A Secure and Efficient Scheme for Signed ElGamal Encryption

Fuw-Yi Yang
Department of Electronic Engineering
Chienkuo Technology University
Changhua 500, Taiwan, R.O.C.
Email: yangfy@ms7.hinet.net

Abstract. Previously, two schemes were proposed to enhance the security of ElGamal encryption. The weakness of the original scheme is improved by adding a message authentication tag to ciphertext. The resultant schemes are non-malleable encryption under chosen ciphertext attack. We observed that the same security goal can be achieved with less computational cost. The proposed scheme saves 33% of computing power on average. In processing decryption, it can save at most 46% of computations if the technique of multiple simultaneous exponentiations is used.

Keywords: Chosen ciphertext attack, ElGamal encryption, indistinguishable encryption, non-malleable encryption.

1. INTRODUCTION

With the quick and ongoing growth of digitalized information, more and more data are being exchanged. Data transferred would be safe from eavesdrop or modification if participants communicate over a secure channel. However, building and maintaining a secure channel for any two prospective participants isn’t a good solution to provide secure communication.

Generally, communication parties communicate over an insecure channel (an open channel). This channel could be a telephone line, computer network or Internet, for example. It is not difficult to have a tapping device tagged on these channels. Therefore, there are eavesdroppers and intruders who can intercept and modify messages transmitted in an open channel.

Let’s call the message to be transmitted plaintext \( m \). Plaintext can be text data, executable program or any kind of information. Instead of sending plaintext to a recipient, an encryption scheme \( E(.) \) is used to encrypt a plaintext and obtain a ciphertext \( c \). Then the ciphertext is transmitted to the recipient. The recipient can turn the received ciphertext into plaintext by a decryption scheme \( D(.) \). Needless to say, there must be some methods so that the processes of enciphering and deciphering work correctly, namely \( m = D(c) = D(E(m)) \), and no one except the recipient can decrypt the ciphertext to obtain the embedded plaintext.

In the public-key cryptography [1-4], each participant has a published public key \( pk \) and a hidden secret key \( sk \). Let \( pk_r \) and \( sk_r \) denote the recipient’s public key and secret key. Then the sender using the recipient’s public key to construct a ciphertext, \( c = E(pk_r, m) \), and sends it to the recipient. Upon receiving a ciphertext \( c \), the recipient uses secret key to recover plaintext, \( m = D(sk_r, c) \). Theoretically, any eavesdropper should have no idea about the plaintext, since only the recipient knows the matched secret key \( sk_r \). In this way, the recipient’s privacy is protected.

Many public key encryption schemes have been developed. ElGamal encryption scheme [5] has been always popular among them. A brief description is as follows. Let \( p \) be a large prime; \( g \) is an element of \( Z_p^* = \{1, 2, \ldots, (p - 1)\} \) with order \( (p - 1) \). A recipient randomly selects \( x_r \) from \( Z_{(p - 1)} = \{0, 1, \ldots, (p - 2)\} \) and computes \( y_r = g^{x_r} \mod p \), then publishes \( y_r \) as her/his public key and keeps \( x_r \) secretly as secret key. To encrypt a plaintext \( m \) (\( m \) is an element of \( Z_p^* \)), a sender randomly selects an element \( k \) from \( Z_{(p - 1)} \) and computes \( r = g^k \mod p \) and \( c = m \cdot (y_r)^r \mod p \). A
ciphertext consists of \( r \) and \( c \), i.e. \( y = (c, e) \). Upon receiving a ciphertext \((c, e)\), the recipient uses secret key \( x_r \) to recover plaintext, i.e. \( m = c / (r)^{x_r} \mod p \).

Assuming that Decision Diffie-Hellman problem is hard, ElGamal encryption scheme is indistinguishable encryption under chosen plaintext attack as can be seen in [6-7]. We give some words to explain secure encryption and attack model. Assume that an adversary is given the advantage of choosing two messages \( m_0 \) and \( m_1 \) (the same length). One of them is encrypted and hands on the ciphertext to the adversary. An encryption scheme is indistinguishable encryption if the adversary cannot guess whether the given ciphertext is an encryption of message \( m_0 \) or \( m_1 \) in polynomial time. This definition of security comes from [8-10] and is called polynomial indistinguishability (also known as semantic security). In the attack model of chosen plaintext, an adversary can access encryption oracle while trying to extract some information about a given ciphertext. A more powerful attack is adaptive chosen ciphertext attack [11-12] (also called CCA2 in [12]; hereafter we simply write chosen ciphertext attack). In this model an adversary is allowed to access another oracle, decryption oracle, while launching attack. Surely, the adversary is inhibited to ask decryption oracle to decrypt the given ciphertext. For more details about notion of security and attack model please refer to [4, 8-12].

1.1 Related Work

However, ElGamal encryption is not secure in the sense of indistinguishability under chosen ciphertext attack. In order to enhance the security of ElGamal encryption, many researchers have improved on the original ElGamal encryption. The work in [6] and [13] put forward a tag of message authentication for improving the security. Essentially, the message authentication is a signature on ciphertext. In paper [13], the enhanced scheme is called signed ElGamal encryption. Although a ciphertext accompanied with a signature, it still is an anonymous ciphertext since the sender uses one-time public key to generate the signature. Namely, for each encryption, the sender chooses a temporary secret key and computes the corresponding public key. After enciphering (include signing), sender sends ciphertext (include signature and the temporary public key) to recipient. A successful verification in signature (with respect to the temporary public key) implies that the ciphertext was really constructed and sent by the original sender.

Since the message authentication is a signature on ciphertext (include other necessary information), it is no longer easy to modify ciphertext. Therefore, both improvements achieve the security goal of non-malleable encryption [14-15] under chosen ciphertext attack; this goal is equivalent to indistinguishable encryption [12]. Assume that a ciphertext is given to an adversary and the adversary can construct another different ciphertext based on the given ciphertext. The security goal requires that their embedded plaintexts are related with negligible probability. The following example shows that ElGamal encryption is a malleable encryption. Assume that \( c_1 = (r, c) \) is a ciphertext, i.e. \( r = g^k \mod p \), \( c = m_1 \cdot (y)^k \mod p \). Then \( c_1 \) and \( c_2 = (r, c \cdot 2 \mod p) \) are related, i.e. \( m_2 = Dec(c_2) = m_1 \cdot 2 \mod p \).

1.2 Contributions

This paper will propose a new scheme of Signed ElGamal encryption, which is non-malleable encryption but computational cost is more efficient than that of the schemes in [6, 13]. More precisely, the schemes in [6, 13] requires three modular exponentiations to encrypt (decrypt) a plaintext, whereas our scheme requires only two modular exponentiations to do it.

As for the size of ciphertext, the proposed scheme and the scheme in [13] have the same bit length and both are smaller than that of the scheme in [6]. A table in Section 5 compares the size of ciphertexts and computations among the three schemes.
### 1.3 Organization

Section 2 introduces notation. Section 3 reviews the Signed ElGamal encryption scheme in [13]. Section 4 describes the proposed scheme and proves its correctness as well as security. Section 5 discusses the scheme’s performance. Finally, Section 6 concludes the paper.

### 2. Notation

Let $p$ and $q$ be two large primes and $q$ divide $(p - 1)$. The notation is as follows: $\mathbb{Z}_p = \{0, 1, ..., (p - 1)\}$; $\mathbb{Z}_p^*$ is a multiplicative group modulo $p$; $a \in \mathbb{Z}_p$ denotes that $a$ is an element of $\mathbb{Z}_p$; $g \in \mathbb{Z}_p^*$ with order $q$; $G \subset \mathbb{Z}_p^*$ is a cyclic group generated by $g$ and $|G|$ denotes its cardinality; randomly selected an element $a$ from $G$ is denoted by $a \leftarrow_R G$; $M$ denotes message space. In algorithms or Games, $l \leftarrow r$ implies that the value of $r$ is assigned to variable $l$. Also, $a || b$ denotes a concatenation of strings $b$ and $a$; $H(.)$ and $H_1(.)$ are one-way collision-resistant hash function. Furthermore, the hash function $H(.)$ has been modeled as a random oracle [16].

Assume that primes $p$, $q$ and generator $g$ have been chosen such that finding the discrete logarithm in $G$ is hard. Namely, given an attacker $p$, $q$, $g$, and $y \leftarrow_R G$, it is assumed that finding $x \in \mathbb{Z}_q$ such that $y = g^x \bmod p$ is computationally intractable.

### 3. Review a Signed ElGamal Encryption

This section reviews the signed ElGamal encryption scheme that was proposed in [13]. The message space is $G$ ($M = G$) and hash function $H(.)$ is defined as $H : G^2 \times M \to \mathbb{Z}_q$. Recipient’s secret and public keys are $x_r \in \mathbb{Z}_q$ and $y_r = g^{x_r} \bmod p$. Note that the sender’s temporary secret and public keys are $x_s \in \mathbb{Z}_q$ and $y_s = g^{x_s} \bmod p$. The algorithms of encryption and decryption are:

**Encryption Algorithm** $Enc(m, y_r)$

- $k, x_r \in_R \mathbb{Z}_q$, $y_s \leftarrow g^{x_s} \bmod p$, $r \leftarrow g^k \bmod p$
- $Y_r \leftarrow (y_s)^k \bmod p$, $c \leftarrow m \cdot Y_r \bmod p$
- $e \leftarrow H(y_r || r || c), s \leftarrow x_s + e \cdot k \bmod q$
- $\psi \leftarrow (r, c, e, s)$

Return ciphertext $\psi$

**Decryption Algorithm** $Dec((r, c, e, s), x_s)$

- $y_r \leftarrow (r^e \cdot (r^c)^e) \bmod p$
- If $e \neq H(y_r || r || c)$
  - Then $m \leftarrow \text{"invalid"}$
  - Else $Y_r \leftarrow (r^c)^r \bmod p$, $m \leftarrow c / Y_r \bmod p$

Return $m$

To simplify the estimation of computational cost, we count only the major operation. For example, the computational cost of modular multiplication and hash function is ignored as compared with the expensive cost of modular exponentiation.

The computational cost for encryption is three modular exponentiations; similarly, decryption algorithm also requires three modular exponentiations.

### 4. The Proposed Scheme and its Security

A recipient randomly selects $x_r$ from $\mathbb{Z}_q$ and computes $y_r = g^{x_r} \bmod p$, then publishes $y_r$ as her/his public key and keeps $x_r$ secretly as secret key. The message space is $G$ ($M = G$) and collision resistant hash functions $H(.)$ and $H_1(.)$ are defined as $H(.) : G^3 \times M \to \mathbb{Z}_q$ and $H_1(.) : G \to G$. Note that the sender’s temporary secret and public keys are $x_s \in R \mathbb{Z}_q$ and $y_s = g^{x_s} \bmod p$. The details of encryption and decryption are as follows.

**Encryption Algorithm** $Enc(m, y_r)$

- $k, x_r \in_R \mathbb{Z}_q$
- $y_s \leftarrow g^{x_s} \bmod p$, $Y_r \leftarrow (y_s)^k \bmod p$
- $c \leftarrow m \cdot H(Y_r) \bmod p$, $e \leftarrow H(Y_r || y_r || y_r || c)$
- $s \leftarrow k \cdot e \cdot x_s \bmod q$, $\psi \leftarrow (y_s, c, e, s)$

Return ciphertext $\psi$

**Decryption Algorithm** $Dec((y_s, c, e, s), x_s)$

- $Y_r \leftarrow (y_s)^e \cdot (y_s)^c \bmod p$
- $m \leftarrow c \cdot H(Y_r) \bmod p$
- If $e \neq H(y_r || y_s || y_s || c)$
  - Then $m \leftarrow \text{"invalid"}$

Return $m$
4.1 Correctness

**Lemma 1.** The algorithm $\text{Dec}(.)$ correctly recovers the plaintext embedded in a ciphertext produced by algorithm $\text{Enc}(.)$.

**Proof.** Assume that a recipient received a ciphertext $(y_s, c, e, s)$. Then she/he runs algorithm $\text{Dec}(y_s, c, e, s)$ to recover plaintext $m$ and verify signature. The quantity $Y_r = (y_r)^s$ is recovered by the computation $Y_r = (g^s \cdot (y_s)^e)^r \mod p$. Then plaintext is obtained by computing $m = c \cdot H(Y_r) \mod p$. Therefore the signature $(e, s)$ on message $(y_r, y_s \parallel c)$ is correctly verified, and a success in verification indicates that the embedded plaintext is correctly recovered. □

4.2 Security of Signature

The signature in encryption algorithm serves as message authentication, which is a variant of Schnorr signature since it permits only receiver to verify its validity. A valid verification of the signature convinces recipient of genuine ciphertext. The security of Schnorr signature scheme has received extensive discussion and has been proven to be *existentially unforgeable* under the adaptive chosen message attack [17-20]. The terminology *existentially unforgeable* originally defined in [21] is a common security goal of signature. It implies that any adversary should have a negligible probability in forging a valid signature on a new message. The attack model is called adaptive chosen message if the adversary is allowed to access a signing oracle while forging a signature.

Assume that an adversary $\mathcal{A}$ has maximum advantage $\text{ADV}_{\text{CMA-Sig-Schnorr}}(\mathcal{A})$ while trying to forge a Schnorr signature under the adaptive chosen message attack. It is estimated that the quantity of $\text{ADV}_{\text{CMA-Sig-Schnorr}}(\mathcal{A})$ is negligible [17, 18]. In the following lemma, we will prove that the signature scheme used in algorithm $\text{Enc}(.)$ (let’s call it *Sig-scheme*) is at least as secure as Schnorr signature scheme. Namely, for the same public-key pair $(x_s, y_s)$, the probability of forging a Schnorr signature is equal or larger than that of forging a *Sig-scheme* signature.

A ciphertext $(y_s, c, e, s)$ is essential a signature of *Sig-scheme*, i.e. a signature on message $(y_r \parallel y_s \parallel c)$ and verified by checking that $e \overset{?}{=} H(Y_r \parallel y_r \parallel y_s \parallel c)$, where $Y_r = (g^s \cdot (y_s)^e)^r \mod p$, $e = H(Y_r \parallel y_r \parallel y_s \parallel c)$, $s = (k - e \cdot x_s) \mod q$. Indeed, *Sig-scheme* is a variant of Schnorr signature with restricted verifiable information. Nobody except sender and recipient can verify this signature unless the recipient releases information $Y_r$ (see Lemma 2). Thus the result of the following lemma seems reasonable.

**Lemma 2.** The signature scheme *Sig-scheme* is at least as secure as Schnorr signature scheme.

**Proof.** In the following, we will include message in signature at our convenience. With the knowledge of secret key $x_r$, recipient transforms a ciphertext $(y_s, c, e, s)$ into a Schnorr signature. The details of transformation are as follows. Recipient runs the algorithm $\text{Dec}(.)$; computes and releases the quantity $y_{sr} = (y_s)^r \mod p$. Then a Schnorr signature $(m', e, s)$ is obtained. Namely, message $m' = (y_r \parallel y_s \parallel c)$, response $s$, and challenge $e$ is a hash value on concatenation of message $m'$ and commitment $Y_r$, i.e. $e = H(Y_r \parallel m')$. However, the signature is verified by checking that $e \overset{?}{=} H((y_r)^e \cdot (y_{sr})^e \parallel m')$. Note that the basis $g$ and the signer’s public key $y_s$ has been transformed into $y_r = g^b \mod p$ and $y_{sr} = (y_s)^b \mod p$, respectively.

Although recipient is able to convert a signature of *Sig-scheme*, $(y_s, c, e, s)$, into a Schnorr signature; recipient cannot transform an ordinary Schnorr signature $(m', e', s')$ into a *Sig-scheme* signature, $(y_s, c, e, s)$. An exception is that message $m' = y_r \parallel y_s \parallel c$. Then $(y_s, c, e', s')$ is a *Sig-scheme* signature. However, this situation will occur with negligible probability due to Schnorr-signature’s property of *existential unforgeability*. 
Let the quantity $\text{ADV}_{\text{Sig-scheme}}^f(\mathcal{A})$ be $\mathcal{A}$’s advantage of forging a Sig-scheme signature under adaptive chosen message attack model. We conclude that

$$\text{ADV}_{\text{Sig-scheme}}^f(\mathcal{A}) \leq \text{ADV}_{\text{ Schnorr-sig}}^f(\mathcal{A}).$$ (1)

If equation (1) does not hold, then the adversary can first forge a Sig-scheme signature and transform it into a Schnorr signature. Therefore a Schnorr signature can be forged with a higher probability than $\text{ADV}_{\text{Schnorr-sig}}^f(\mathcal{A})$. This contradicts the assumption that $\text{ADV}_{\text{Schnorr-sig}}^f(\mathcal{A})$ is $\mathcal{A}$’s maximum advantage of forging a Schnorr signature. □

Benefiting by that a ciphertext is also a signature of Sig-scheme; the proposed scheme is non-malleable encryption. Assume that an adversary is given a ciphertext and is able to forge a different ciphertext such that the embedded plaintexts are related. It concludes that the adversary has either decrypted the ciphertext or succeeded in forging a signature of Sig-scheme. In the former case, the adversary can obtain plaintext and re-encrypt it using another one-time key pair. However, it is impossible since the adversary does not learn the receiver’s secret key. The latter case implies that the adversary received $\text{Enc}(m) = (y_s, c, e, s)$ and constructed $\text{Enc}(m') = (y'_s, c', e', s')$ such that $m$ and $m'$ are related. Using any one-time key pair, the adversary can construct any ciphertext, $\text{Enc}(m') = (y'_s, c', e', s')$, but the probability that $m$ and $m'$ are related is negligible, by Lemma 2.

The scheme is also secure against one-more-decryption attack. In our scheme, a success in one-more-decryption attack implies that given $l$ ciphertexts (signatures), an adversary can construct another ciphertext different from those $l$ ciphertexts. It would be impossible for an adversary (include recipient) to do that, since the security of Sig-scheme is strong enough to resist this attack, by Lemma 2.

**4.3 Security of Encryption**

In the following, we will use the technique of sequences games described in [7] to prove that the proposed scheme is indistinguishable encryption under chosen ciphertext attack. The proof seems unnecessary since non-malleable encryption implies indistinguishable encryption as can be seen in [12, 14-15]. However, we are trying to obtain a more concrete result of security goal. So let’s proceed as planned.

Our proof consists of four games: $\text{Game}_0$, $\text{Game}_1$, $\text{Game}_2$, and $\text{Game}_3$. $\text{Game}_0$ demonstrates the definition of security goal and attack model. Namely, a challenger who honestly runs the algorithm specified in $\text{Game}_0$, and an adversary trying to break the algorithm in the sense of definition. $\text{Game}_1$ evolves from $\text{Game}_0$ with a small change. Finally, we achieve the final game, $\text{Game}_3$. Assume that the differences between games and the success probability of adversary in final game can be calculated efficiently. Then we can compute the adversary’s probability of successful experiment in $\text{Game}_0$, namely, in the original definition.

During an attack game, we assume that the adversary, $\mathcal{A}(\cdot)$, called encryption oracle $q_e$ times and decryption oracle $q_d$ times. Let $\psi$ denote ciphertext $(y_s, c, e, s)$ and $Q_j$ denote the number of encryption queries made prior to the $j$th decryption query. It is evident that $0 \leq Q_j \leq q_e$.

**Game0:** Definition of security goal and attack model

$x_r \leftarrow_R Z_q, y_r \leftarrow g^{x_r} \bmod p, b \leftarrow_R \{0, 1\}$ // recipient’s key pair; random bit $b$

Upon the $i$th encryption query:

$x_{si} \leftarrow_R Z_q, y_{si} \leftarrow g^{x_{si}} \bmod p$ // challenger picks sender’s one-time key pair

$(m_{i0}, m_{i1}) \leftarrow \mathcal{A}(y_r, \psi_1, ..., \psi_{i-1})$ // $\mathcal{A}$ picks message pair according to $\mathcal{A}$’s record
\[ k_i \leftarrow R \mathbb{Z}_q, Y_i \leftarrow (y_i)^{k_i} \mod p, c_i \leftarrow (m_{i0} / H_i(Y_{i})) \mod p /\text{ challenger’s enciphering} \]
\[ e_i \leftarrow H(Y_{i} || y_r || Y_{i} || c_i), s_i \leftarrow (k_i - e_i \cdot x_i) \mod q, \psi_i \leftarrow (y_{si}, c_i, e_i, s_i) \]
Return \( \psi_i \) as the answer // challenger send ciphertext \( \psi_i \) to \( A \)

Upon the \( j \)th decryption query (\( \psi_j' = (y_{sj}, c', e', s') \)):
\[ Y'_{ij} \leftarrow (g^{s_j'} \cdot (y_{sj})^{e_j'} \cdot y_r) \mod p, m'_j \leftarrow c' \cdot H_i(Y'_{ij}) \mod p \]
\[ e''_j \leftarrow H(Y'_{ij} || y_r || y_{sj} || c') \]
If \( e''_j = e'_j \) then return \( m'_j \) else return “invalid” // decryption oracle send \( m'_j \) to \( A \)
\[ b' \leftarrow A(\psi_1, \ldots, \psi_{q_1}, m'_1, \ldots, m'_{q_d}) \in \{0, 1\} /\text{ decides the random bit } b' /\text{ according to } A’s /\text{ record} \]
Output \( b' \) // \( A \) outputs the random bit \( b' \)

It is clearly that \( Game_0 \) describes the model of chosen ciphertext attack (CCA2) [12]. The adversary freely selects two plaintexts \( (m_{i0}, m_{i1}) \) and hands on them to challenger. According to the value of a pre-selected random bit \( b \), challenger enciphers either \( m_{i0} \) or \( m_{i1} \). Then challenger returns the resultant ciphertext to the adversary. While selecting plaintexts, adversary has knowledge of public information and some ciphertexts generated previously, \( \text{i.e.} \psi_1, \ldots, \psi_{q_1} \). Also, the adversary can query decryption oracle to decrypt some ciphertexts \( \psi_j' \) that \( \psi_j' \notin \{ \psi_a | a = 1, \ldots, Q_j \} \). Finally, the adversary is asked what value the random bit \( b \) is.

Let \( S_0 \) denote the event that \( b = b' \) in \( Game_0 \). If the adversary simply tosses a fair coin to decide random bit \( b' \), then the probability of event \( S_0 \) will be \( 1 / 2, \text{i.e.} \Pr[S_0] = 1 / 2 \). Since the adversary has been equipped with some capabilities to guess random bit \( b' \), the probability \( \Pr[S_0] \) may be higher than \( 1 / 2 \). Thus it is straightforward to define
\[ \text{ADV}_{\text{Enc}}^{\text{CCA2}} = 2 \cdot (\Pr[S_0] - 1 / 2). \]

Namely, it is the excess probability than that of guessing randomly. Now we try to compute the quantity of \( \text{ADV}_{\text{Enc}}^{\text{CCA2}} \) using the following games and differences between consecutive games.

**Game1: A change in decryption oracle**
\[ x_i \leftarrow R \mathbb{Z}_q, y_r \leftarrow g^{x_i} \mod p, b \leftarrow R \{0, 1\} \]
Upon the \( i \)th encryption query:
Remain the same as \( Game_0 \)
Upon the \( j \)th decryption query:
Return “invalid” as the answer
\[ b' \leftarrow A(\psi_1, \ldots, \psi_{q_1}, m'_1, \ldots, m'_{q_d}) \in \{0, 1\}, \text{Output } b' \]

Note that in \( Game_1 \), the decryption oracle always returns “invalid” to the adversary in response to his/her decryption queries. Let \( S_1 \) define the event that \( b = b' \) in \( Game_1 \). Also \( F \) defines the event that for some \( j = 1, \ldots, q_d \), the adversary queried decryption oracle with a ciphertext \( (y_{sj}, c', e', s') \), which is such that \( Y'_{ij} = (g^{s_j} \cdot (y_{sj})^{e_j} \cdot y_r) \mod p, m'_j = c' \cdot H_i(Y'_{ij}) \mod p, e' = H(Y'_{ij} || y_r || y_{sj} || c') \). It is clear that \( Game_0 \) and \( Game_1 \) will proceed identically unless the event \( F \) occurs. Essentially, the event \( F \) implies that the adversary has successfully forged a signature of \( \text{Sig-scheme} \). By Lemma 2 in Section 4.2, event \( F \) occurs with probability less than \( q_d \cdot \text{ADV}_{\text{Sig-scheme}}^{\text{cma-cma}} \), \text{i.e.}
\[ \Pr[F] \leq q_d \cdot \text{ADV}_{\text{Sig-scheme}}^{\text{cma-cma}} \leq q_d \cdot \text{ADV}_{\text{Schmorr-sig}}^{\text{cma-cma}}. \]

The Difference Lemma in [7] describes the relationship between events \( S_0, S_1 \), and \( F \). It states that \( | \Pr[S_0] - \Pr[S_1] | \leq \Pr[F] \). Therefore the following inequality is obtained.

**Equation (3)**

\[ |\Pr[S_0] - \Pr[S_1]| \leq q_d \cdot ADV_{\text{Schnorr-sig}}. \] (4)

**Game2:** Hash function \(H(.)\) is replaced by a truly faithfully random function

\[ x_r \leftarrow R Z_q, y_r \leftarrow g^{x_r} \mod p, b \leftarrow R \{0, 1\} \]

Upon the \(i\)th encryption query:

\[ x_{si} \leftarrow R Z_q, y_{si} \leftarrow g^{x_{si}} \mod p \]

\( (m_{0i}, m_{1i}) \leftarrow A(y_r, \psi_i, ..., \psi_{i-1}) \)

\( e_i \leftarrow R Z_q, s_i \leftarrow R Z_q, Y_{ri} \leftarrow (y_r)^{\psi_i} \cdot y_{si} \mod p \) // now \( e_i, s_i \) and \( Y_{ri} \) are random

\( c_i \leftarrow m_{bi} / H(Y_{ri}) \mod p \) // now \( c_i \) is random

If \( (Y_{ri} \| y_r \| y_{si} \| c_i = Y_{rij} \| y_r \| y_{sj} \| c_j) \) for some \( j < i \)

Then \( e_i \leftarrow e_j, s_i \leftarrow s_j \)

The hash value of \( H(Y_{ri} \| y_r \| y_{si} \| c_i) \) is \( e_i \) //simulation of random oracle \( H(.) \)

\( \psi_i \leftarrow (y_{si}, c_i, e_i, s_i) \)

Return \( \psi_i \) as the answer

Decryption query and \( A \)'s processing remain unchanged.

Note that in **Game2**, the hash function \(H(.)\) is replaced by a truly faithfully random function. Then the difference between **Game2** and **Game2** is just the difference between the hash function \(H(.)\) and a truly faithfully random function.

It is assumed that the hash function \(H(.)\) is an instance of pseudorandom function family. A random instance of pseudorandom function family is computationally indistinguishable from that of a random function. According to the value of a random bit, an adversary accesses to a pseudorandom function or random function. The adversary queries the oracle (say function \(f(.)\)) and receives answer (\(f(str), str \in \text{domain of } f(.)\)). After some queries, the adversary guesses whether the oracle is pseudorandom function or random function. Let \(ADV_{H(.)}^{\text{prf}}\) denote adversary’s advantage of guessing and \(S_2\) the event that \(b = b'\) in **Game2**. Then we have

\[ |\Pr[S_1] - \Pr[S_2]| \leq ADV_{H(.)}^{\text{prf}}. \] (5)

**Game 3:** The truly faithfully random function is replaced by a truly forgetfully random function

\[ x_r \leftarrow R Z_q, y_r \leftarrow g^{x_r} \mod p, b \leftarrow R \{0, 1\} \]

Upon the \(i\)th encryption query:

Remain the same as **Game2**, except the “if--then” statement is removed, which functions as collision detection.

Decryption query and \( A \)'s processing remain unchanged.

Note that in **Game3**, the truly faithfully random function is replaced by a truly forgetfully random function. Let \(\text{Collision}_3\) denote the event that \((Y_{ri} \| y_r \| y_{si} \| c_i = Y_{rij} \| y_r \| y_{sj} \| c_j)\) for some \(j \neq i\). \(\text{Collision}_3\) is expressed as follows \((G^3 = G \times G \times G)\):

\[
\Pr[\text{Collision}_3] \leq \frac{1}{|G^3|} + \frac{2}{|G^3|} + \ldots + \frac{q_e(q_e - 1)}{2|G^3|} \leq \frac{(q_e)^2}{2|G^3|}.
\] (6)

It is clear that **Game2** and **Game3** proceed identically unless \(\text{Collision}_3\) occurs. Let \(S_3\) denote the event that \(b = b'\) in **Game3**. Thus we have

\[ |\Pr[S_2] - \Pr[S_1]| \leq \Pr[\text{Collision}_3] \leq \frac{(q_e)^2}{2|G^3|}. \] (7)

Also that \(\Pr[S_3] = 1 / 2\). Therefore, combining equations (4), (5), (6), and (7), the adversary’s advantage of distinguishability is as follows.
\[ ADV_{\text{SIG}} = 2 \cdot | \Pr[S_0] - \Pr[S_3] | = 2 \cdot | \Pr[S_0] - \Pr[S_1] + \Pr[S_1] - \Pr[S_2] + \Pr[S_2] - \Pr[S_3] | \]
\[ \leq 2 \cdot (q_d \cdot ADV_{\text{CMA-Schnorr-SIG}} + ADV_{\text{PRF}}^{\text{H}} + \frac{(q_e)^2}{2|G|^2}) \]

In the public key cryptosystems, an adversary can ask encryption query at his/her will, but the advantage is still negligible, since \(q_e << |G|\). Assume that \( ADV_{\text{PRF}}^{\text{H}} < ADV_{\text{CMA-Schnorr-SIG}}^{\text{H}}\), then the adversary’s advantage of distinguishing random bit \(b\) is proportional to the product of \(ADV_{\text{CMA-Schnorr-SIG}}^{\text{H}}\) and the quantity of \(q_d\). Generally, \(q_d\) is controllable, say \(q_d < 2^{20}\). The equation above indicates that the proposed scheme is practical.

5. PERFORMANCE

Counting only modular exponentiation, the computational cost for encryption is two modular exponentiations; similarly, decryption algorithm also requires two modular exponentiations. The technique of efficient simultaneous multiple exponentiations [2] can be applied to further reduce the computations in decryption algorithm. Namely, the quantity \(Y_r = (g^s \cdot (y_r)^e)^{x_r} \mod p\) can be computed at the computational cost of 1.17 modular exponentiations (ME). Note that in the estimation we have neglected the modular multiplications \(s \cdot x_r \mod q\) and \(e \cdot x_r \mod q\). Table 1 compares the computational cost and size of ciphertext among the schemes in [6, 13] and the proposed scheme.

<table>
<thead>
<tr>
<th>Table 1. Comparisons of bit length and computations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
</tr>
<tr>
<td>Encryption</td>
</tr>
<tr>
<td>Decryption*</td>
</tr>
<tr>
<td>Decryption*</td>
</tr>
</tbody>
</table>

* using the technique of efficient simultaneous multiple exponentiations
# processing a valid ciphertext
$ processing an invalid ciphertext

6. CONCLUSION

This paper has presented a secure and efficient encryption scheme. In the aspect of signature, it is shown at least as secure as the Schnorr signature scheme. It is possible to integrate with other signature schemes in our encryption algorithm, e.g. DSA signature scheme.

As for the encryption, it is \textit{non-malleable encryption} and secure against one-more-decryption attack. Also, the adversary’s advantage in guessing random bit \(b\) is directly related to the advantage of forging a signature as shown in equation (8).

Further, Table 1 indicates that the proposed scheme is more efficient in computation than other schemes. It saves 33% of computing resources on average. Specifically, saves 46% when decrypting ciphertext if the technique of efficient simultaneous multiple exponentiations is implemented. We should point out that the out-performance in deciphering does not hold for invalid ciphertext.

Acknowledgements

We thank for anonymous reviewers for comments.

The research described in the document has been partially supported by the National Science Council under contract NSC 94-2213-E-270-008.

References