Immunizing Public Key Cryptosystems against Chosen Ciphertext Attacks[1]

Yuliang Zheng
Jennifer Seberry [2]

Centre for Computer Security Research
Department of Computer Science
University of Wollongong
Northfields Avenue, Wollongong, NSW 2522
AUSTRALIA
E-mail: {yuliang,jennie}@cs.uow.edu.au

21 September 1992

Abstract

This paper presents three methods for strengthening public key cryptosystems in such a way that they become secure against *adaptively* chosen ciphertext attacks. In an adaptively chosen ciphertext attack, an attacker can query the deciphering algorithm with any ciphertexts, *except* for the exact object ciphertext to be cryptanalyzed. The first strengthening method is based on the use of one-way hash functions, the second on the use of universal hash functions and the third on the use of digital signature schemes. Each method is illustrated by an example of a public key cryptosystem based on the intractability of computing discrete logarithms in finite fields. Security of the three example cryptosystems is formally proved. Two other issues, namely applications of the methods to public key cryptosystems based on other intractable problems and enhancement of information authentication capability to the cryptosystems, are also discussed.

1 Introduction

A considerable amount of research has been done in recent years, both from the theoretical [1, 2, 3, 4] and practical [5] points of view, in the pursuit of the construction of public key cryptosystems secure against chosen ciphertext attacks. In such an attack, the attacker (cryptanalyst) has access to the deciphering algorithm of a cryptosystem. The attacker can query the deciphering algorithm with any ciphertexts, obtain the

[1]Preprint 2/92. Also in *the Special Issue on Secure Communications, IEEE Journal on Selected Areas in Communications*, Vol.11, No.5, pp.715-724, 1993. Preliminary results were reported in "Practical approaches to attaining security against adaptively chosen ciphertext attacks (extended abstract)," Crypto'92, Santa Barbara, August 1992.

[2]Supported in part by the Australian Research Council under the reference numbers A49030136, A49130102 and A49131885.

matching plaintexts and use the attained knowledge in the cryptanalysis of an object ciphertext.

The theoretical results are appealing in that the schemes which embody them are provably secure under certain assumptions. However, most of these schemes are impractical due to the large expansion of the resulting ciphertext. The recent and notable schemes by Damgård overcome the problem of impracticality, but they are totally insecure against *adaptively* chosen ciphertext attacks in which an attacker has access to the deciphering algorithm even after he or she is given an object ciphertext to be cryptanalyzed. The attacker is allowed to query the deciphering algorithm with any ciphertext, *except* for the exact object ciphertext.

Adaptively chosen ciphertext attacks would impose serious problems on many services provided by modern information technology. To illustrate the possible attacks, consider the case of a security-enhanced electronic mail system where a public key cryptosystem is used to encipher messages passed among users. Nowadays it is common practice for an electronic mail user to include the original message he or she received into a reply to the message. For instance, a reply to a message may be as follows

```
  (original message)
> ......
> Hi, is Yum-Cha still on tonight ?
> ......

  (reply to the message)
......
Yes, it's still on.  I've already made the bookings.
......
```

This practice provides an avenue for chosen ciphertext attacks, as an attacker can send a ciphertext to a target user and expect the user to send back the corresponding plaintext as part of the reply. Now suppose that a user Alice is in the process of negotiating, through the electronic mail system, with two other users Bob and Cathy who are rivals of each other in a business. Let c be a ciphertext from Bob to Alice. Naturally, Cathy would like to know the contents of the communications between Alice and Bob. Cathy can obtain the ciphertext c by eavesdropping. However, it would be infeasible for her to extract its contents immediately. Instead, Cathy might try to discover *implicitly* the contents of c through discussions with Alice using the electronic mail. The problem facing Cathy is that she can not simply pass c to Alice with the hope that Alice would include the contents of c into her reply, as Alice would detect that c is actually a ciphertext created by Bob but not by Cathy. Nevertheless, if the cryptosystem is insecure against adaptively chosen ciphertext attacks, Cathy might still be able to obtain *indirectly* what she wants in the following way

1. Send Alice ciphertexts c_1, c_2, $\ldots$, c_n, none of which is the same as the object ciphertext c.

2. Receive the matching plaintext messages (hopefully) and

3. Extract the contents of c by the use of information obtained from the n plaintext-ciphertext pairs.

In this paper we present three pragmatic methods for immunizing public key cryptosystems against adaptively chosen ciphertext attacks. The first method is based on the use of one-way hash functions, the second on the use of universal hash functions and the third on the use of digital signature schemes. Each method is illustrated by an example of a public key cryptosystem based on the intractability of computing discrete logarithms in finite fields. Security of the three cryptosystems against adaptively chosen ciphertext attacks is formally proved under reasonable assumptions.

In Section 2, we summarize various types of possible attack to cryptosystems and introduce a formal definition for security of public key cryptosystems. In Section 3 previous proposals together with their problems are reviewed. Our immunization methods are illustrated in Section 4, by three public key cryptosystems based on the intractability of computing discrete logarithms in finite fields. This is followed by an analysis of security of the cryptosystems in Section 5. Section 6 is concerned with two other issues, namely applications of the immunization methods to public key cryptosystems based on other intractable problems, such as the problem of factoring large composite numbers, and the addition of information authentication capability to the three cryptosystems. Finally Section 7 presents some concluding remarks.

2 Notion and Notations

We will be concerned with the alphabet $\Sigma = \{0, 1\}$. The length of a string x over Σ is denoted by $|x|$, and the concatenation of two strings x and y is denoted by $x||y$. The bit-wise exclusive-or of two strings x and y of the same length is denoted by $x \oplus y$. The i-th bit of x is denoted by x_i and the substring of x from x_i to x_j, where $i \leqq j$, is denoted by $x_{[i \cdots j]}$. $\#S$ indicates the number of elements in a set S, and $x \in_R S$ means choosing randomly and uniformly an element x from the set S. The Cartesian product of two sets S and T is denoted by $S \times T$.

Denote by $\mathbb{N}$ the set of all positive integers, and by n a security parameter which determines the length of messages, the length of ciphertexts, the security of cryptosystems etc. As in the Diffie-Hellman/ElGamal's public key scheme [6, 7], p is an n-bit prime and g is a generator for the multiplicative group $GF(p)^*$ of the finite field $GF(p)$. Both p and g are public. To guarantee the security of cryptosystems based on the discrete logarithm problem, the length n of p should be large enough, preferably $n > 512$, and $p - 1$ should contain a large prime factor [8, 9]. Unless otherwise specified, all exponentiation operations appearing in the remaining part of this paper are assumed to be over the underlying groups.

Note that there is a natural one-to-one correspondence between strings in Σ^n and elements in the finite field $GF(2^n)$. Similarly, there is a natural one-to-one correspondence between strings in Σ^n and integers in $[0, 2^n - 1]$. Therefore, we will not distinguish among strings in Σ^n, elements in $GF(2^n)$ and integers in $[0, 2^n - 1]$.

A *public key cryptosystem*, invented by Diffie and Hellman [6], consists of three polynomial time algorithms (C, E, D). C is called a *key-generation algorithm* which, on input n, generates probabilistically a pair (pk, sk) of public and secret keys. Following the tradition in the field, when a security parameter n is used as input to an algorithm, it will be represented by the all-1 string of n bits which is denoted by 1^n. E is called an *enciphering algorithm* which, on input a public key pk and a plaintext message m, outputs a ciphertext c. Here m is chosen from a message space M_n. D is called a *deciphering algorithm* which, on input a secret key sk and a ciphertext c, outputs a message m or a special symbol $\emptyset$ meaning "no plaintext output". E and D satisfy the following unique decipherability condition, namely $D(sk, E(pk, m)) = m$.

2.1 Attacks to Cryptosystems

There are four common types of attack to a cryptosystem, namely *ciphertext only attacks*, *known plaintext attacks*, *chosen plaintext attacks* and *chosen ciphertext attacks* [10]. Related attacks against digital signatures are fully discussed in [11].

In a ciphertext only attack, which is the least severe among the four types of attack, an attacker is given an object ciphertext and tries to find the plaintext which is hidden in the object ciphertext.

In a known plaintext attack, an attacker has a collection of plaintext-ciphertext pairs besides an object ciphertext. The attacker may use the knowledge gained from the pairs of plaintexts and ciphertexts in the cryptanalysis of the object ciphertext.

In a chosen plaintext attack, an attacker has access to the enciphering algorithm. During the cryptanalysis of an object ciphertext, the attacker can choose whatever plaintexts he or she desires, feed the enciphering algorithm with the desired plaintexts and obtain the corresponding ciphertexts. Note that this type of attack is always applicable to a public key cryptosystem, since the attacker always has access to the public enciphering algorithm.

In a chosen ciphertext attack, which is the most severe among the four types of attack, an attacker has access to the deciphering algorithm. The attacker can query the deciphering algorithm with any ciphertexts and obtain the corresponding plaintexts. Then the attacker can use the knowledge obtained in the query and answer process to extract the plaintext of an object ciphertext.

Researchers further distinguish two forms of chosen ciphertext attack: *indifferently chosen ciphertext attacks* and *adaptively chosen ciphertext attacks*. An indifferently chosen ciphertext attack is also called a *lunchtime attack* or a *midnight attack* [2]. In such an attack the ciphertexts fed into the deciphering algorithm are chosen without being related to the object ciphertext. However the ciphertexts fed into the deciphering algorithm may be correlated with one another. This form of attack models the situation where the attacker has access to the deciphering algorithm *before* he or she is actually given the object ciphertext.

In adaptively chosen ciphertext attacks all ciphertexts fed into the deciphering algorithm can be correlated to the object ciphertext. This form of attack is more severe than the indifferently chosen ciphertext attacks and it models the situation where the attacker has access to the deciphering algorithm even *after* he or she is given the object ciphertext. The attacker is thus permitted to give the deciphering algorithm any available ciphertexts, *except for* the exact object ciphertext, and obtain the matching plaintexts. See the Introduction for a practical application where adaptively chosen ciphertext attacks would be a considerable threat.

2.2 Notion of Security

Much effort has been directed towards formalizing the notion of security of (public) key cryptosystems [12, 13, 14, 3]. To be called secure, a cryptosystem should fulfill at least the condition that it is infeasible for an attacker to obtain the complete plaintext of an object ciphertext. This requirement for the attacker can be weakened to that of obtaining just partial information of the plaintext. This intuition is well captured by the notion of *semantic security*, which can be viewed as the polynomially bounded version of Shannon's "perfect secrecy" [15]. Informally, a cryptosystem is semantically secure if whatever can be computed by an attacker about the plaintext given an object ciphertext, can also be computed without the object ciphertext. Semantic security ensures that no partial information on the plaintext is leaked from an object ciphertext to probabilistic polynomial time bounded attackers.

We can further classify semantic security into the following four kinds according to different types of attack. These four kinds of semantic security are (1) semantic security against ciphertext only attacks, (2) semantic security against known plaintext attacks, (3) semantic security against chosen plaintext attacks and (4) semantic security against chosen ciphertext attacks, respectively. As this paper is concerned with public key cryptosystems, we will restrict our attention to the later two kinds of semantic security, namely semantic security against chosen plaintext attacks and semantic security against chosen ciphertext attacks. In the following, a definition for semantic security of public key cryptosystems is given in terms of two probabilistic polynomial time Turing machines (algorithms): a collector and a partial information extractor. (See also [2].)

A collector is a probabilistic polynomial time algorithm $\mathcal{L}$ and it corresponds to the first stage of cryptanalysis in which an attacker gathers information useful for the next stage. The output of $\mathcal{L}$ is a string which can be the entire history of its computation. We are interested in the following three types of collectors:

1. *a chosen plaintext collector* $\mathcal{L}_{cp}$ which has as input only a security parameter n and a public key pk. Note that $\mathcal{L}_{cp}$ can always obtain plaintext-ciphertext pairs by the use of the public key pk.

2. *an indifferently chosen ciphertext collector* $\mathcal{L}_{icc}$ which, in addition to n and pk, has access to the deciphering algorithm. The collector can query the deciphering algorithm with polynomially many ciphertexts, obtain answers from the algorithm and use the information in its computation.

3. *an adaptively chosen ciphertext collector* $\mathcal{L}_{acc}$ which has as input n, pk and an object ciphertext. Like an indifferently chosen ciphertext collector $\mathcal{L}_{icc}$, an adaptively chosen ciphertext collector $\mathcal{L}_{acc}$ also has access to the deciphering algorithm. $\mathcal{L}_{acc}$ can query the deciphering algorithm with polynomially many ciphertexts, except for the exact object ciphertext. The ciphertexts given to the deciphering algorithm can be related to the object ciphertext to be cryptanalyzed.

A partial information extractor is a probabilistic polynomial time algorithm $\mathcal{T}$ which corresponds to the second stage of cryptanalysis in which an attacker actually computes information about the plaintext of an object ciphertext. $\mathcal{T}$ has n, pk and an object ciphertext as input, and has access to the output of a collector $\mathcal{L}$. The output of $\mathcal{T}$ is a string which may represent some partial information of the plaintext message obscured in the object ciphertext.

Definition 1 *Let (C, E, D) be a public key cryptosystem, $M_n = \Sigma^P$ a message space induced by a security parameter n, where P is a polynomial in n. Assume that a message m is drawn from M_n with a probability $p(m)$. Let V be any set and f_n^{pk} any function from M_n to V, where pk is a public key generated probabilistically by C on input n. Denote by $p_{f_n^{pk}}$ the maximum probability with which one could guess the output of the function f_n^{pk} without having any idea about its actual input. Note that $p_{f_n^{pk}} = \max_{v \in V}\{\Sigma_{m \in pre[f_n^{pk}(v)]} p(m)\}$, where $pre[f_n^{pk}(v)]$ denotes the set of the pre-images of v under f_n^{pk}. The public key cryptosystem (C, E, D) is semantically secure against chosen plaintext (indifferently and adaptively, respectively, chosen ciphertext) attacks if for any chosen plaintext (indifferently and adaptively, respectively, chosen ciphertext) collector $\mathcal{L}_{cp}$ ($\mathcal{L}_{icc}$ and $\mathcal{L}_{acc}$ respectively), for any partial information extractor $\mathcal{T}$, for any polynomial $Q = Q(n)$, for all sufficiently large n,*

$$\Pr\{\mathcal{T}(1^n, pk, c) = f_n^{pk}(m)\} < p_{f_n^{pk}} + 1/Q$$

where m is a message chosen from M_n with probability $p(m)$, pk a public key generated probabilistically by C on input n and c the ciphertext of m with respect to pk.

An equivalent notion of semantic security is that of *polynomial security*. A cryptosystem is polynomially secure if no probabilistic polynomial time algorithms can distinguish between the ciphertexts of two plaintext messages m_1 and m_2. We refer the reader to [13, 14, 2, 4] for a more detailed treatment of the notion of security for cryptosystems. A related notion called *non-malleable security* was introduced in [3], where an example of non-malleable public key cryptosystems was also demonstrated.

3 Problems with Previous Proposals

Rabin pioneered the research of constructing provably secure public key cryptosystems by designing a public key cryptosystem with the property that extracting the complete plaintext of an object ciphertext is computationally equivalent to factoring large numbers [16]. Goldwasser and Micali invented the first public key cryptosystem that hides all partial information [13]. The cryptosystem is a probabilistic one and it enciphers a plaintext in a bit-by-bit manner. A common drawback of these and many other cryptosystems is that, although secure against chosen plaintext attacks, they are easily compromised by chosen ciphertext attackers. On the other hand, much progress has been made in recent years in the construction of public key cryptosystems secure against chosen ciphertext attacks. We will review this development, and point out problems and weakness of the proposed schemes.

3.1 Theoretical Results

Theoretical study into the construction of public key cryptosystems secure against chosen ciphertext attacks was initiated by Blum, Feldman and Micali [1], who suggested the potential applicability of non-interactive zero-knowledge proofs to the subject. Naor and Yung carried further the study and gave the first concrete public key cryptosystem that is (semantically) secure against indifferently chosen ciphertext attacks [2]. Rackoff and Simon considered a more severe type of attack, namely adaptively chosen ciphertext attacks, and gave a concrete construction for public key cryptosystems withstanding the attacks [4]. In [3] Dolev, Dwork and Naor proposed a non-malleable (against chosen plaintext attacks) public key cryptosystem and proved that the cryptosystem is also secure against adaptively chosen ciphertext attacks.

All of these cryptosystems are provably secure under certain assumptions. However since they rely heavily on non-interactive zero-knowledge proofs, the resulting ciphertexts are in general much longer than original plaintexts. This disadvantage makes the cryptosystems highly impractical and difficult to realize in practice.

3.2 Damgård's Schemes

In [5], Damgård took a pragmatic approach to the subject. He proposed two simple public key cryptosystems that appear to be secure against indifferently chosen ciphertext attacks. The first is based on deterministic public key cryptosystems. Let (E_0, D_0) be the pair of enciphering and deciphering algorithms of a deterministic public key cryptosystem. Let (pk_1, sk_1) and (pk_2, sk_2) be two pairs of public and secret keys and h be an invertible one-to-one length-preserving function. The enciphering algorithm of Damgård's first cryptosystem operates in the following way:

$$\begin{aligned} E(pk_1, pk_2, m) &= (E_0(pk_1, r),\ E_0(pk_2, h(r)) \oplus m) \\ &= (c_1, c_2) \end{aligned}$$

where $m \in \Sigma^n$ is a plaintext message and $r \in_R \Sigma^n$ is a random string. The corresponding deciphering algorithm is as follows:

$$D(sk_1, pk_2, c_1, c_2) = E_0(pk_2, h(D_0(sk_1, c_1))) \oplus c_2$$

Damgård's second scheme is based on the Diffie-Hellman/ElGamal public key cryptosystem [6, 7], whose security relies on the intractability of computing discrete logarithms in finite fields. A user Alice's secret key is a pair (x_{A1}, x_{A2}) of elements chosen independently at random from $[1, p-1]$. Her public key is (y_{A1}, y_{A2}), where $y_{A1} = g^{x_{A1}}$ and $y_{A2} = g^{x_{A2}}$. When a user Bob wants to send an n-bit message m in secret to Alice, he sends her the following enciphered message

$$E(y_{A1}, y_{A2}, p, g, m) = (g^r,\ y_{A1}^r,\ y_{A2}^r \oplus m) = (c_1, c_2, c_3)$$

where $r \in_R [1, p-1]$. Note that here n is the length of the prime p. The deciphering algorithm for Alice, who possesses the secret key (x_{A1}, x_{A2}), is as follows

$$D(x_{A1}, x_{A2}, p, g, c_1, c_2, c_3) = \begin{cases} c_1^{x_{A2}} \oplus c_3 & \text{if } c_1^{x_{A1}} = c_2 \\ \emptyset & \text{otherwise} \end{cases}$$

Here $\emptyset$ is a special symbol meaning "no plaintext output".

Although Damgård's schemes are very simple and seem to be secure against indifferently chosen ciphertext attacks, they are *insecure* against adaptively chosen ciphertext attacks. Given an object ciphertext c ($c = (c_1, c_2)$ for the first scheme, and $c = (c_1, c_2, c_3)$ for the second scheme), an attacker can choose a random message m_r from Σ^n, calculate the bit-wise exclusive-or of m_r and the last part of the ciphertext c, and feed the deciphering algorithm with the modified ciphertext c'. The attacker will get $m' = m \oplus m_r$ as an answer, and obtain the desired message [3] m by computing $m' \oplus m_r$. Our cryptosystems to be described below share the same simplicity possessed by Damgård's cryptosystems, yet they attain a higher level of security, namely security against adaptively chosen ciphertext attacks.

4 Strengthening Public Key Cryptosystems

This section presents three simple methods for immunizing public key cryptosystems against chosen ciphertext attacks. The nature of the three immunization methods is the same — they all immunize a public key cryptosystem by appending to each ciphertext a tag that is correlated to the message to be enciphered. This is also the main technical difference between our proposals and Damgård's schemes. The three methods differ in the ways in which tags are generated. In the first method tags are generated by the use of a one-way hash function, in the second method by the use of a function chosen from a universal class of hash functions, and in the third method by the use of a digital signature scheme. The second immunization method is superior to the other two immunization methods in that no one-way hash functions are needed. This property is particularly attractive given the current state of research, whereby many one-way hash functions exist, few are efficient, and even fewer are provably secure.

We will illustrate our immunization methods with cryptosystems based on the Diffie-Hellman/ElGamal public key scheme. In Section 6, applications of the immunization methods to cryptosystems based on other intractable problems will be discussed. Denote by G the cryptographically strong pseudo-random string generator based on the difficulty of computing discrete logarithms in finite fields [17, 18, 19]. G stretches an n-bit input string into an output string whose length can be an arbitrary polynomial in n. This generator produces $O(\log n)$ bits output at each exponentiation. In the authors' opinion, for practical applications the generator could produce more than $\frac{3n}{4}$ bits at each exponentiation, without sacrificing security. Recently Micali and Schnorr discovered a very efficient pseudo-random string generator based on polynomials in the finite field $GF(p)$ (see Section 4 of [20]). The generator can

[3]One might argue that since at least half bits in the original ciphertext c remain untouched in the modified ciphertext c', adding a checking step to the deciphering algorithms would effectively thwart the attack. This countermeasure, however, does *not* work in general, as the deciphering algorithms may *not* know c. Even if the deciphering algorithms have a list of ciphertexts containing c, a more sophisticated attacker might still succeed in extracting m by generating c' in such a way that it passes the checking step.

produce, for example, $\frac{n}{2}$ bits with 1.25 multiplications in $GF(p)$. The efficiency of our cryptosystems to be described below can be further improved if Micali and Schnorr's pseudo-random string generator is employed.

A user Alice's secret key is an element x_A chosen randomly from $[1, p-1]$, and her public key is $y_A = g^{x_A}$. It is assumed that all messages to be enciphered are chosen from the set Σ^P, where $P = P(n)$ is an arbitrary polynomial with $P(n) \geqq n$. Padding can be applied to messages whose lengths are less than n bits. In addition, let $\ell = \ell(n)$ be a polynomial which specifies the length of tags. It is recommended that ℓ should be at least 64 for the sake of security.

4.1 Immunizing with One-Way Hash Functions

Assume that h is a one-way hash function compressing input strings into ℓ-bit output strings. A user Bob can use the following enciphering algorithm to send in secret a P-bit message m to Alice.

```
Algorithm 1 $E_{owh}(y_A, p, g, m)$
    1. $x \in_R [1, p-1]$.
    2. $z = G(y_A^x)_{[1 \cdots (P+\ell)]}$.
    3. $t = h(m)$.
    4. $c_1 = g^x$.
    5. $c_2 = z \oplus (m || t)$.
    6. output $(c_1, c_2)$.
end
```

The deciphering algorithm for Alice, who possesses the secret key x_A, is as follows:

```
Algorithm 2 $D_{owh}(x_A, p, g, c_1, c_2)$
    1. $z' = G(c_1^{x_A})_{[1 \cdots (P+\ell)]}$.
    2. $w = z' \oplus c_2$.
    3. $m' = w_{[1 \cdots P]}$.
    4. $t' = w_{[(P+1) \cdots (P+\ell)]}$.
    5. if $h(m') = t'$ then
         output $(m')$
       else
         output $(\emptyset)$.
end
```

When messages are of n bits, i.e. $P = n$, instead of the one-way hash function h the exponentiation function can be used to generate the tag t. In this case, the enciphering algorithm can be modified as follows: (a) Change the step 2 to "$z = G(y_A^x)_{[1 \cdots 2n]}$." (b) Change the step 3 to "$t = g^m$." The deciphering algorithm can be modified accordingly.

4.2 Immunizing with Universal Hash Functions

A class H of functions from Σ^P to Σ^ℓ is called a *(strongly) universal class of hash functions* [21, 22] mapping P-bit input into ℓ-bit output strings if for every $x_1 \neq x_2 \in \Sigma^P$ and every $y_1, y_2 \in \Sigma^\ell$, the number of functions in H taking x_1 to y_1 and x_2 to y_2 is $\#H/2^{2\ell}$. An equivalent definition is that when h is chosen uniformly at random from H, the concatenation of the two strings $h(x_1)$ and $h(x_2)$ is distributed randomly and uniformly over the Cartesian product $\Sigma^\ell \times \Sigma^\ell$. Wegman and Carter found a nice application of universal classes of hash functions to unconditionally secure authentication codes [22].

Now assume that H is a universal class of hash functions which map P-bit input into ℓ-bit output strings. Also assume that $Q = Q(n)$ is a polynomial and that each function in H is specified by a string of exactly Q bits. Denote by h_s the function in H that is specified by a string $s \in \Sigma^Q$. The enciphering algorithm for Bob who wants to send in secret a P-bit message m to Alice is the following:

```
Algorithm 3  $E_{uhf}(y_A, p, g, m)$
    1. $x \in_R [1, p-1]$.
    2. $r = y_A^x$.
    3. $z = G(r)_{[1 \cdots P]}$.
    4. $s = G(r)_{[(P+1) \cdots (P+Q)]}$.
    5. $c_1 = g^x$.
    6. $c_2 = h_s(m)$.
    7. $c_3 = z \oplus m$.
    8. output $(c_1, c_2, c_3)$.
end
```

The deciphering algorithm for Alice, who possesses the secret key x_A, is as follows:

```
Algorithm 4  $D_{uhf}(x_A, p, g, c_1, c_2, c_3)$
    1. $r' = c_1^{x_A}$.
    2. $z' = G(r')_{[1 \cdots P]}$.
    3. $s' = G(r')_{[(P+1) \cdots (P+Q)]}$.
    4. $m' = z' \oplus c_3$.
    5. if $h_{s'}(m') = c_2$ then
         output $(m')$
       else
         output $(\emptyset)$.
end
```

Note that the second part $c_2 = h_s(m)$ in the ciphertext can be obscured in the same way as Algorithm 1. This would improve practical security of the cryptosystem, at the expense of more computation time spent in generating pseudo-random bits.

The following is a simple universal class of hash functions which is originated from linear congruential generators in finite fields. (See also Propositions 7 and 8 of [21].) Let k be an integer. For $k+1$ elements $a_1, a_2, \ldots, a_k, b \in GF(2^\ell)$, let s be their concatenation, i.e., $s = a_1||a_2||\cdots||a_k||b$, and let h_s be the function defined by $h_s(x_1, x_2, \ldots, x_k) = \sum_{i=1}^{k} a_i x_i + b$ where $x_1, x_2, \ldots, x_k$ are variables in $GF(2^\ell)$. Then the collection H of the functions h_s defined by all $k+1$ elements from $GF(2^\ell)$ is a universal class of hash functions. Functions in H compress $k\ell$-bit input into ℓ-bit output strings. By padding to input strings, these functions can be applied to input strings whose lengths are not exactly $k\ell$. In particular, when $k = \lceil \frac{P}{\ell} \rceil$, they can be used to compress P-bit input into ℓ-bit output strings. In this case, a function in H can be specified by a string of $Q = P + (1+\alpha)\ell$ bits, where $0 \leqq \alpha = \frac{P \bmod \ell}{\ell} < 1$. This universal class of hash functions is particularly suited to the case where the length P of messages to be enciphered is much larger than the length ℓ of tags. We refer the reader to [22, 23] for other universal classes of hash functions.

4.3 Immunizing with Digital Signature Schemes

Assume that h is a one-way hash function compressing input strings into n-bit output strings. Also assume that Bob wants to send in secret a P-bit message m to Alice. The enciphering algorithm employed by Bob is the following:

```
Algorithm 5  $E_{sig}(y_A, p, g, m)$
        1. $x \in_R [1, p-1]$.
        2. $k \in_R [1, p-1]$ such that $\gcd(k, p-1) = 1$.
        3. $r = y_A^{x+k}$.
        4. $z = G(r)_{[1 \cdots P]}$.
        5. $c_1 = g^x$.
        6. $c_2 = g^k$.
        7. $c_3 = (h(m) - xr)/k \bmod (p-1)$.
        8. $c_4 = z \oplus m$.
        9. output $(c_1, c_2, c_3, c_4)$.
end
```

The corresponding deciphering algorithm for Alice, who possesses the secret key x_A, is as follows:

```
Algorithm 6  $D_{sig}(x_A, p, g, c_1, c_2, c_3, c_4)$
        1. $r' = (c_1 c_2)^{x_A}$.
        2. $z' = G(r')_{[1 \cdots P]}$.
        3. $m' = z' \oplus c_4$.
        4. if $g^{h(m')} = c_1^{r'} c_2^{c_3}$ then
              output $(m')$
           else
              output $(\emptyset)$.
end
```

Similar to the cryptosystem based on the use of universal hash functions described in Section 4.2, security of the cryptosystem can also be improved by hiding the third part $c_3 = (h(m) - xr)/k \bmod (p-1)$ with extra pseudo-random bits produced by the pseudo-random string generator G. In addition, when messages to be enciphered are of n bits, neither the one-way hash function h nor the pseudo-random string generator G is necessary. The enciphering algorithm for this case can be simplified by changing the step 4 of the above enciphering algorithm to "$z = r$." and the step 7 into "$c_3 = (m - xr)/k \bmod (p-1)$." The deciphering algorithm can be simplified accordingly.

The first three parts (c_1, c_2, c_3) of the ciphertext represents an adaptation of the ElGamal's digital signature. However, since everyone can generate these parts, they do *not* really form the digital signature of m. This immunization method was first proposed in [24], where other ways for generating the third part c_3 in the ciphertext were also suggested.

In Section 5, we will prove that the three cryptosystems are secure against adaptively chosen ciphertext attacks under reasonable assumptions. For convenience, we will denote by $\mathcal{C}_{owh}$ the first cryptosystem which applies one-way hash functions, by $\mathcal{C}_{uhf}$ the second cryptosystem which applies universal hash functions and by $\mathcal{C}_{sig}$ the third cryptosystem which applies the ElGamal digital signature.

5 Security of the Cryptosystems

This section is concerned with issues related to security of the three cryptosystems. First we discuss security of the cryptosystems against chosen plaintext attacks. We prove that both $\mathcal{C}_{owh}$ and $\mathcal{C}_{uhf}$ are secure against chosen plaintext attacks under the Diffie-Hellman Assumption to be defined below. Security of the cryptosystem $\mathcal{C}_{sig}$ is also discussed briefly. Then we introduce a notion called *sole-samplability*, and apply the notion in the proofs of security of the cryptosystems against chosen ciphertext attacks.

Security of our cryptosystems rely on the intractability of computing discrete logarithms in finite fields. More specifically, it relies on the *Diffie-Hellman Assumption* which can be informally stated as follows:

Assumption 1 *Given y_1, y_2, g and p, where $y_1 = g^{x_1}$ and $y_2 = g^{x_2}$ for some x_1 and x_2 chosen randomly and independently from $[1, p-1]$, it is computationally infeasible for any probabilistic polynomial time algorithm to compute $y = g^{x_1 x_2}$.*

Note that an algorithm for computing $y = g^{x_1(x_2+x_3)}$ from y_1, y_2, y_3, g and p, where y_1 and y_2 are the same as above, and $y_3 = g^{x_3}$ for some $x_3 \in [1, p-1]$, can be used to compute $y = g^{x_1 x_2}$ from y_1, y_2, g and p in the following way: In addition to y_1, y_2, g and p, the algorithm is also provided with $y_3 = g^{x_3}$, where x_3 is a known element chosen from $[1, p-1]$. Let the output of the algorithm be y. Then we have $g^{x_1 x_2} = y / y_1^{x_3}$. Therefore under the Diffie-Hellman Assumption, it is also infeasible to compute $y = g^{x_1(x_2+x_3)}$ from y_1, y_2, y_3, g and p.

5.1 Security against Chosen Plaintext Attacks

Let $x_1, x_2 \in_R [1, p-1]$, $y_1 = g^{x_1}$ and $y_2 = g^{x_2}$. Let z_1 be a P-bit string taken from the output of the pseudo-random string generator G on input $g^{x_1 x_2}$, and z_2 a truly random P-bit string. Then, by an argument similar to that for semantic security of a public key cryptosystem [25] based on the intractability of factoring large composite numbers, one can show that under the Diffie-Hellman Assumption, no probabilistic polynomial time algorithm can distinguish between z_1 and z_2. The algorithm is allowed to have access to p, g, y_1 and y_2.

It follows from the above result that the cryptosystem $\mathcal{C}_{owh}$ is semantically secure against chosen plaintext attacks. In other words, it leaks no partial information to attackers mounting chosen plaintext attacks. Note that if the $t = h(m)$ part is not enciphered together with m, some partial information on m may be leaked, and the resultant cryptosystem may not be semantically secure against chosen plaintext attacks.

Next we consider the cryptosystem $\mathcal{C}_{uhf}$. For *truly random* strings $z \in \Sigma^P$ and $s \in \Sigma^Q$, neither $z \oplus m$ nor $h_s(m)$ leaks any information on m (in the sense of Shannon [15]), where h_s is the hash function specified by the string s (see Section 4.2). In addition, when z and s are *independent* of each other, no information on m is leaked from $z \oplus m$ together with $h_s(m)$. Now let $x_1, x_2 \in_R [1, p-1]$, $y_1 = g^{x_1}$ and $y_2 = g^{x_2}$. Let z be the first P-bit substring and s be the next Q-bit substring of the output of the pseudo-random string generator G on input $g^{x_1 x_2}$. Then to a probabilistic polynomial time algorithm which is allowed to have access to p, g, y_1 and y_2, the two strings z and s look like independent random strings. Consequently, no partial information on m can be obtained by a probabilistic polynomial time algorithm, which is given as input $z \oplus m$, $h_s(m)$, p, g, y_1 and y_2. From this it follows that $\mathcal{C}_{uhf}$ is semantically secure against chosen plaintext attacks.

The above informal arguments for the semantic security of the first two public key cryptosystems, $\mathcal{C}_{owh}$ and $\mathcal{C}_{uhf}$, can be easily translated into formal proofs in a way similar to the proof of security of the cryptosystem proposed in [25]. Thus we have the following result.

Theorem 1 *Under the Diffie-Hellman Assumption (Assumption 1), both $\mathcal{C}_{owh}$ and $\mathcal{C}_{uhf}$, are semantically secure against chosen plaintext attacks.*

Unlike the previous two cryptosystems, we are not able to prove that the cryptosystem $\mathcal{C}_{sig}$ is also semantically secure against chosen plaintext attacks. This is mainly caused by the difficulty in measuring the amount of information on m leaked by the third part $c_3 = (h(m) - xr)/k \bmod (p-1)$ in the ciphertext. It is further complicated by the requirement that $c_4 = z \oplus m$ also has to be taken into consideration together with c_3. Nevertheless, when the one-way hash function is carefully chosen so that it behaves like a random function, the cryptosystem apparently leaks no partial information to attackers mounting chosen plaintext attacks.

5.2 Security against Chosen Ciphertext Attacks

Recall that the output of the enciphering algorithm of the cryptosystem $\mathcal{C}_{owh}$ is (c_1, c_2), where $c_1 = g^x$, $c_2 = z \oplus (m||t)$, $t = h(m)$ and h is a one-way hash function. The enciphering algorithm defines a function that maps an element (x, m) from $[1, p-1] \times \Sigma^P$ to an element (c_1, c_2) in $[1, p-1] \times \Sigma^{P+\ell}$. Due to the involvement of $t = h(m)$, the creation of the ciphertext is apparently impossible without the knowledge of x and m. Similar observations apply to the cryptosystems $\mathcal{C}_{uhf}$ and $\mathcal{C}_{sig}$. This motivates us to introduce a notion called *sole-samplable space*. A related notion was used by Damgård in the investigation of the security of his second public key cryptosystem [5].

Let f be a function from $D = \bigcup_n D_n$ to $R = \bigcup_n R_n$, where $D_n \subseteq \Sigma^n$, $R_n \subseteq \Sigma^{Q_1}$ and $Q_1 = Q_1(n)$ is a polynomial. We call $R = \bigcup_n R_n$ *the space induced by the function* f. Informally, we say that the space $R = \bigcup_n R_n$ is *sole-samplable* if there is no other way to generate an element y in R_n than to pick an element x in D_n first and then to evaluate the function at the point x. To formally define sole-samplability, we need the following two types of Turing machines: sample generators and pre-image extractors.

A *sample generator* for the space $R = \bigcup_n R_n$ induced by a function f is a probabilistic polynomial time Turing machine $\mathcal{S}$ that, given n as input and access to an oracle O_R for the space R, outputs a Q_1-bit string. The oracle prints in one step a sample string $y \in R_n$ as the answer to a request $n \in \mathbb{N}$. $\mathcal{S}$ can query the oracle only by writing $n \in \mathbb{N}$ on a special tape and will read the oracle answer $y \in R_n$ on a separate answer-tape.

A *pre-image extractor* of a sample generator $\mathcal{S}$ is a probabilistic polynomial time Turing machine $\mathcal{X}$ that has complete access to the contents of $\mathcal{S}$'s tapes and can observe thoroughly the entire computation of $\mathcal{S}$. The input of $\mathcal{X}$ is an integer $n \in \mathbb{N}$ and the output of $\mathcal{X}$ is an n-bit string.

Definition 2 *Let f be a function from $D = \bigcup_n D_n$ to $R = \bigcup_n R_n$, where $D_n \subseteq \Sigma^n$, $R_n \subseteq \Sigma^{Q_1}$ and $Q_1 = Q_1(n)$ is a polynomial. The space $R = \bigcup_n R_n$ induced by the function f is* sole-samplable *if for any sample generator $\mathcal{S}$ and for any polynomial $Q_2 = Q_2(n)$, there is a pre-image extractor $\mathcal{X}$ of the sample generator $\mathcal{S}$ such that for all sufficiently large n,*

$$\Pr\{\mathcal{X}(1^n, \mathcal{S})\} \geqq 1 - 1/Q_2,$$

where $\Pr\{\mathcal{X}(1^n, \mathcal{S})\}$ is the probability that, when the output of $\mathcal{S}$ is a sample y from R_n that is different from those given by an oracle O_R, $\mathcal{X}$ outputs a string $x \in D_n$ such that $y = f(x)$.

Note that when a function f is *not* one-way, that is, the inverse function f^{-1} of f is computable in probabilistic polynomial time, the space R induced by f is trivially sole-samplable, as one can always compute the pre-image $x \in D_n$ of an element $y \in R_n$, which implies that there is essentially only one way to sample R_n, namely, picking x first and then computing $y = f(x)$. In this paper we are only interested in spaces induced by one-way functions.

A necessary condition for the space $R = \bigcup_n R_n$ induced by a one-way function f to be sole-sample is that R be *sparse*. That is, $\#R_n/2^{Q_1} < 1/Q_2$ for any polynomial

$Q_2 = Q_2(n)$ and for all sufficiently large n. Otherwise, if R is non-sparse, one can always generate with a high probability a sample of R_n simply by flipping Q_1 coins. However, sparseness is *not* a sufficient condition for sole-samplability. As an example, consider the space induced by the one-way function $f(x) = f'(x)||f'(x)$, where f' is a one-way permutation on $\bigcup_n \Sigma^n$. Although the space is sparse (as we have $R_n/2^{2n} = 2^n/2^{2n} = 1/2^n < 1/Q_2$ for any polynomial $Q_2 = Q_2(n)$ and for all sufficiently large n), a sample $y = y'||y' \in \Sigma^{2n}$ can be readily obtained by flipping n coins. It is an interesting subject for future research to investigate other conditions for the space induced by a one-way function to be sole-samplable.

We will use the following assumptions in the proofs of security of the three cryptosystems. The assumptions are concerned with the sole-samplability of the spaces induced by the functions defined by the enciphering algorithms of the cryptosystems. These assumption are apparently reasonable thanks to the involvement of a tag in the generation of the ciphertext of a plaintext message. For the sake of simplicity, "the space induced by the functions defined by the enciphering algorithm" will be called "the space induced by the enciphering algorithm".

Assumption 2 *The space induced by the enciphering algorithm of the cryptosystem $\mathcal{C}_{owh}$ is sole-samplable.*

Assumption 3 *The space induced by the enciphering algorithm of the cryptosystem $\mathcal{C}_{uhf}$ is sole-samplable.*

Assumption 4 *The space induced by the enciphering algorithm of the cryptosystem $\mathcal{C}_{sig}$ is sole-samplable.*

We say that two assumptions A1 and A2 are comparable if either A1 implies A2 or A2 implies A1. Otherwise we say that A1 and A2 are incomparable. Examples of comparable assumptions are the Diffie-Hellman Assumption (Assumption 1) and the assumption that discrete logarithms over large finite fields are intractable. They are comparable as the former implies the latter. Now we consider the Diffie-Hellman Assumption and Assumption 2 (Assumptions 3 and 4, respectively). Note that Assumption 2 (Assumptions 3 and 4, respectively) holds even if the Diffie-Hellman Assumption does *not* hold. The former may hold if the latter *does* hold. Therefore Assumption 2 (Assumptions 3 and 4, respectively) may hold regardless of the Diffie-Hellman Assumption. In other words, Assumption 2 (Assumptions 3 and 4, respectively) and the Diffie-Hellman Assumption may be incomparable. It is worthwhile to investigate the exact relations among the assumptions.

The following theorem reveals the relevance of sole-samplability to security of cryptosystems.

Theorem 2 *Assume that the space induced by the enciphering algorithm of a public key cryptosystem is sole-samplable. Then the cryptosystem is semantically secure against adaptively chosen ciphertext attacks if and only if it is semantically secure against chosen plaintext attacks.*

Proof. The "only if" part is trivially true. Now we prove the "if" part by showing that for a public key cryptosystem whose enciphering algorithm induces a sole-samplable space, an adaptively chosen ciphertext attacker can do *no* better than a chosen plaintext attacker. Thus security of the cryptosystem against adaptively chosen ciphertext attacks is reduced to its security against chosen plaintext attacks.

Recall that an adaptively chosen ciphertext attacker consists of a pair $(\mathcal{L}_{acc}, \mathcal{T}_{acc})$ of probabilistic polynomial time Turing machines, where $\mathcal{L}_{acc}$ is an adaptively chosen ciphertext collector and $\mathcal{T}_{acc}$ a partial information extractor. Suppose $\mathcal{L}_{acc}$ queries the deciphering algorithm $Q = Q(n)$ times, each time with a different ciphertext c_i. Consider the first ciphertext c_1. Since the space induced by the enciphering algorithm is sole-samplable, the pre-image of c_1, part of which is the plaintext m_1 of c_1, can be computed in probabilistic polynomial time from the history of $\mathcal{L}_{acc}$'s computation. In other words, querying the deciphering algorithm with c_1 gives $\mathcal{L}_{acc}$ no more information, since the history of $\mathcal{L}_{acc}$'s computation contains already the answer to c_1. Similar arguments apply to c_2, c_3, $\ldots$, c_Q. Thus the ability to have access to the deciphering algorithm gives $\mathcal{L}_{acc}$ *no* advantage in its computation, and hence $\mathcal{L}_{acc}$ can be completely simulated by a probabilistic polynomial time Turing machine $\mathcal{L}'$ which has n, pk and an object ciphertext as input and has *no* access to the deciphering algorithm.

Now we have reduced the adaptively chosen ciphertext attacker $(\mathcal{L}_{acc}, \mathcal{T}_{acc})$ into another pair $(\mathcal{L}', \mathcal{T}_{acc})$ of probabilistic polynomial time Turing machines. Note that the input to $\mathcal{L}'$ consists of n, pk and an object ciphertext, while the input to $\mathcal{T}_{acc}$ consists of n, pk, an object ciphertext and the output of $\mathcal{L}'$. Consider a chosen plaintext attacker $(\mathcal{L}_{cp}, \mathcal{T}_{cp})$. The input to $\mathcal{L}_{cp}$ consists of n and pk, while the input to $\mathcal{T}_{cp}$ consists of, in addition to n and pk, an object ciphertext and the output of $\mathcal{L}_{cp}$. Therefore, the main difference between $(\mathcal{L}', \mathcal{T}_{acc})$ and $(\mathcal{L}_{cp}, \mathcal{T}_{cp})$ is that $\mathcal{L}'$ has, in addition to n and pk, an object ciphertext as input, while $\mathcal{L}_{cp}$ has only n and pk as input. This difference can be eliminated by letting $\mathcal{T}_{acc}$, which has an object ciphertext as input, accomplish that part of $\mathcal{L}'$'s computation which has to use an object ciphertext. Thus $(\mathcal{L}', \mathcal{T}_{acc})$ can be completely simulated by a chosen plaintext attacker $(\mathcal{L}_{cp}, \mathcal{T}_{cp})$.

Putting the above discussions together, we know that an adaptively chosen ciphertext attacker $(\mathcal{L}_{acc}, \mathcal{T}_{acc})$ can be completely simulated by a chosen plaintext attacker $(\mathcal{L}_{cp}, \mathcal{T}_{cp})$. From this it follows immediately that the "if" part is true, i.e., the cryptosystem is semantically secure against adaptively chosen ciphertext attacks if it is semantically secure against chosen plaintext attacks. □

Theorem 2 is interesting in that it not only relates sole-samplability to security of a cryptosystem, but also suggests an approach to the construction of public key cryptosystems that attain security against adaptively chosen ciphertext attacks.

By Theorems 1 and 2, our first two cryptosystems, $\mathcal{C}_{owh}$ and $\mathcal{C}_{uhf}$, are both semantically secure against adaptively chosen ciphertext attacks, under Assumptions 2 and 3, respectively, and the Diffie-Hellman Assumption. As discussed at the end of Section 5.1, we are not able to prove semantic security against chosen plaintext attacks of the cryptosystem $\mathcal{C}_{sig}$ under the Diffie-Hellman Assumption. In order to prove semantic security against adaptively chosen ciphertext attacks of $\mathcal{C}_{sig}$, we have to use an assumption stronger than the Diffie-Hellman Assumption, namely, that $\mathcal{C}_{sig}$ is semantically secure against chosen plaintext attacks. These discussions lead to the following theorem:

Theorem 3 *The three cryptosystems, $\mathcal{C}_{owh}$, $\mathcal{C}_{uhf}$ and $\mathcal{C}_{sig}$, are all semantically secure against adaptively chosen ciphertext attacks, under (1) Assumption 2 and the Diffie-Hellman Assumption, (2) Assumption 3 and the Diffie-Hellman Assumption, and (3) Assumption 4 and the assumption that it is semantically secure against chosen plaintext attacks, respectively.*

6 Extensions of the Cryptosystems

We have focused our attention on cryptosystems based on the discrete logarithm problem in finite fields. The cryptosystems can also be based on discrete logarithms over other kinds of finite abelian groups, such as those on elliptic or hyper-elliptic curves defined over finite fields [26, 27]. Another variant of the cryptosystems is to have a different large prime for each user. This variant can greatly improve practical security of the cryptosystems when a large number of users are involved.

Our first two methods for immunization, namely immunization with one-way hash functions and immunization with universal hash functions, can be applied to public key cryptosystems based on other intractable problems. For example, the methods can be used to immunize the probabilistic public key cryptosystem proposed in [25], which is based on the intractability of factoring large composite numbers. The methods might be extended further in such a way that allows us to construct from *any* trap-door one-way function a public key cryptosystem secure against adaptively chosen ciphertext attacks.

Authentication is another important aspect of information security. In many situations, the receiver of a message needs to be assured that the received message is truly originated from its sender and that it has not been tampered with during its transmission. Researchers have proposed many, unconditionally or computationally, secure methods for information authentication [28]. We take the cryptosystem $\mathcal{C}_{uhf}$ as an example to show that our cryptosystems can be easily enhanced with information authentication capability.

To do so, it is required that the sender Bob also has a pair (y_B, x_B) of public and secret keys. Information authentication is achieved by letting Bob's secret key x_B be involved in the creation of a ciphertext. More specifically, we change the step 2 of the enciphering Algorithm 3 to "$r = y_A^{x_B+x}$." and the step 1 of the corresponding deciphering Algorithm 4 to "$r' = (y_B c_1)^{x_A}$." Although ciphertexts from Alice to Bob are indistinguishable from those from Bob to Alice, it is infeasible for a user differing from Alice and Bob to create a "legal" ciphertext from Alice to Bob or from Bob to Alice. This property ensures information authentication capability of the cryptosystem. From the observation following the definition of the Diffie-Hellman Assumption (Assumption 1), we know that computing $g^{x_1(x_2+x_3)}$ from g^{x_1}, g^{x_2} and g^{x_3}, and computing

$g^{x_1 x_2}$ from g^{x_1} and g^{x_2}, are equally difficult. Therefore the authentication-enhanced cryptosystem is as secure as the original one.

The cryptosystem $\mathcal{C}_{owh}$ can be enhanced with information authentication capability in a similar way. For the cryptosystem $\mathcal{C}_{sig}$, the capability can be added by simply replacing x, a random string chosen from $[1, p-1]$, with Bob's secret key x_B.

7 Conclusions

We have presented three methods for immunizing public key cryptosystems against chosen ciphertext attacks, among which the second immunization method based on the use of universal hash functions is particularly attractive in that no one-way hash functions are needed. Each immunization method is illustrated by an example of a public key cryptosystem based on the intractability of computing discrete logarithms in finite fields. The notion of sole-samplability has been formally defined, and an interesting relation between sole-samplability and security of cryptosystems has been revealed. This relation has been further applied in the formal proofs of security of the example public key cryptosystems. The generality of our immunization methods is shown by their applicability to public key cryptosystems based on other intractable problems, such as that of factoring large composite numbers. An enhancement of information authentication capability to the example cryptosystems has also been suggested.

Acknowledgment

We would like to thank Thomas Hardjono for his insightful suggestions during the various stages of this research. We also would like to thank anonymous referees for their helpful comments.

References

[1] M. Blum, P. Feldman, and S. Micali, "Non-interactive zero-knowledge proof systems and applications," in *Proceedings of the 20-th Annual ACM Symposium on Theory of Computing*, pp. 103–112, 1988.

[2] M. Naor and M. Yung, "Public-key cryptosystems provably secure against chosen ciphertext attacks," in *Proceedings of the 22-nd Annual ACM Symposium on Theory of Computing*, pp. 427–437, 1990.

[3] D. Dolev, C. Dwork, and M. Naor, "Non-malleable cryptography," in *Proceedings of the 23-rd Annual ACM Symposium on Theory of Computing*, 1991.

[4] C. Rackoff and D. Simon, "Non-interactive zero-knowledge proof of knowledge and chosen-ciphertext attacks," in *Advances in Cryptology - Proceedings*

of Crypto'91, Lecture Notes in Computer Science, Vol.576 (J. Feigenbaum, ed.), pp. 433–444, Springer-Verlag, 1992.

[5] I. Damgård, "Towards practical public key systems secure against chosen ciphertext attacks," in *Advances in Cryptology - Proceedings of Crypto'91,* Lecture Notes in Computer Science, Vol. 576 (J. Feigenbaum, ed.), pp. 445–456, Springer-Verlag, 1992.

[6] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 472–492, 1976.

[7] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. IT-31, no. 4, pp. 469–472, 1985.

[8] S. C. Pohlig and M. E. Hellman, "An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance," *IEEE Transactions on Information Theory*, vol. IT-24, no. 1, pp. 106–110, 1978.

[9] B. A. LaMacchia and A. M. Odlyzko, "Computation of discrete logarithms in prime fields," *Designs, Codes and Cryptography*, vol. 1, pp. 47–62, 1991.

[10] R. Rivest, "Cryptography," in *Handbook of Theoretical Computer Science, Volume A, Algorithms and Complexity* (J. van Leeuwen, ed.), ch. 13, pp. 717–755, Cambridge, Massachusetts: The MIT Press, 1990.

[11] S. Goldwasser, S. Micali, and R. Rivest, "A digital signature scheme secure against adaptively chosen message attacks," *SIAM Journal on Computing*, vol. 17, no. 2, pp. 281–308, 1988.

[12] A. C. Yao, "Theory and applications of trapdoor functions (extended abstract)," in *Proceedings of the 23-rd Annual IEEE Symposium on Foundations of Computer Science*, pp. 80–91, IEEE Computer Society Press, 1982.

[13] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of Computer and System Sciences*, vol. 28, no. 2, pp. 270–299, 1984.

[14] S. Micali, C. Rackoff, and B. Sloan, "The notion of security for probabilistic cryptosystems," *SIAM Journal on Computing*, vol. 17, no. 2, pp. 412–426, 1988.

[15] C. E. Shannon, "Communication theory of secrecy systems," *Bell System Technical Journal*, vol. 28, pp. 656–715, 1949.

[16] M. Rabin, "Digitalized signatures as intractable as factorization," Technical Report MIT/LCS/TR-212, MIT, Laboratory for Computer Science, 1979.

[17] M. Blum and S. Micali, "How to generate cryptographically strong sequences of pseudo-random bits," *SIAM Journal on Computing*, vol. 13, no. 4, pp. 850–864, 1984.

[18] D. L. Long and A. Wigderson, "The discrete logarithm hides O(log n) bits," *SIAM Journal on Computing*, vol. 17, no. 2, pp. 363–372, 1988.

[19] R. Peralta, "Simultaneous security of bits in the discrete log," in *Advances in Cryptology - Proceedings of EuroCrypt'85,* Lecture Notes in Computer Science, Vol. 219 (F. Pichler, ed.), pp. 62–72, Springer-Verlag, 1985.

[20] S. Micali and C. P. Schnorr, "Efficient, perfect polynomial random number generators," *Journal of Cryptology*, vol. 3, no. 3, pp. 157–172, 1991.

[21] J. Carter and M. Wegman, "Universal classes of hash functions," *Journal of Computer and System Sciences*, vol. 18, pp. 143–154, 1979.

[22] M. Wegman and J. Carter, "New hash functions and their use in authentication and set equality," *Journal of Computer and System Sciences*, vol. 22, pp. 265–279, 1981.

[23] D. R. Stinson, "Combinatorial techniques for universal hashing," Report Series #127, Department of Computer Science, University of Nebraska, Lincoln, November 1990. (Also submitted to *Journal of Computer and System Sciences*).

[24] Y. Zheng, T. Hardjono, and J. Seberry, "A practical non-malleable public key cryptosystem," Technical Report CS91/28, Department of Computer Science, University College, University of New South Wales, 1991.

[25] M. Blum and S. Goldwasser, "An efficient probabilistic public key encryption scheme which hides all partial information," in *Advances in Cryptology - Proceedings of Crypto'84,* Lecture Notes in Computer Science, Vol. 196 (G. R. Blakeley and D. Chaum, eds.), pp. 289–299, Springer-Verlag, 1985.

[26] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203–209, 1987.

[27] N. Koblitz, "Hyperelliptic cryptosystems," *Journal of Cryptology*, vol. 1, no. 3, pp. 139–150, 1989.

[28] G. J. Simmons, "A survey of information authentication," *Proceedings of IEEE*, vol. 76, pp. 603–620, 1988.