathsf{AKEncBool}(y_1, \ldots, y_{n+1})$ as described above.

5 Security Definitions

Next, we will give the definition of a weakly and strongly secure cryptographic RRT (CRRT). The security definitions will be in accordance with the ones in secure two-party computation. We will also explain why these requirements are relevant in the case of CRRT.

Assume we have a concrete variant of RRT, like RRT-W or RRT-IQ. Let Φ_p be the function that implements the desired functionality. For example, in the case of RRT-W, $\Phi_{p_{\mathsf{ct}}}(x)$ is a randomized function that with probability p_{ct} returns x, and with probability $1 - p_{\mathsf{ct}}$ returns $1 - x$. The ideal-world CRRT protocol, has three parties, the interviewer $\mathcal{I}$, the respondent $\mathcal{R}$, and the trusted third party $\mathcal{T}$. $\mathcal{R}$ has her type, $t_{\mathcal{R}}$ as her private input, while $\mathcal{I}$ has no private input. Then, $\mathcal{R}$ communicates $t_{\mathcal{R}}$ to $\mathcal{T}$, who selects the value $r_{\mathcal{R}} \leftarrow \Phi_{p_{\mathsf{ct}}}(t_{\mathcal{R}})$ and sends $r_{\mathcal{R}}$ to $\mathcal{I}$. After that, the private output of $\mathcal{I}$ will be $\Phi_{p_{\mathsf{ct}}}(t_{\mathcal{R}})$, while $\mathcal{R}$ will have no private output. It is required that at the end of the protocol, the participants will have no information about the private inputs and

outputs of their partners, except for what can be deduced from their own private inputs and outputs. In particular, $\mathcal{I}$ (resp. $\mathcal{R}$) has no information about the value of $t_{\mathcal{R}}$ (resp. $r_{\mathcal{R}}$), except what they can deduce from their private inputs and outputs.

In an ideal world, exactly the next three types of attacks are possible: a party can (a) refuse to participate in the protocol; (b) substitute his private input to the trusted third party with a different value; or (c) abort the protocol prematurely. In our case, the attack (c) is irrelevant, since $\mathcal{R}$ has no output. (Attack (c) models the case when the first party halts the protocol after receiving his private output but before the second party has enough information to compute her output.) Therefore, in an ideal-world RRT protocol, we cannot protect against a participant, who (a) refuses to participate in polling *(non-participation attack)* or (b) claims that her type is $1 - t_{\mathcal{R}}$, where $t_{\mathcal{R}}$ is her real type *(absolute denial attack).* No other attacks should be possible. Note that neither (a) nor (b) is traditionally considered an attack in the context of polling or voting. The argument here is game-theoretic, and the solutions must be proposed by mechanism design, instead of cryptography: namely, a non-manipulable mechanism (e.g., the algorithm with which the election winner is determined from all the collected votes) must be designed so that answering against one's true type (or non-participation) would not give more beneficial results to the respondent than the truthful answer.

On the other hand, as we stated, no other attacks should be allowed. This requirement is very strict, so we will explain why it is necessary in the RRT's context. Clearly, one must protect the privacy of $\mathcal{R}$, since this is the primarily goal of a RRT. It is also necessary to protect the privacy of $\mathcal{I}$, although the reason here is more subtle. Namely, if $\mathcal{R}$ obtains any additional information about $r_{\mathcal{R}}$ before the end of the protocol (for example, if she suspects that $r_{\mathcal{R}} \neq t_{\mathcal{R}}$), she might halt the protocol. Such a behavior by a malicious respondent might cause a bias in the poll, as already explained. (Halting the protocol while having no information on $r_{\mathcal{R}}$ is equivalent to the non-participation attack.) The third requirement on the protocol, of course, is that $\mathcal{I}$ either halts or receives $\Phi_{p_{\mathrm{ct}}}(x)$, where x is the input submitted by the $\mathcal{R}$.

In a real-world implementation, we want to replace $\mathcal{T}$ by a cryptographic protocol $\Pi = (\mathcal{R}, \mathcal{I})$ between $\mathcal{R}$ and $\mathcal{I}$. This protocol $(\mathcal{R}, \mathcal{I})$ is assumed to be "indistinguishable" from the ideal-world protocol, that is, with a high probability, it should be secure against all attacks that do not involve attacks (a) or (b). "Secure" means that the privacy of $\mathcal{R}$ (resp. $\mathcal{I}$) must be protected, if $\mathcal{R}$ (resp. $\mathcal{I}$) follows the protocol, and that $\mathcal{I}$ either halts, or receives the value $\Phi_{p_{\mathrm{ct}}}(x)$, where x was the submitted value of $\mathcal{R}$. The security of the respondent should be information-theoretical, while the security of interviewer can be computational. That is, a secure CRRT-W protocol must have the next three properties (here, k is the security parameter):

Privacy of Respondent: Let $\mathcal{I}^*$ be an algorithm. After the end of the protocol execution $(\mathcal{R}, \mathcal{I}^*)$, $\mathcal{I}^*$ will have no more information on $t_{\mathcal{R}}$ than it would have had after the execution of the ideal world protocol. That is, assuming that $\mathsf{view}_{\mathcal{I}^*}$ is his view of the protocol $(\mathcal{R}, \mathcal{I}^*)$, define $\mathsf{Adv}_k^{\mathsf{pri-r}}(\mathcal{R}, \mathcal{I}^*) := |\Pr[\mathcal{I}^*(\mathsf{view}_{\mathcal{I}^*}, r_{\mathcal{R}}) = t_{\mathcal{R}}] - \Pr[t_{\mathcal{R}} | r_{\mathcal{R}}]|$, where the probability is taken over the internal coin tosses of $\mathcal{I}^*$ and $\mathcal{R}$. We say that a CRRT protocol is *privacy-preserving for the respondent,* if $\mathsf{Adv}_k^{\mathsf{pri-r}}(\mathcal{R}, \mathcal{I}^*)$ is negligible (in k) for any unbounded adversary $\mathcal{I}^*$.

Privacy of Interviewer: Let $\mathcal{R}^*$ be an algorithm. Assume that $\mathcal{I}$ halts when $\mathcal{R}^*$ halts. After the end of the protocol execution $(\mathcal{R}^*, \mathcal{I})$, $\mathcal{R}^*$ will have no more information on $t_{\mathcal{R}}$ than it would have had after the execution of the ideal world protocol. That is, assuming that $\mathsf{view}_{\mathcal{R}^*}$ is her view of the protocol $(\mathcal{I}, \mathcal{R}^*)$, define $\mathsf{Adv}_k^{\mathsf{pri-i}}(\mathcal{R}^*, \mathcal{I}) := |\Pr[\mathcal{R}^*(\mathsf{view}_{\mathcal{R}^*}, t_{\mathcal{R}}) = r_{\mathcal{R}}] - \Pr[\mathcal{R}^*(t_{\mathcal{R}}) = r_{\mathcal{R}}]|$, where the probability is taken over the internal coin tosses of $\mathcal{R}^*$ and $\mathcal{I}$. We say that a CRRT protocol is *privacy-preserving for the interviewer,* if for any adversary $\mathcal{R}^*$, if $\mathsf{Adv}_k^{\mathsf{pri-i}}(\mathcal{R}^*, \mathcal{I}) \leq \varepsilon$ and $\mathcal{R}^*$ takes τ steps of computation then $\varepsilon\tau$ is negligible (in k).

Correctness: Let $\mathcal{R}^*(x)$ be an algorithm with private input x to the protocol $(\mathcal{R}^*, \mathcal{I})$. Assume that $\mathcal{I}$ halts when $\mathcal{R}^*$ halts. We require that at the end of the protocol execution $(\mathcal{R}^*, \mathcal{I})$, $\mathcal{I}$ will either halt, or otherwise receive $\Phi_{p_{\mathsf{ct}}}(x)$ with high probability. That is, assuming that $\mathsf{view}_{\mathcal{I}}$ is $\mathcal{I}$'s view of the protocol $(\mathcal{R}^*, \mathcal{I})$, define $\mathsf{Adv}_k^{\mathsf{crct}}(\mathcal{R}^*, \mathcal{I}) := 1 - \Pr[\mathcal{I}(\mathsf{view}_{\mathcal{I}}) = \Phi_{p_{\mathsf{ct}}}(x) | \mathcal{I} \text{ does not halt}]$, where the probability is taken over the internal coin tosses of $\mathcal{I}$ and $\mathcal{R}^*$. We say that a CRRT protocol is *correct,* if for any adversary $\mathcal{R}^*$, if $\mathsf{Adv}_k^{\mathsf{crct}}(\mathcal{R}^*, \mathcal{I}) = \varepsilon$ and $\mathcal{R}^*$ takes up to τ steps of computation then $\varepsilon\tau$ is negligible (in k).

We call a cryptographic RRT (CRRT) protocol *weakly secure* if it is privacy-preserving for the respondent and correct. We call CRRT protocol *(strongly) secure* if it is weakly secure and it is privacy-preserving for the interviewer. While a secure CRRT protocol is preferable in many situations, there are settings where a weakly secure CRRT protocol suffices, such as where halting can be easily detected and punished, or means for state recovery prevent modifications between a first and second attempt of executing the protocol.

6 Cryptographic RRT

We will propose three different CRRT-W protocols. In the first two protocols, the common parameters are $p_{\mathsf{ct}} = \ell/n > 1/2$ for $\ell, n \in \mathbb{Z}$; generators g and h whose mutual discrete logs are unknown (at least by $\mathcal{R}$); and $K = (g; h)$. $\mathcal{R}$ has private input $t = t_{\mathcal{R}}$, and $\mathcal{I}$'s private output is $r_{\mathcal{R}}$.

CRRT Protocol Based on Oblivious Transfer. Our first implementation of RRT-W is described in Protocol 1. The arguments of knowledge can be efficiently constructed, see Sect. 4. Here, we can use $\mathsf{AKLin}(y_1, \ldots, y_{n+1}; 2\ell - n; \ell)$ since $\sum_{i \leq n} \mu_i + (2\ell - n)\mu_{n+1} = \ell$ independently of the value of t. All the steps in this protocol must be authenticated.

If we take the number of bits that must be committed as the efficiency measure (communication complexity of the protocol), then our protocol has complexity $O(n)$. In the polling application, one can most probably assume that $n \leq 5$. The security proofs of this protocol follow directly from the properties of underlying primitives. As a direct corollary from Theorem 1, we get that Protocol 1 is privacy-preserving for respondent ($\mathsf{Adv}_k^{\mathsf{pri-r}}(\mathcal{R}, \mathcal{I}^*) \leq 2/q + O(1/q)$, where the constant comes in from the use of statistically-hiding zero-knowledge arguments). It is privacy preserving for interviewer, given the Decisional Diffie-Hellman (DDH) assumption. The correctness of

PRECOMPUTATION STEP:

1. $\mathcal{R}$ prepares n random bits $\mu_i \in \{0,1\}$ for $i \in [1,n]$, such that $\sum \mu_i = \ell$ if $t = 1$ and $\sum \mu_i = n - \ell$ if $t = 0$. Additionally, she sets $\mu_{n+1} \leftarrow 1 - t$.
2. $\mathcal{I}$ chooses an index $\sigma \in [1,n]$.

INTERACTIVE STEP:

1. $\mathcal{I}$ and $\mathcal{R}$ follow $\binom{1}{n}$-$\mathsf{OT}_K(g^{\mu_1}, \ldots, g^{\mu_n}; \sigma)$. $\mathcal{I}$ obtains g^{μ_σ}, and computes μ_σ from that.
2. $\mathcal{R}$ performs zero-knowledge arguments $\mathsf{AKEncBool}(y_1, \ldots, y_{n+1})$ and $\mathsf{AKLin}(y_1, \ldots, y_{n+1}; 2\ell - n; \ell)$ with $\mathcal{I}$ as the verifier.
3. $\mathcal{I}$ halts if the verification fails.

Protocol 1: A secure CRRT-W protocol based on oblivious transfer

this protocol follows from the properties of the zero-knowledge arguments used under the DDH assumption.

In a simplified weakly secure protocol based on the same idea, $\mathcal{R}$ commits to all μ_i by computing and publishing $y_i \leftarrow \mathsf{C}_K(\mu_i; \rho_i)$. Next, $\mathcal{R}$ argues that $\mathsf{AKEncBool}(y_1, \ldots, y_{n+1})$, and $\mathsf{AKLin}(y_1, \ldots, y_{n+1}; 2\ell - n; \ell)$. After that, $\mathcal{I}$ sends σ to $\mathcal{R}$, who then reveals μ_σ and ρ_σ. Upon obtaining these, $\mathcal{I}$ verifies the correctness of the previous corresponding commitment, outputting μ_σ.

CRRT from Coin-Flipping. Protocol 2 depicts a secure CRRT-W protocol with communication complexity $\Theta(d \log_2 n)$, where $d := \lceil 1/(1 - p_{\mathsf{ct}}) \rceil$, and $p_{\mathsf{ct}} = \ell/n$ as previously. While in the common RRT application one can usually assume that n is relatively small, this second protocol is useful in some specific game-theoretic applications where for the best outcome, p_{ct} must have a very specific value. The idea behind this protocol is that at least one of the integers $\mu + \nu + i\ell \mod n$ must be in interval $[0, \ell - 1]$, and at least one of them must be in interval $[\ell, n-1]$. Hence, $\mathcal{I}$ gets necessary proofs for both the 0 and the 1 answer, which is sufficient for his goal. For his choice to be accepted, he must accompany the corresponding r with $\mathcal{R}$-s signature on his commitment on σ.

PRECOMPUTATION STEP:

1. $\mathcal{R}$ chooses a random $\mu \leftarrow_R [0, n-1]$.
2. $\mathcal{I}$ chooses random $\nu \leftarrow_R [0, n-1]$ and $\sigma \leftarrow_R [0, d-1]$.

INTERACTIVE STEP:

1. $\mathcal{R}$ commits to t and μ, and sends the commitments to $\mathcal{I}$.
2. $\mathcal{I}$ commits to σ, by setting $y \leftarrow \mathsf{C}_K(\sigma; \rho)$ for some random ρ. He sends ν and y to $\mathcal{R}$, together with a zero-knowledge argument that y is a commitment of some $i \in [0, d-1]$.
3. $\mathcal{R}$ verifies the argument. She computes values μ'_i, for $i \in [0, d-1]$, such that $\mu'_i = t \iff (\mu + \nu + i\ell \mod n) < \ell$. She signs y, and sends her signature together with $\{\mu'_i\}$ and the next zero-knowledge argument for every $i \in [0, d-1]$: $[\mu'_i = t \iff (\mu + \nu + i\ell \mod n) < \ell]$.
4. After that, $\mathcal{I}$ sets $r_{\mathcal{R}} \leftarrow \mu'_\sigma$. He will accompany this with $\mathcal{R}$-s signature on the commitment, so that both $\mathcal{R}$ and third parties can verify it.

Protocol 2: A secure CRRT-W protocol based on coin-flipping

PRECOMPUTATION STEP:

1. $\mathcal{I}$ chooses random $u_0 \leftarrow_R [0,1]$, $u_1 \leftarrow_R [0,1]$. He generates quantum states $|\psi_0\rangle = \sqrt{p_{\mathsf{ct}}}|u_0\rangle + \sqrt{1-p_{\mathsf{ct}}}|1-u_0\rangle$, $|\psi_1\rangle = \sqrt{p_{\mathsf{ct}}}|u_1\rangle + \sqrt{1-p_{\mathsf{ct}}}|1-u_1\rangle$.
2. $\mathcal{R}$ chooses a random $i \leftarrow_R [0,1]$.

INTERACTIVE STEP:

1. $\mathcal{I}$ sends $|\psi_0\rangle$ and $|\psi_1\rangle$ to $\mathcal{R}$.
2. $\mathcal{R}$ sends i to $\mathcal{I}$.
3. $\mathcal{I}$ sends u_i to $\mathcal{R}$.
4. $\mathcal{R}$ measures the state $|\psi_i\rangle$ in the basis $|\psi_{u_i}\rangle = \sqrt{p_{\mathsf{ct}}}|u_i\rangle + \sqrt{1-p_{\mathsf{ct}}}|1-u_i\rangle$, $|\psi_{u_i}^{\perp}\rangle = \sqrt{1-p_{\mathsf{ct}}}|u_i\rangle - \sqrt{p_{\mathsf{ct}}}|1-u_i\rangle$ and halts if the result is not $|\psi_{u_i}\rangle$.
5. If the verification is passed, $\mathcal{R}$ performs the transformation $|0\rangle \rightarrow |t\rangle$, $|1\rangle \rightarrow |1-t\rangle$ on the state $|\psi_{1-i}\rangle$ and sends it back to $\mathcal{I}$.
6. $\mathcal{I}$ measures the state in the basis $|0\rangle$, $|1\rangle$, gets outcome s. $\mathcal{I}$ outputs $r \leftarrow u_i \oplus s$.

Protocol 3: A quantum CRRT-W protocol.

A weakly secure version of this protocol is especially efficient. There, one should set $d \leftarrow 1$, and omit the steps in Protocol 2 that depend on σ being greater than 1. (E.g., there is no need to commit to σ anymore.) Thus, such a protocol would have communication complexity $\Theta(\log_2 n)$. Now, $p_{\mathsf{ct}} > 1/2$ (otherwise one could just do a bit-flip on the answers), and hence $d > 2$. On the other hand, the privacy of respondents is in danger if say $p_{\mathsf{ct}} \geq 3/4$. Thus, we may assume that $d \in [3,4]$. Therefore, Protocol 2 will be more communication-efficient than Protocol 1 as soon as $n/\log_2 n > 4 \geq d$, or $n \geq 16$. The weakly secure version will be *always* more communication-efficient.

This protocol is especially efficient if the used commitment scheme is an integer commitment scheme. In this case, to argue that $(\mu + \nu + i\ell \mod n) < \ell$ one only must do the next two simple steps: first, argue that $\mu + \nu + i\ell = z + en$ for some z, e, and then, argue that $z \in [0, \ell - 1]$. This can be done efficiently by using the range proofs from [Lip03a]. One can also use Pedersen's scheme, but this would result in more complicated arguments.

Quantum-Cryptographic RRT. The next *quantum CRRT protocol* (see Protocol 3) works also for irrational p_{ct}, and provides a relaxed form of information-theoretic security to *both* parties. While not secure by our previous definitions, it provides meaningfully low bounds on the probabilities of success for a cheater. Namely, (a) if dishonest, $\mathcal{R}$ cannot make his vote count as more than $\sqrt{2}$ votes: if $p_{\mathsf{ct}} = \frac{1}{2} + \varepsilon$, then $p_{\mathsf{adv}} \leq \frac{1}{2} + \sqrt{2}\varepsilon$ (The full version of this paper has a slightly better bound with a more complicated expression for p_{adv}). (b) if dishonest strategy allows $\mathcal{I}$ to learn t with probability $p_{\mathsf{ct}} + \varepsilon$, it also leads to $\mathcal{I}$ being caught cheating with probability at least $\frac{2p_{\mathsf{ct}}-1}{2}\varepsilon$. This form of security (information-theoretic security with relaxed definitions) is common for quantum protocols for tasks like bit commitment or coin flipping. The security guarantees of our quantum protocol compare quite well to ones achieved for those tasks. A desirable property of this quantum protocol is that it can be implemented by using contemporary technology, since it only involves transmitting and measuring single qubits, and no maintaining of coherent multi-qubit states.

To show the main ideas behind quantum protocol, we now show how to analyze a simplified version of protocol 3. The security proof for the full protocol is quite complicated and will be given in the full version of this paper.

The simplified version of Protocol 3 is: (1) $\mathcal{I}$ chooses a random $u \leftarrow_R [0, 1]$, prepares a quantum bit in the state $|\psi_u\rangle = \sqrt{p_{\mathsf{ct}}}|u\rangle + \sqrt{1-p_{\mathsf{ct}}}|1-u\rangle$ and sends it to $\mathcal{R}$; (2) $\mathcal{R}$ performs a bit flip if her type $t = 1$, and sends the quantum bit back to $\mathcal{I}$; (3) $\mathcal{I}$ measures the state in the computational basis $|0\rangle$, $|1\rangle$, gets answer s. The answer is $r = u \oplus s$. If both parties are honest, the state returned by respondent is unchanged: $\sqrt{p_{\mathsf{ct}}}|u\rangle + \sqrt{1-p_{\mathsf{ct}}}|1-u\rangle$ if $t = 0$ and $\sqrt{p_{\mathsf{ct}}}|1-u\rangle + \sqrt{1-p_{\mathsf{ct}}}|u\rangle$ if $t = 1$. Measuring this state gives the correct answer with probability $1 - p_{\mathsf{ct}}$. Next, we show that respondent is unable to misuse this protocol.

Theorem 2. *For any respondent's strategy $\mathcal{R}^*$, the probability of honest interviewer $\mathcal{I}$ getting $r = 1$ is between $1 - p_{\mathsf{ct}}$ and p_{ct}. Therefore, the previous protocol is both correct and privacy-preserving for the interviewer.*

Proof. We show that the probability of $r = 1$ is at most p_{ct}. The other direction is similar. We first modify the (simplified) protocol by making $\mathcal{R}^*$ to measure the state and send the measured result to $\mathcal{I}$, this does not change the result of the honest protocol since the measurement remains the same. Also, any cheating strategy for $\mathcal{R}^*$ in the original protocol can be used in the new protocol as well. So, it is sufficient to bound the probability of $r = 1$ in the new protocol. The answer is $r = 1$ if $\mathcal{I}$ sent $|\psi_i\rangle$ and $\mathcal{R}^*$ sends back j, with $i = j$. By a well-known fact, the maximum success probability with what one can distinguish two qubits is $1/2 + \sin\beta/2$, where β is the angle between two qubits. The rest is a calculation: to determine the angle β between $|\psi_0\rangle$ and $|\psi_1\rangle$, it suffices to determine the inner product which is $\sin\beta = 2\sqrt{p_{\mathsf{ct}}(1-p_{\mathsf{ct}})}$. Therefore, $\cos\beta = \sqrt{1-\sin^2\beta} = 2p_{\mathsf{ct}} - 1$ and $\frac{1}{2} + \frac{\cos\beta}{2} = p_{\mathsf{ct}}$. □

On the other hand, when using this simplified version, a dishonest interviewer $\mathcal{I}^*$ can always learn t with probability 1. Namely, it suffices to send the state $|0\rangle$. If $t = 0$, $\mathcal{R}$ sends $|0\rangle$ back unchanged. If $t = 1$, $\mathcal{R}$ applies a bit flip. The state becomes $|1\rangle$. $\mathcal{I}$ can then distinguish $|0\rangle$ from $|1\rangle$ with certainty by a measurement in the computational basis.

Note that this is similar to a classical "protocol", where $\mathcal{I}$ first generates a random u and sends a bit i that is equal to u with probability p_{ct} and $1-u$ with probability $1-p_{\mathsf{ct}}$. $\mathcal{R}$ then flips the bit if $t = 1$ and sends it back unchanged if $t = 0$. The interviewer XORs it with u, getting t with probability p_{ct} and $1 - t$ with probability $1 - p_{\mathsf{ct}}$. In this "protocol", $\mathcal{R}$ can never cheat. However, $\mathcal{I}^*$ can learn t with probability 1 by just remembering i and XORing the answer with i instead of u. In the classical world, this flaw is fatal because $\mathcal{I}$ cannot prove that he has generated i from the correct probability distribution and has not kept a copy of i for himself. In the quantum case, $\mathcal{I}$ can prove to $\mathcal{R}$ that he has correctly prepared the quantum state. Then, we get Protocol 3 with $\mathcal{I}$ sending two states $|\psi_{u_0}\rangle$ and $|\psi_{u_1}\rangle$, one of which is verified and the other is used for transmitting t. A detailed analysis of this protocol is omitted from this extended abstract.

7 Protocols for Other RRTs and Extensions

Protocol for Cryptographic RRT-IQ. Recall that in one version of RRT-IQ, the respondent would reply with his true opinion $t_{\mathcal{R}}$ with a rational probability $p_{\mathsf{ct}} = \ell/n$, while he would otherwise flip a coin and answer whether it came up tails. Like for CRRT-W, it is important to guarantee the use of correct distributions. Protocol 1 can be easily changed to work for this version of RRT-IQ. Instead of n random bits, $\mathcal{R}$ prepares $2n$ random bits μ_i, so that either $\sum_{i=1}^{n} \mu_i \in \{\ell, n-\ell\}$ and $\sum_{i=n+1}^{2n} \mu_i = n/2$ or $\sum_{i=n+1}^{2n} \mu_i \in \{\ell, n-\ell\}$ and $\sum_{i=1}^{n} \mu_i = n/2$. She then uses standard techniques to prove that the bits were prepared correctly, after which $\mathcal{I}$ chooses one of the $2n$ bits by using the verifiable oblivious transfer protocol. (Here, of course, n must be even.)

Protocol for Cryptographic PRRT-BD. The next protocol is a modification of Protocol 1 as well. Let p_i be such that $p_{\mathsf{ct}} + \sum_{i \in [1,m]} p_i = 1$, and assume that every respondent has a type $t_{\mathcal{R}} \in [1, m]$. Assume $p_{\mathsf{ct}} = \ell/n$, $p_i = \ell_i/n$ and that $p_i = 0$ if $i \notin [1, m]$. Assume $D \geq \max(\ell, \ell_1, \ldots, \ell_m) + 1$. The respondent prepares n numbers D^{μ_i}, such that $\sharp\{i : \mu_i = t_{\mathcal{R}}\} = \ell_{t_{\mathcal{R}}} + \ell$, and $\sharp\{i : \mu_i = j\} = \ell_j$, if $j \neq t_{\mathcal{R}}$. Then the interviewer and respondent will execute a variant of OT with choice σ, during which the interviewer only gets to know the value μ_σ. Then the respondent argues that the sum of all commitments is a commitment to the value $\sum \ell_i D^{\mu_i} + \ell D^j$, for some $j \in [1, m]$, by using range-proofs in exponents [LAN02]. (A more efficient proof methodology is available when D is a prime [LAN02], given that one uses an integer commitment scheme.) Additionally, she argues that every single commitment corresponds to a value D^i for $i \in [1, m]$, also using range-proofs of exponents [LAN02]. After the OT step, the interviewer gets g^{μ_σ}, and recovers μ_σ from it efficiently. (Note that $m \leq 10$ is typical in the context of polling.)

Extensions to Hierarchies of Interviewers. One can consider a hierarchy of interviewers, reporting to some central authority. If there is a trust relationship between these two types of parties, no changes to our protocol would be required. However, if the central authority would like to be able to avoid having to trust interviewers, the following modifications could be performed. First, each respondent would have to authenticate the transcript he generates, whether with a standard signature scheme, a group signature scheme, etc. Second, and in order to prevent collusions between interviewers and respondents, the interviewers must not be allowed to know the choice σ made in a particular interview. Thus, the triple *(A, B, C)* normally generated by the interviewer during the Naor-Pinkas OT protocol would instead have to be generated by the central authority, and kept secret by the same. More efficient versions of *proxy* OT satisfying our other requirements are beneficial for this application.

Full version. Due to the space constraints, we had to omit the security proof of the new verifiable oblivious transfer protocol and a detailed analysis of the quantum RRT protocol. The full version of this paper is available from the IACR eprint archive.

Acknowledgments. The third author was supported by the Finnish Defense Forces Research Institute of Technology and by the Finnish Academy of Sciences. We would like to thank Jouni K. Seppänen and Benny Pinkas for useful comments.

References

[BR99] Mihir Bellare and Ronald Rivest. Translucent Cryptography — An Alternative to Key Escrow, and Its Implementation via Fractional Oblivious Transfer. *Journal of Cryptology,* 12(2): 117–139, 1999.

[CK88] Claude Crépeau and Joe Kilian. Achieving Oblivious Transfer Using Weakened Security Assumptions (Extended Abstract). In *29th Annual Symposium on Foundations of Computer Science,* pages 42–52, White Plains, New York, USA, October24–26 1988. IEEE Computer Society Press.

[CM88] Arijit Chaudhuri and Rahul Mukerjee. *Randomized Response: Theory and Techniques,* volume 95 *of Statistics: Textbooks and Monographs.* Marcel Dekker, Inc., 1988. ISBN: 0824777859.

[Cré97] Claude Crépeau. Efficient Cryptographic Protocols Based on Noisy Channels. In Walter Fumy, editor, *Advances in Cryptology — EUROCRYPT '97,* volume 1233 of *Lecture Notes in Computer Science,* pages 306–317, Konstanz, Germany, 11–15 May 1997. Springer-Verlag.

[GASH69] Bernard G. Greenberg, Abdel-Latif A. Abul-Ela, Walt R. Simmons, and Daniel G. Horvitz. The Unrelated Question Randomized Response Model: Theoretical Framework. *Journal of the American Statistical Association,* 64(326):520–539, June 1969.

[KANG99] Hiroaki Kikuchi, Jin Akiyama, Gisaku Nakamura, and Howard Gobioff. Stochastic Voting Protocol To Protect Voters Privacy. In *1999 IEEE Workshop on Internet Applications,* pages 103–111, July 26–27 1999.

[Lai03] Chi Sung Laih, editor. *Advances on Cryptology—ASIACRYPT2003,* volume 2894 of *Lecture Notes in Computer Science,* Taipei, Taiwan, November 30–December 4 2003. Springer-Verlag.

[LAN02] Helger Lipmaa, N. Asokan, and Valtteri Niemi. Secure Vickrey Auctions without Threshold Trust. In Matt Blaze, editor, *Financial Cryptography — Sixth International Conference,* volume 2357 of *Lecture Notes in Computer Science,* pages 87–101, Southhampton Beach, Bermuda, March 11–14 2002. Springer-Verlag.

[Lip03a] Helger Lipmaa. On Diophantine Complexity and Statistical Zero-Knowledge Arguments. In Laih [Lai03], pages 398–415.

[Lip03b] Helger Lipmaa. Verifiable Homomorphic Oblivious Transfer and Private Equality Test. In Laih [Lai03], pages 416–433.

[NP01] Moni Naor and Benny Pinkas. Efficient Oblivious Transfer Protocols. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms,* pages 448–457, Washington, DC, USA, January 7–9 2001. ACM Press.

[Ped91] Torben P. Pedersen. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In J. Feigenbaum, editor, *Advances in Cryptology—CRYPTO '91,* volume 576 of *Lecture Notes in Computer Science,* pages 129–140, Santa Barbara, California, USA, August 11–15 1991. Springer-Verlag, 1992.

[War65] Stanley L. Warner. Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias. *Journal of the American Statistical Association,* 60(309):63–69, March 1965.

A Correct, Private, and Efficient Mix Network

Kun Peng, Colin Boyd, Ed Dawson, and Kapali Viswanathan

ISRC, IT Faculty, Queensland University of Technology
2 George Street, Brisbane, QLD 4001, Australia
{k.peng, c.boyd, e.dawson, k.viswanathan}@qut.edu.au
http://www.isrc.qut.edu.au

Abstract. A mix network achieving strong correctness and privacy is proposed. The degree of correctness and privacy are precisely stated and a formal proof of correctness is given. A grouping function is employed to achieve stronger correctness and higher efficiency without compromising strong privacy. In order to further improve the efficiency of the mix network a new batch verification technique, suitable for verifying multiple proofs of knowledge, is presented together with a formal proof of soundness.

Keywords: re-encryption mix network, shuffling, batch verification

1 Introduction

Mix networks are important tools to implement anonymity and are widely employed in many cryptographic applications such as e-voting and e-auctions. Since the original proposal of Chaum [6] many mix networks have been proposed in the research literature. However, most of them are inefficient, vulnerable, or limited to some special applications. Abe [1] introduced the idea of improving efficiency by dividing a costly large-scale verification operation into a few efficient small-scale verification operations. In this paper, we use Abe's idea in a new way to design a mix network with several novel features and avoiding some shortcomings of Abe's scheme. Our final proposal is simpler and more efficient than Abe's mix network, and also more efficient than other mix networks employing verification of shuffling on each server (e.g. [8, 14, 10]), especially when a large number of values are shuffled. Unlike other schemes, the new proposal achieves correctness and privacy more clearly and precisely. Therefore, our scheme is more suitable for many applications.

We divide the explanation of the new mix network into three stages. First a prototype Mix-1 is proposed, which employs a new verification mechanism to achieve formally proved correctness. Then Mix-1 is optimised to Mix-2 by adopting a grouping function. Compared to Mix-1, Mix-2 improves efficiency, strengthens correctness, and maintains strong privacy. Finally, a formally proved batch verification technique is applied to optimize Mix-2 to Mix-3, the final protocol achieving even higher efficiency.

F. Bao et al. (Eds.): PKC 2004, LNCS 2947, pp. 439-454, 2004.

The remainder of this paper is structured as follows. In section 2, previous work on mix networks is introduced. In section 3, a new method of correctness verification and a new batch verification technique are proposed. In section 4, the new mix network is presented. In section 5, security and other properties of the proposed mix network are analysed. Section 6 is a conclusion.

Parameter settings in the remainder of this paper are as follows.

- Let q and $p = 2q + 1$ be large primes. G is the cyclic subgroup of Z_p^* with order q. Let g and h be generators of G. ElGamal encryption algorithm is applied on G with private key $x \in Z_q$ and public key $(g, y = g^x)$. In this paper, when an ElGamal ciphertext (a, b) is presented for decryption, $a \in G$ and $b \in G$ are not checked. If $a \in Z_p^*$ and $b \in Z_p^*$, the ciphertext is decrypted and the decryption result is only guaranteed to be in Z_p^*.
- There are n users and m servers in the mix network. The number of honest servers is ϵ. If secret sharing is performed among the servers, the threshold is t (usually $m = 2t + 1$).

2 Related Work

A mix network shuffles a number of ciphertext inputs, each from one user, to the same number of plaintext outputs, so that 1) the outputs are a permutation of the plaintexts of the inputs; 2) the permutation between the inputs and the outputs is unknown, so that the users cannot be linked to their outputs. These two properties are called *correctness* and *privacy*. A mix network achieves *robustness* if it can still work properly in abnormal situations, such as failure of one or more switching nodes. A mix network is *publicly verifiable* if its correctness can be publicly verified. A mix network is usually composed of a few servers, working in sequence. Each server gets its inputs from the previous server and randomly permutes them to a set of outputs, which are inputs to the next server.

According to the processing performed by the servers, mix networks can be classified into two types: decryption chain mix networks and re-encryption mix networks. In the former type each input is sequentially encrypted for each server by the user. Consequently failure of any server means that the input message cannot be recovered if each server keeps his private key secret as required to achieve strong privacy. Therefore decryption chain mix networks inherently lack robustness. Only re-encryption mix networks are discussed further in this paper.

Ogata *et al.* [15], introduced a basic structure for re-encryption mix networks, which was further developed in many later papers. Suppose ElGamal encryption scheme is employed with private key x and public key $(g, y = g^x)$. Several decrypting authorities share x by t-out-of-m threshold verifiable secret sharing. The m servers SV_j for $j = 1, 2, \ldots, m$ form a mix network to shuffle n encrypted inputs c_i for $i = 1, 2, \ldots, n$. Inputs to SV_j are $c_{j-1,i}$ for $i = 1, 2, \ldots, n$ while $c_{0,i} = c_i$ for $i = 1, 2, \ldots, n$. Outputs of SV_j are $c_{j,i}$ for $i = 1, 2, \ldots, n$. On server SV_j, input $c_{j-1,i} = (a_{j-1,i}, b_{j-1,i})$ is permuted to $c_{j,\pi_j(i)} = (a_{j,\pi_j(i)}, b_{j,\pi_j(i)}) = (g^{r_{j,i}} a_{j-1,i}, y^{r_{j,i}} b_{j-1,i})$ where $r_{j,i}$ is randomly chosen and π_j is a secret random permutation of $\{1, 2, \ldots, n\}$. The outputs of the

mix network are $c'_i = c_{m,i}$ for $i = 1, 2, \ldots, n$. The shuffling from n inputs to n outputs on every server is denoted as $PN(n)$, correctness of which must be verified. Finally, the decrypting authorities (e.g. the servers themselves) cooperate to decrypt c'_i for $i = 1, 2, \ldots, n$.

Mix networks can be further classified into three categories according to the different correctness verification mechanisms.

- In the first category, correctness is not verified and the servers are trusted to perform the shuffling correctly. Ohkubo and Abe [16] designed an example in this category. Strong trust is necessary in such a mix network.
- Mix networks in the second category do not provide a verification of correct shuffling by each server separately. Instead, correctness of the shuffling by the whole mix network is verified after the mix network outputs the shuffled results in plaintexts. Several published schemes fall into this category [6, 17, 19, 9]. Drawbacks of this category include 1) a cheating server cannot be identified instantly; 2) in case of verification of incorrect shuffling, a mix network in the third category must be employed to perform the shuffling again; 3) some outputs may be revealed in plaintext even when the shuffling is incorrect and a re-shuffling is needed.
- In the third category [18, 13, 1, 2, 8, 15, 12, 4, 14, 10] each server verifies correctness of the previous servers' shuffling before performing its own shuffling and proves that its own shuffling is correct before sending them to the next server. Although the schemes in the first two categories are more efficient, the third category is still very useful because
  1. it overcomes the shortcomings of the first two categories;
  2. it is a necessary sub-function (to deal with the abnormal situation when cheating in the shuffling is found) in the second category.

  However, in this category, various problems exist: [13] is not publicly verifiable; the guarantee for correctness and privacy is not strong enough for many applications [12, 4]; [1, 2, 15, 18] are inefficient. Among them, three recently proposed schemes [8, 14, 10] are best. However, these three schemes are still not efficient enough for large-scale applications (e.g. national voting) as their computational cost is linear to the number of inputs.

In the third category, Abe's scheme [1] has a particularly useful feature which is an efficiency improvement on the following naive mix network. Let $\pi_{j,l}$ for $l = 1, 2, \ldots, n!$ be all the $n!$ possible permutations for π_j. A naive method to verify correctness of shuffling by SV_j is to test the following equation.

$$\begin{aligned} & \log_g (a_{j,\pi_{j,1}(i)}/a_{j-1,i}) = \log_y (b_{j,\pi_{j,1}(i)}/b_{j-1,i}) \text{ for } i = 1, 2, \ldots n \\ \vee\ & \log_g (a_{j,\pi_{j,2}(i)}/a_{j-1,i}) = \log_y (b_{j,\pi_{j,2}(i)}/b_{j-1,i}) \text{ for } i = 1, 2, \ldots n \\ \ldots \vee\ & \log_g (a_{j,\pi_{j,n!}(i)}/a_{j-1,i}) = \log_y (b_{j,\pi_{j,n!}(i)}/b_{j-1,i}) \text{ for } i = 1, 2, \ldots n \end{aligned} \quad (1)$$

This verification allows correctness to be proved without breaching privacy. Zero knowledge proof of 1-out-of-$n!$ equality of logarithms can be applied to implement (1), based on the zero knowledge proof of partial knowledge by Cramer *et al* [7].

This test is very inefficient because the computational cost for both the prover and verifier on every server is $O(n \cdot n!)$ exponentiations. So Abe improved its efficiency by dividing a n-input-to-n-output mixing (denoted as $PN(n)$ in [1]) into a few 2-input-to-2-output mixing (denoted as $PN(2)$ in [1]). However, Abe's schemes are still not efficent enough for many applications. Our proposal is to design a re-encryption mix network employing correctness verification per server. The new scheme overcomes the shortcomings of Abe's schemes [1, 2], while retaining the idea that efficiency can be saved by dividing a large-scale correctness verification into several small-scale correctness verifications. It achieves higher computational efficiency than that of [8, 14, 10] in that the computational cost is independent of the number of users, but determined by the extent of correctness and privacy required by a certain application.

3 Preliminary Work

In this section we introduce the building blocks used to construct our mix network. We first propose a new method for shuffling verification in a mix network and prove that it is sufficient to guarantee validity of the shuffling. Then we present a new batch verification technology to improve efficiency of simultaneous proofs of equality of logarithms, which appear in the verification of the shuffling.

3.1 Improvement on the Naive Verification Technique

Although naive verification by Equation (1) can explicitly guarantee the correctness of SV_j's shuffling, it is too inefficient to be practical. A more efficient verification technique uses the following equation.

$$\begin{aligned}
&\log_g (a_{j,1}/a_{j-1,i}) = \log_y (b_{j,1}/b_{j-1,i}) \ \vee \\
&\log_g (a_{j,2}/a_{j-1,i}) = \log_y (b_{j,2}/b_{j-1,i}) \vee \ldots \vee \\
&\log_g (a_{j,n}/a_{j-1,i}) = \log_y (b_{j,n}/b_{j-1,i}) \text{ for } i = 1, 2, \ldots n.
\end{aligned} \tag{2}$$

Equation (2) must be proved with zero knowledge proof of 1-out-of-n equality of logarithms. The computational cost of proof and verification of this equation is $n(4n-2)$ and $4n^2$ exponentiations respectively. The zero knowledge proof of Equation (2) by SV_j is denoted by *CV (correctness verification)* in the rest of this paper.

It is proved in Theorem 1 that *CV* is enough for the correctness verification.

Definition 1 *$SV_j(c_{j-1,\mu}, c_{j,\nu}) = 1$ means SV_j knows $r_{j,\nu}$ satisfying $a_{j,\nu} = g^{r_{j,\nu}} a_{j-1,\mu}$ and $b_{j,\nu} = y^{r_{j,\nu}} b_{j-1,\mu}$.*

Theorem 1. *If the shuffling by SV_j is incorrect, CV can be satisfied with a probability no more than $1/q$ without collusion of all the previous $j-1$ servers and at least two users, assuming DL problem is intractable.*

To prove Theorem 1, the following lemma is used.

Lemma 1. *If the shuffling by SV_j is incorrect and for every $c_{j-1,\mu}$ with $1 \leq \mu \leq n$ there exists some $c_{j,\nu}$ with $1 \leq \nu \leq n$ such that $SV_j(c_{j-1,\mu}, c_{j,\nu}) = 1$, then SV_j knows $\log_g a_{j-1,i'} - \log_g a_{j-1,i''}$ where $1 \leq i' < i'' \leq n$.*

Proof: If the shuffling is incorrect and for every $c_{j-1,\mu}$ for $\mu = 1, 2, \ldots, n$, there exists a $c_{j,\nu}$ with $1 \leq \nu \leq n$ satisfying $SV_j(c_{j-1,\mu}, c_{j,\nu}) = 1$, then there must be two inputs $c_{j-1,\mu 1}$ and $c_{j-1,\mu 2}$ satisfying $SV_j(c_{j-1,\mu 1}, c_{j,\tau}) = 1$ and $SV_j(c_{j-1,\mu 2}, c_{j,\tau}) = 1$ with $1 \leq \tau \leq n$. Otherwise there exists a permutation PM between the inputs and outputs such that $c_{j,\nu} = PM(c_{j-1,\mu})$ if $SV_j(c_{j-1,\mu}, c_{j,\nu})$, which is contradictory to the assumption that the shuffling is incorrect.

$SV_j(c_{j-1,\mu 1}, c_{j,\tau}) = 1$ and $SV_j(c_{j-1,\mu 2}, c_{j,\tau}) = 1$ means SV_j knows λ_1 and λ_2, so that $a_{j,\tau} = g^{\lambda_1} a_{j-1,\mu 1}$, $b_{j,\tau} = y^{\lambda_1} b_{j-1,\mu 1}$, $a_{j,\tau} = g^{\lambda_2} a_{j-1,\mu 2}$ and $b_{j,\tau} = y^{\lambda_2} b_{j-1,\mu 2}$. □

Proof of Theorem 1: As SV_j cannot get collusion of all the previous $j-1$ servers and at least two users, the inputs to SV_j are encrypted randomly from the viewpoint of SV_j and SV_j knows $\log_g a_{j-1,i}$ for at most one $c_{j-1,i} = (a_{j-1,i}, b_{j-1,i})$ where $1 \leq i \leq n$ if *DL* problem is intractable. So, if the shuffling by SV_j is incorrect, there exists $c_{j-1,\mu}$, so that $SV_j(c_{j-1,\mu}, c_{j,\nu}) \neq 1$ for $\nu = 1, 2, \ldots, n$. Otherwise according to Lemma 1 SV_j knows $\log_g a_{j-1,i'} - \log_g a_{j-1,i''}$ where $1 \leq i' < i'' \leq n$, which is contradictory to the above assumption. So

$$\log_g (a_{j,1}/a_{j-1,\mu}) = \log_y (b_{j,1}/b_{j-1,\mu}) \vee \log_g (a_{j,2}/a_{j-1,\mu}) = \log_y (b_{j,2}/b_{j-1,\mu}) \vee \ldots \vee \log_g (a_{j,n}/a_{j-1,\mu}) = \log_y (b_{j,n}/b_{j-1,i})$$

can be proved in *CV* with a probability no more than $1/q$ as proof of equality of logarithms in *CV* implies knowledge of logarithm (without knowledge of the logarithm, SV_j can only guess the challenge and the success probability of the guess is $1/q$).

Therefore, *CV* can be satisfied with a probability no more than $1/q$. □

Even when SV_j colludes with all previous $j-1$ servers and at least two users, invalid shuffling of the honest users' inputs will still be discovered in *CV* with an overwhelmingly large probability. This conclusion is straightforward from the proof of Lemma 1. In proof of Lemma 1, it is illustrated that the only possible attack against correctness is for a malicious server to collude with two or more malicious users and all the previous servers to tamper any of these malicious users' inputs. Since an honest user will not conspire with the malicious server and will conceal the randomising factor in his encrypted input, the attack against the integrity of his input can only succeed with a negligible probability if DL is intractable. Due to space limitations, this conclusion is not proved in detail.

3.2 Batch Verification of Equality of Logarithms

A theorem for batch verification is presented in this section, which extends known batch techniques [3, 5, 11]. This technique can batch verify equality of logarithms and optimize efficiency of the verification protocol in Section 3.1. Batch verification of equality of logarithms was first mentioned in a voting scheme [18]. However, in [18], batch verification is not formally proposed or proved to be secure.

The formal description of batch verification of equality of logarithms is provided in Theorem 2, which will be formally proved.

Definition 2 $|\ |$ *is the absolute-value function from* Z_p^* *to* G *defined by*

$$|\sigma| = \begin{cases} \sigma & \text{if } \sigma \in G \\ -\sigma & \text{if } \sigma \in Z_p^* \setminus G \end{cases}$$

Theorem 2. *Suppose* $y_i \in Z_p^*$ *and* $z_i \in Z_p^*$ *for* $i = 1, 2, \ldots, n$. *Let* l *be a security parameter and* t_i *satisfying* $t_i < 2^l < q$ *for* $i = 1, 2, \ldots, n$ *be random values. If there exists* v, *such that* $1 \le v \le n$ *and* $\log_g |y_v| \neq \log_h |z_v|$, *then* $\log_g \prod_{i=1}^n y_i^{t_i} \neq log_h \prod_{i=1}^n z_i^{t_i}$ *with a probability no less than* $1 - 2^{-l}$.

To prove Theorem 2, a lemma is proved first.

Lemma 2. *Suppose* $y_i \in Z_p^*$ *and* $z_i \in Z_p^*$ *for* $i = 1, 2, \ldots, n$ *and* $t_1, t_2, \ldots, t_{v-1}, t_{v+1}, t_{v+2}, \ldots, t_n$ *are constant. If* $\log_g |y_v| \neq \log_h |z_v|$ *with* $1 \le v \le n$ *and* $\log_g \prod_{i=1}^n y_i^{t_i} = log_h \prod_{i=1}^n z_i^{t_i}$, *then there is only one possible solution for* t_v.

Proof: If the lemma is not correct, the following two equations are satisfied simultaneously where $\log_g |y_v| \neq \log_h |z_v|$, $t_1, t_2, \ldots$ and $t_v \neq \hat{t}_v$.

$$log_g \prod_{i=1}^{n} y_i^{t_i} = \log_h \prod_{i=1}^{n} z_i^{t_i} \tag{3}$$

$$\log_g \prod_{i=1}^{v-1} y_i^{t_i} \cdot y_v^{\hat{t}_v} \prod_{i=v+1}^{n} y_i^{t_i} = \log_h \prod_{i=1}^{v-1} z_i^{t_i} \cdot z_v^{\hat{t}_v} \prod_{i=v+1}^{n} z_i^{t_i} \tag{4}$$

Without losing generality, suppose $t_v > \hat{t}_v$. (3)−(4):

$$\log_g y_v^{t_v - \hat{t}_v} = \log_h z_v^{t_v - \hat{t}_v}$$

As y_v and z_v are members of Z_p^*, there are two possibilities.

1. y_v and z_v are members of G. Then $(t_v - \hat{t}_v) \log_g y_v = (t_v - \hat{t}_v) \log_h z_v \bmod q$. Note that $t_v - \hat{t}_v \neq 0 \bmod q$ because $1 \le \hat{t}_v < t_v < 2^l < q$. Therefore, $\log_g y_v = \log_h z_v \bmod q$

2. y_v or $z_v \in Z_p^* \backslash G$. Then $t_v - \hat{t}_v$ must have a factor 2 and $\frac{t_v - \hat{t}_v}{2} \log_g y_v^2 = \frac{t_v - \hat{t}_v}{2} \log_h z_v^2 \bmod q$. Note that $\frac{t_v - \hat{t}_v}{2} \neq 0 \bmod q$ because $1 \leq \hat{t}_v < t_v < 2^l < q$. Therefore, $\log_g y_v^2 = \log_h z_v^2 \bmod q$. Namely $\log_g |y_v| = \log_h |z_v| \bmod q$.

In both cases, $\log_g |y_v| = \log_h |z_v|$. That is contradictory to the assumption $\log_g |y_v| \neq \log_h |z_v|$. □

Proof of Theorem 2: Lemma 2 means that among the $(2^l)^n$ possible combinations of t_i for $i = 1, 2, \ldots, n$, at most $(2^l)^{n-1}$ of them can satisfy $\log_g \prod_{i=1}^n y_i^{t_i} = log_h \prod_{i=1}^n z_i^{t_i}$ when $\log_g |y_v| \neq \log_h |z_v|$. So if $\log_g |y_v| \neq \log_h |z_v|$ and t_i for $i = 1, 2, \ldots, n$ are randomly chosen, $\log_g \prod_{i=1}^n y_i^{t_i} = log_h \prod_{i=1}^n z_i^{t_i}$ is satisfied with probability no more than 2^{-l}. □

4 The Proposed Mix Network

When the server SV_j performs ElGamal re-encryption and permutation π_j and Equation (2) is employed to verify the correctness of shuffling, the following properties are achieved.

1. A dishonest server SV_j can prove its incorrect shuffling to be correct with probability no more than $1/q$ without collusion of all the previous $j-1$ servers and at least two users. Even when SV_j colludes with all the previous $j-1$ servers and at least two users, invalid shuffling of honest users' inputs will still be discovered in CV with an overwhelmingly large probability.
2. Identified incorrect shuffling can be removed and the mix network can recover efficiently.
3. Computational costs for the prover and verifier of the correctness verification of a server's shuffling are $n(4n-2)$ and $4n^2$ exponentiations respectively.
4. If at least one server is honest, all the $n!$ permutation are equally possible in the mix network and if the number of malicious decrypting authorities is no more than t, privacy is achieved.

This mix network is denoted as Mix-1. However there are still some drawbacks of this solution:

- when two users conspire with the first server, correctness is not guaranteed;
- when n is large, $O(n^2)$ exponentiations is still a high cost.

To solve these problems, an idea of Abe[1, 2] is used: divide a $PN(n)$ into a few smaller shufflings, verification of whose correctness is efficient. However, switching gate $PN(2)$ is not applied in this paper to avoid complex construction of gate circuit. Instead, a simpler grouping technique is used.

4.1 Group Shuffling

On each server the n inputs are divided into groups with same size k, while re-encryption and random permutation are applied to each group. For simplicity,

suppose $n = k^u$. There are $z = k^{u-1}$ groups. Usually $m \leq u$ as the number of servers is often small. The grouping function on every server is specially designed according to a general rule: if an input to the mix network is likely to be permuted to a few outputs after the shuffling of the first j servers, any two of these outputs (inputs to the $j+1^{th}$ server) cannot be divided into a same group on the $j+1^{th}$ server. This rule can provide the greatest diffusion, and thus as strong privacy as possible.

Before the shuffling, each server SV_j randomly generates $v_{j,i} \in G$ for $i = 1, 2, \ldots, n$. Inputs to the mix network c_i for $i = 1, 2, \ldots, n$ are sorted to $c_{0,i} = (a_{0,i}, b_{0,i})$ for $i = 1, 2, \ldots, n$, so that $a_{0,i} + \sum_{j=1}^{m} v_{j,i} \bmod p$ increases as i increases. On server SV_j, the shuffling is as follows.

1. Grouping
   - SV_j get inputs $c_{j-1,i}$ for $i = 1, 2, \ldots, n$ from SV_{j-1}. So far $c_{0,k^{j-1}w+1}$, $c_{0,k^{j-1}w+2}$, $\ldots c_{0,k^{j-1}w+k^{j-1}}$ have been shuffled to $c_{j-1,k^{j-1}w+1}$, $c_{j-1,k^{j-1}w+2}$, $\ldots c_{j-1,k^{j-1}w+k^{j-1}}$ for $w = 0, 1, \ldots, k^{u-j+1} - 1$. Denote $c_{j-1,k^{j-1}w+1}$, $c_{j-1,k^{j-1}w+2}$, $\ldots c_{j-1,k^{j-1}w+k^{j-1}}$ as a shuffling range $R_{j-1,w+1}$, then SV_j in fact receives k^{u-j+1} shuffling ranges $R_{j-1,1}$, $R_{j-1,2}$, $\ldots$, $R_{j-1,k^{u-j+1}}$.
   - SV_j regroups in every k successive shuffling ranges. The k inputs in the same position in every k successive shuffling ranges are regrouped into the same group. Namely, input $c_{j-1,i}$ is mapped to $c_{j,\alpha,\beta}$, which is the β^{th} element in Group α, where $\alpha = ((i-1)/k^j)k^{j-1} + ((i-1) \bmod k^{j-1}) + 1$ and $\beta = ((i-1) \bmod k^j)/k^{j-1} + 1$.
2. Re-encryption and permutation
   $c_{j,\alpha,\beta} = (a_{j,\alpha,\beta}, b_{j,\alpha,\beta})$ is permuted to $c'_{j,\alpha,\pi_{j,\alpha}(\beta)} = (a'_{j,\alpha,\pi_{j,\alpha}(\beta)}, b'_{j,\alpha,\pi_{j,\alpha}(\beta)}) = (g^{r_{j,\alpha,\beta}} a_{j,\alpha,\beta}, y^{r_{j,\alpha,\beta}} b_{j,\alpha,\beta})$ for $\alpha = 1, 2, \ldots, z$ and $\beta = 1, 2, \ldots, k$ where $r_{j,\alpha,\beta}$ is randomly chosen and $\pi_{j,\alpha}$ for $\alpha = 1, 2, \ldots, z$ are random secret permutations from $\{1, 2, \ldots, k\}$ to $\{1, 2, \ldots, k\}$.
3. De-grouping
   $c_{j,i} = c'_{j,\alpha,\beta}$ where $i = k(\alpha - 1) + \beta$.

Shuffling of SV_j is verified by SV_{j+1} before it starts its own shuffling using the following equation.

$$\log_g (a'_{j,\alpha,1}/a_{j,\alpha,\beta}) = \log_y (b'_{j,\alpha,1}/b_{j,\alpha,\beta}) \vee \log_g (a'_{j,\alpha,2}/a_{j,\alpha,\beta}) = \log_y (b'_{j,\alpha,2}/b_{j,\alpha,\beta}) \vee \ldots \vee \log_g (a'_{j,\alpha,k}/a_{j,\alpha,\beta}) = \log_y (b'_{j,\alpha,k}/b_{j,\alpha,\beta}) \quad \text{for } \alpha = 1, 2, \ldots, z \text{ and } \beta = 1, 2, \ldots, k \tag{5}$$

Realization of verification of Equation (5) is denoted as *GCV (grouped correctness verification)*. If the verification fails, SV_{j+1} gets the outputs of SV_{j-1}, verifies them and uses them as its inputs if they are valid. If SV_{j-1}'s outputs are invalid too, he gets the outputs of the previous server until he finds a set of valid outputs as its inputs. After the shuffling of the last server, the outputs are decrypted as in Mix-1. This mix network applying group shuffling is denoted as Mix-2.

The following theorem can be proved in a way similar to the proof of theorem 1.

Theorem 3. *If the group shuffling by SV_j is incorrect, GCV can be satisfied with a probability no more than $1/q$ without collusion of all the previous $j-1$ servers and at least two users in a same group on SV_j, assuming DL problem is intractable.*

When conspiracy of all the previous servers and at least two malicious users is available, attack against correctness is more difficult than in Mix-1. As the grouping function is dependent on $v_{j,i}$ for $j = 1, 2, \ldots m$ and $i = 1, 2, \ldots n$, if at least one server is honest to generate them randomly, the grouping on any server is random. So if only static attack (all colluding users and servers are chosen before the attack starts) is considered and at least one server SV_j is honest to choose $v_{j,i}$ for $i = 1, 2, \ldots n$ randomly, the probability that the colluding users are in the same group on any server is low. For example, even if SV_1 colludes with two users, they happen to fall in a same group with a probability $1/z$. That means although attacks involving more than one user and the first few servers against correctness is still possible, they succeed with a low probability[1]. Like in Mix-1, the probability to tamper with an honest user's input successfully is negligible if DL is intractable. Therefore, correctness property is improved.

The computational cost to produce the proof is $n(4k-2)$ exponentiations. The computational cost to verify the proof is $4nk$ exponentiations. Better efficiency is achieved compared to Mix-1.

Privacy of Mix-2 is achieved if the number of malicious decrypting authorities is no more than t. The extent of privacy is measured by two factors: diffusion of any single input and diffusion of the inputs as a whole. As stated before, in normal applications $m < u$. So, if a dishonest server reveals its shuffling, it makes no difference to the situation where this server performs re-encryption without permutation. Therefore, the only impact of this attack on the privacy of the shuffling of the whole mix network is to degrade the mix network to a mix network containing one fewer servers. The shuffling of the other servers is not affected and can still provide strong privacy protection.

- Diffusion of any single input: each input may be permuted to any of a set of k^ϵ outputs with an equal probability, where ϵ is the number of honest servers.
- Diffusion of the inputs as a whole: $(k!)^{z\epsilon}$ possible permutations from the inputs of the mix network to its outputs are equally likely.

If $m \geq u$, greater privacy is possible.

- When $\epsilon = u$, diffusion of single input may be as great as that in Mix-1 (any input to n equally likely outputs).
- When $\epsilon > u$, diffusion of the inputs as a whole may be as great as that in Mix-1 (all $n!$ possible permutations are equally likely).

However, it is only possible as it depends on the distribution of the honest servers.

[1] As k is usually small, z is large when n is large. So the probability is very low when n is very large as in a large-scale voting.

4.2 Batched Group-Shuffling Mix Network

Efficiency of correctness verification of Mix-2 is better compared to that of Mix-1. However it is still costly when n is large. The batch verification technique in Section 3.2 can be employed to improve the efficiency further. If every server SV_j uses a same permutation π_j to replace $\pi_{j,\alpha}$ for $\alpha = 1, 2, \ldots, z$, according to Theorem 2 verification equation (5) can be batched as follows.

$$\begin{aligned}
&\log_g(\prod_{\alpha=1}^{z} a'^{t_{j,\alpha}}_{j,\alpha,1} / \prod_{\alpha=1}^{z} a^{t_{j,\alpha}}_{j,\alpha,\beta}) = \log_y(\prod_{\alpha=1}^{z} b'^{t_{j,\alpha}}_{j,\alpha,1} / \prod_{\alpha=1}^{z} b^{t_{j,\alpha}}_{j,\alpha,\beta}) \\
\vee\ &\log_g(\prod_{\alpha=1}^{z} a'^{t_{j,\alpha}}_{j,\alpha,2} / \prod_{\alpha=1}^{z} a^{t_{j,\alpha}}_{j,\alpha,\beta}) = \log_y(\prod_{\alpha=1}^{z} b'^{t_{j,\alpha}}_{j,\alpha,2} / \prod_{\alpha=1}^{z} b^{t_{j,\alpha}}_{j,\alpha,\beta}) \qquad (6) \\
\vee\ \ldots\ \vee\ &\log_g(\prod_{\alpha=1}^{z} a'^{t_{j,\alpha}}_{j,\alpha,k} / \prod_{\alpha=1}^{z} a^{t_{j,\alpha}}_{j,\alpha,\beta}) = \log_y(\prod_{\alpha=1}^{z} b'^{t_{j,\alpha}}_{j,\alpha,k} / \prod_{\alpha=1}^{z} b^{t_{j,\alpha}}_{j,\alpha,\beta}) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{for } \beta = 1, 2, \ldots, k
\end{aligned}$$

where $t_{j,\alpha}$ for $\alpha = 1, 2, \ldots, z$ are random integers with length l. The verification in Equation 6 for any β is denoted as $BGCV_{j,\beta}$. If $BGCV_{j,\beta}$ holds for $\beta = 1, 2, \ldots, k$, it is denoted as $BGCV(j-1 \rightarrow j)$, which means the correction verification for SV_j is passed. $BGCV(j-1 \rightarrow j)$ is checked for $j = 1, 2, \ldots, m$ to ensure the correctness of the mix network.

This mix network is denoted as Mix-3.

Definition 3 *In Mix-3, group shuffling by SV_j is correct if for any $1 \leq \alpha \leq z$, the same permutation exists between $|D(c_{j,\alpha,\beta})|$ for $\beta = 1, 2, \ldots, k$ and $|D(c'_{j,\alpha,\beta})|$ for $\beta = 1, 2, \ldots, k$ where D() denotes decryption.*

To apply equation (6), the construction of the mix network must be changed slightly as follows. After the shuffling of all the servers, the outputs of the mix network are decrypted. Every decrypted message M_i for $i = 1, 2, \ldots, n$ is checked to be in G by testing whether $M_i^q = 1$. If $M_i^q \neq 1$, an additional computation is performed: $M_i = -M_i = g_0^q M_i$.

5 Analysis

5.1 Correctness Analysis

Correctness of Mix-3 is proved in this subsection.

Definition 4 *Inputs of SV_j are divided into k vectors $V_\beta = (c_{j,1,\beta}, c_{j,2,\beta}, \ldots, c_{j,z,\beta})$ for $\beta = 1, 2, \ldots, k$ where $c_{j,\alpha,\beta} = (a_{j,\alpha,\beta}, b_{j,\alpha,\beta})$ is in $(Z_p^*)^2$. Outputs of SV_j are divided into k vectors $V'_\beta = (c'_{j,1,\beta}, c'_{j,2,\beta}, \ldots, c'_{j,z,\beta})$ for $\beta = 1, 2, \ldots, k$ where $c'_{j,\alpha,\beta} = (a'_{j,\alpha,\beta}, b'_{j,\alpha,\beta})$ is in $(Z_p^*)^2$.*

Definition 5 *$SV_j(V_\mu, V'_\nu) = 1$ means SV_j knows r_α satisfying $|a'_{j,\alpha,\nu}| = g^{r_\alpha}|a_{j,\alpha,\mu}|$ and $|b'_{j,\alpha,\nu}| = y^{r_\alpha}|b_{j,\alpha,\mu}|$ for $\alpha = 1, 2, \ldots, z$.*

Lemma 3. *If the shuffling by SV_j in Mix-3 is incorrect and for every V_μ with $1 \le \mu \le k$ there exists some V'_ν with $1 \le \nu \le k$ satisfying $SV_j(V_\mu, V'_\nu) = 1$, then SV_j knows $\log_g a_{j-1,\alpha,i'} - \log_g a_{j-1,\alpha,i''}$ for $\alpha = 1, 2, \ldots, z$ where $1 \le i' < i'' \le k$.*

Proof of Lemma 3 is very similar to that of Lemma 1, so is left to the reader.

Lemma 4. *$y_i \in Z_p^*$ for $i = 1, 2, \ldots, n$. $1 \le t_i < 2^l < p$ for $i = 1, 2, \ldots, n$ where t_i are random values and l is a security parameter. If there exists v, such that $1 \le v \le n$ and the logarithm $\log_g |y_v|$ is not known, then $\log_g \prod_{i=1}^n y_i^{t_i}$ is known only with a probability no more than 2^{-l}.*

Proof: First we prove a statement: if there exists v, such that $1 \le v \le n$ and $\log_g |y_v|$ is not known, given a definite set $S = \{t_i \mid t_i < 2^l$ and $i = 1, 2, \ldots v-1, v+1, \ldots n\}$, then $\log_g \prod_{i=1}^n y_i^{t_i}$ is known for at most one t_v.

If this statement is not correct, $\log_g (\prod_{i=1}^{v-1} y_i^{t_i} \cdot y_v^{t_v} \cdot \prod_{i=v+1}^n y_i^{t_i})$ and $\log_g (\prod_{i=1}^{v-1} y_i^{t_i} \cdot y_v^{\hat{t}_v} \cdot \prod_{i=v+1}^n y_i^{t_i})$ are known where $\log_g |y_v|$ is not known and $t_v \neq \hat{t}_v$.

So $\log_g (\prod_{i=1}^{v-1} y_i^{t_i} \cdot y_v^{t_v} \cdot \prod_{i=v+1}^n y_i^{t_i}) - \log_g (\prod_{i=1}^{v-1} y_i^{t_i} \cdot y_v^{\hat{t}_v} \cdot \prod_{i=v+1}^n y_i^{t_i})$
$= \log_g \frac{\prod_{i=1}^{v-1} y_i^{t_i} \cdot y_v^{t_v} \cdot \prod_{i=v+1}^n y_i^{t_i}}{\prod_{i=1}^{v-1} y_i^{t_i} \cdot y_v^{\hat{t}_v} \cdot \prod_{i=v+1}^n y_i^{t_i}} = \log_g y_v^{t_v - \hat{t}_v} = (t_v - \hat{t}_v) \log_g |y_v|$ is known.

Since $t_v - \hat{t}_v$ is public, $\log_g |y_v|$ is known. A contradiction is found, which means the statement is correct. So for every definite set $\{t_i \mid t_i < 2^l, i = 1, 2, \ldots, n\}$, the probability that t_v happens to be the unique possible value, so that $\log_g \prod_{i=1}^n y_i^{t_i}$ is known, is no more than 2^{-l} as there are 2^l choices for t_v. □

Theorem 4. *If the shuffling by SV_j is incorrect according to Definition 3, BGCV $(j-1 \to j)$ holds with a probability no more than $1 - (q-1)(1-2^{-l})/q$ without collusion of all the previous $j-1$ servers and at least $2z$ users with their re-encrypted inputs as $c_{j,\alpha,\rho}$ and $c_{j,\alpha,\delta}$ for $\alpha = 1, 2, \ldots, z$ where $1 \le \rho < \delta \le k$ if DL problem is intractable.*

Proof: The following denotations are used.
C denotes the shuffling is correct.
E_μ denotes $BGCV(j, \mu)$ holds.
Q denotes $BGCV(j-1 \to j)$ holds.
$N1_\mu$ denotes $D(c_{j,\alpha,\mu}) \neq D(c'_{j,\alpha,\nu})$ for $1 \le \nu \le k$.
$N2_\mu$ denotes $D(c_{j,\alpha,\mu}) = D(c'_{j,\alpha,\nu})$, but SV_j does not know $\log_g |a'_{j,\alpha,\nu}/a_{j,\alpha,\mu}|$ for $1 \le \nu \le k$.

As supposed, SV_j cannot get collusion of all the previous $j-1$ servers and at least $2z$ users with their re-encrypted inputs as $c_{j,\alpha,\rho}$ and $c_{j,\alpha,\delta}$ for $\alpha = 1, 2, \ldots, z$ where $1 \le \rho < \delta \le k$. So for any $\log_g a_{j,\alpha,\rho}$ and $\log_g a_{j,\alpha,\delta}$ for $\alpha = 1, 2, \ldots, z$ where $1 \le \rho < \delta \le k$, SV_j knows at most $2z-1$ of them and the left one is

independent of these $2z-1$ values in the viewpoint of SV_j if DL problem is intractable. According to Lemma 3, if the shuffling by server SV_j is incorrect and DL problem is intractable, there exists a vector V_μ and no V'_ν with $1 \le \nu \le k$ can satisfy $SV_j(V_\mu, V'_\nu) = 1$, where vector $V_\mu = (c_{j,1,\mu}, c_{j,2,\mu}, \ldots, c_{j-1,z,\mu})$ and $V'_\nu = (c'_{j,1,\nu}, c'_{j,2,\nu}, \ldots, c'_{j,z,\nu})$. Otherwise, SV_j knows $\log_g a_{j-1,\alpha,i'} - \log_g a_{j-1,\alpha,i''}$ for $\alpha = 1, 2, \ldots, z$ where $1 \le i' < i'' \le k$, which is contradictory to the fact that for any $\log_g a_{j,\alpha,\rho}$ and $\log_g a_{j,\alpha,\delta}$ for $\alpha = 1, 2, \ldots, z$ where $1 \le \rho < \delta \le k$ SV_j knows at most $2z-1$ of them and the left one is independent of those $2z-1$ values in the viewpoint of SV_j.

$SV_j(V_\mu, V'_\nu) \ne 1$ means there exists α, such that $1 \le \alpha \le k$ and $N1_\mu \vee N2_\mu$ is true. Namely, $P(N1_\mu/\bar{C}) + P(N2_\mu/\bar{C}) = 1$.

According to Theorem 2, $P(\bar{E}_\mu, N1_\mu) \ge 1 - 2^{-l} - (1-2^{-l})/q = (q-1)(1-2^{-l})/q$.

According to Lemma 4, when $N2_\mu$ happens, $\log_g \prod_{\alpha=1}^{z} (a_{j,\alpha}/a_{j-1,\alpha,\mu})^{t_\alpha}$ is known to SV_j with a probability no more than 2^{-l}. So $P(\bar{E}_\mu, N2_\mu) \ge 1 - 2^{-l} - (1-2^{-l})/q = (q-1)(1-2^{-l})/q$.

So, $P(\bar{E}_\mu/\bar{C}) = P(N1_\mu/\bar{C})P(\bar{E}_\mu/N1_\mu) + P(N2_\mu/\bar{C})P(\bar{E}_\mu/N2_\mu) = (q-1)(1-2^{-l})/q$.

Therefore, $P(\bar{Q}/\bar{C}) = P(\bar{E}_1/\bar{C}) \vee P(\bar{E}_2/\bar{C}) \ldots \vee P(\bar{E}_k/\bar{C}) \ge P(\bar{E}_\mu/\bar{C}) = (q-1)(1-2^{-l})/q$.
Namely $P(Q/\bar{C}) \le 1 - (q-1)(1-2^{-l})/q$. □

According to Theorem 4, Mix-3 can provide correctness on every server with an overwhelmingly large probability if DL problem is intractable and on a condition that this server cannot obtain the collusion of all the previous $j-1$ servers and users with at least 2 inputs in each group on two same positions. This condition is much weaker than the conditions for correctness in Mix-1 and Mix-2 as even though collusion of $2z$ or more users is available, the probability of their inputs are in each group on two same positions is very small if at least one server SV_j is honest to choose $v_{j,i}$ for $i = 1, 2, \ldots n$ randomly. Like in Mix-1 and Mix-2, the probability to tamper with an honest user's input is negligible. If the shuffling on every server is correct, the plaintexts in the inputs to the mix network $\{m_1, m_2, \ldots, m_n\}$ and its plaintext outputs $\{M_1, M_2, \ldots, M_n\}$ have a relationship $\{m_1, m_2, \ldots, m_n\} = \{|M_1|, |M_2|, \ldots, |M_n|\}$. If $M_i^q \ne 1$, an additional computation $M_i = M_i g_0^q$ is performed to obtain correct outputs. Therefore, stronger correctness is achieved in Mix-3 than in Mix-1 and Mix-2 as less trust on the users is needed in Mix-3.

5.2 Other Properties

Shuffling by every server can be verified publicly and efficiently and a cheating server can be identified immediately. Any identified cheating server is deleted and its inputs become inputs to the next server. So abnormal situations can be dealt with efficiently and the proposed scheme is robust.

Recall that as defined in Section 1 and Section 4 there are n users and m servers in the mix network; the number of honest servers is ϵ; t-out-of-m threshold

distributed decryption is used; k is the size of a group, z is the number of groups and $k^u = n$.

The computational cost for correctness proof and verification on a server in Mix-3 are $k(4k-2)$ and $4k^2$ exponentiations respectively. These costs are independent of the number of inputs and more efficient than those in Mix-2.

	Extent of Correctness	Diffusion of a single input	Diffusion of all the inputs	Is the diffusion uniform?
Abe[1]	not specified	1 among n if $\epsilon > t$	$n!$ permutations if $\epsilon > t$	No
Abe[2]	not specified	1 among n if $\epsilon > t$	$n!$ permutations if $\epsilon > t$	Yes
Furukawa[8]	not specified	1 among n	$n!$ permutations	Yes
Neff[14]	not specified	1 among n	$n!$ permutations	Yes
Groth[10]	not specified	1 among n	$n!$ permutations	Yes
Mix-1	$P(P/\bar{C}) \leq 1/q$	1 among n	$n!$ permutations	Yes
Mix-2	$P(P/\bar{C}) \leq 1/q$	1 among k^ϵ	$(k!)^{z\epsilon}$ permutations	Yes
Mix-3	$P(P/\bar{C}) \leq 1-(q-1)(1-2^{-l})/q$	1 among k^ϵ	$(k!)^\epsilon$ permutations	Yes

Table 1. Comparison of the mix networks

Privacy of Mix-3 is achieved if the number of malicious decrypting authorities is no more than t. Extent of privacy in Mix-3 is as follows when $m < u$ is assumed.

- Diffusion of any single input in Mix-3 is the same as that in Mix-2 (each input may be permuted to any of k^ϵ outputs with an equal probability).
- Diffusion of the inputs as a whole in Mix-3 is weaker: $(k!)^\epsilon$ possible permutation from the inputs of the mix network to its outputs are equally likely.

So, stronger correctness and higher efficiency in Mix-3 compared to in Mix-2 is achieved by sacrificing some privacy. By selecting appropriate k and m, a good trade-off between efficiency and privacy can be achieved. When $m \geq u$, as in the case of Mix-2, privacy may be improved in both factors if the distribution of honest servers is appropriate.

In Table 1 and Table 2, the proposed scheme is compared against the best mix networks in the third category (defined in Section 2). Note the following points

- "not specified" in Table 1 means the probability of correctness (with how much a probability the mix network is correct)is not provided.
- In [1] only $t+1$ out of the m servers take part in the shuffling.
- Re-encryption on each server cost $4(n \log_2 n - n + 1)$ exponentiations in [1] if ElGamal encryption is employed, while in other shuffling schemes this cost is usually $2n$. That is another aspect of inefficiency in [1].
- In [14], it was declared that the total computational cost of proof and verification of shuffling correctness is $8k+5$. However, the shuffling scheme in [14] is not concrete and it is commonly believed that Neff's scheme is not

so efficient as he claimed. Like Groth's analysis in [10], in this paper it is concluded that Neff's shuffling scheme costs ςn exponentiations (where ς is a small integer) and is not as efficient as [8] or [10].

In [1] only $t+1$ out of the m servers take part in the shuffling. The final version of the proposed scheme, Mix-3, achieves correctness more clearly (with a concrete extent) than the other schemes. Suppose in the proposed scheme, the decrypting authorities are chosen from the shuffling servers and the decryption key is shared among them with a t-out-of-m threshold like in most other mix networks. Then, when $\epsilon \leq t$, there is no privacy in either Abe's schemes [1, 2] or the proposed scheme as the inputs can be decrypted by $t+1$ malicious servers. When $\epsilon > t$, privacy in Mix-3 is sufficient for most applications although dependent on ϵ it may not achieve the maximum privacy as in [1]. The proposed scheme is more efficient than all the other schemes, especially when n is large. Moreover, the proposed scheme is simpler than Abe's schemes as complex gate circuit is not employed and the achieved properties are not dependent on theorems in gate circuit theory.

	Correctness proof on a server	Correctness verification on a server
Abe[1]	$12(n\log_2 n - n + 1)$	$16(n\log_2 n - n + 1)$
Furukawa[8]	$8n$	$10n$
Neff[14]	$o(n)$	$o(n)$
Groth[10][a]	$6n + 3n/\kappa + 3$	$6n + 3n/\kappa + 6$
Mix-3	$k(4k-2)$	$4k^2$

[a] κ is a parameter smaller than n.

Table 2. Comparison of computation cost in full-length exponentiations

In Table 3, an example is given to make a clearer comparison where $|q| = 1024, n = 10000, m = 5, t = 2, k = 10, \epsilon = 4 > t, \kappa = 100$ and SV_5 is assumed to be dishonest. Note that computational cost in Table 3 is in full-length exponentiations while some multiplications and short-length exponentiations are ignored as they are trivial compared to the costs of the full-length exponentiations . The results of this table clearly demonstrate enormous improvement on efficiency in the proposed scheme without losing strong correctness and privacy when there are a large number of users. In a national wide election involving millions of voters, the efficiency advantage of the proposed scheme is greater.

6 Conclusion

The proposed mix network provides strong and precise correctness and privacy. With the help of a grouping function and a batch verification technique, the mix network is very efficient. The mix network is robust and can deal with dishonest servers efficiently.

	Extent of Correctness	Diffusion of a single input	Diffusion of all the inputs	Cost of proof on a server	Cost of server verification
Abe[1]	not specified	1 among 10000	10000! permutations	1474537	1966050
Furukawa[8]	not specified	1 among 10000	10000! permutations	80000	100000
Groth[10]	not specified	1 among 10000	10000! permutations	60303	60306
Mix-3	$P(P/\bar{C})$ is extremely small	1 among 10000	1.734×10^{26} permutations	380	400

Table 3. Example for comparison

Acknowledgements

We acknowledge the support of the Australian government through ARC Discovery Grant No. DP0345458.

References

[1] M Abe. Mix-networks on permutation net-works. In *Asiacrypt 98,* pages 258–273, Berlin, 1999. Springer-Verlag. Lecture Notes in Computer Science 1716.

[2] Masayuki Abe and Fumitaka Hoshino. Remarks on mix-network based on permutation networks. In *Public Key Cryptography 2001,* pages 317–324, Berlin, 2001. Springer-Verlag. Lecture Notes in Computer Science 1992.

[3] M Bellare, J A Garay, and T Rabin. Fast batch verification for modular exponentiation and digital signatures. In *EuroCrypt'98,* pages 236–250, Berlin, 1998. Springer-Verlag. Lecture Notes in Computer Science 1403.

[4] Dan Boneh and Philippe Golle. Almost entirely correct mixing with applications to voting. In *Proceedings of the 9th ACM conference on Computer and communications security,* pages 68–77, 2002.

[5] Colin Boyd and Chris Pavlovski. Attacking and repairing batch verification schemes. In *Advances in Cryptology -ASIACRYPT 2000,* pages 58–71, Berlin, 2000. Springer-Verlag. Lecture Notes in Computer Science 1976.

[6] D Chaum. Untraceable electronic mail, return address and digital pseudonym. In *Communications of the ACM, 24(2),* pages 84–88, 1981.

[7] R. Cramer, I. B. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology - Crypto '94,* pages 174–187, Berlin, 1994. Springer-Verlag. Lecture Notes in Computer Science Volume 839.

[8] Jun Furukawa and Kazue Sako. An efficient scheme for proving a shuffle. In *CRYPTO2001,* volume 2139 of *Lecture Notes in Computer Science,* pages 368–387. Springer, 2001.

[9] Philippe Golle, Sheng Zhong, Dan Boneh, Markus Jakobsson, and Ari Juels. Optimistic mixing for exit-polls. In *ASIACRYPT 2002,* pages 451–465, Berlin, 2002. Springer-Verlag. Lecture Notes in Computer Science Volume 1592.

[10] Jens Groth. A verifiable secret shuffle of homomorphic encryptions. In *Public Key Cryptography 2003,* pages 145–160, Berlin, 2003. Springer-Verlag. Lecture Notes in Computer Science Volume 2567.

[11] Fumitaka Hoshino, Masayuki Abe, and Tetsutaro Kobayashi. Lenient/Strict batch verification in several groups. In *Information Security, 4th International Conference, ISC 2001,* pages 81–94, Berlin, 2001. Springer-Verlag. Lecture Notes in Computer Science Volume 2200.

[12] Markus Jakobsson, Ari Juels, and Ronald L. Rivest. Making mix nets robust for electronic voting by randomizsed partial checking. In *Proceedings of the 11th USENIX Security Symposium, San Francisco, CA, USA, August 5-9, 2002,* pages 339–353. USENIX, 2002.

[13] Ari Juels and Markus Jakobsson. An optimally robust hybrid mix network. In *Proc. of the 20th annual ACM Symposium on Principles of Distributed Computation,* pages 284–292. ACM, 2001.

[14] C. Andrew Neff. A verifiable secret shuffle and its application to e-voting. In *ACM Conference on Computer and Communications Security 2001,* pages 116–125, 2001.

[15] W Ogata, K Kurosawa, K Sako, and K Takatani. Fault tolerant anonymous channel. In *Proc. of International Conference on Information and Communication Security 1997,* pages 440–444, Berlin, 2000. Springer-Verlag. Lecture Notes in Computer Science Volume 1334.

[16] Miyako Ohkubo and Masayuki Abe. A length-invariant hybrid mix. In *ASIACRYPT 2000,* pages 178–191, Berlin, 2000. Springer-Verlag. Lecture Notes in Computer Science Volume 1976.

[17] C. Park, K. Itoh, and K. Kurosawa. Efficient anonymous channel and all/nothing election scheme. In *Advances in Cryptology - EuroCrypt '93,* pages 248–259, Berlin, 1993. Springer-Verlag. Lecture Notes in Computer Science Volume 765.

[18] K. Sako and J. Kilian. Receipt-free mix-type voting scheme–a practical solution to the implementation of a voting booth. In *EUROCRYPT '95,* pages 393–403, Berlin, 1995. Springer-Verlag. Lecture Notes in Computer Science Volume 921.

[19] Kapali Viswanathan, Colin Boyd, and Ed Dawson. A three phased schema for sealed bid auction system design. In *Information Security and Privacy, 5th Australasian Conference, ACISP'2000,* pages 412–426, Berlin, 2000. Springer-Verlag. Lecture Notes in Computer Science 1841.

Author Index