IND-CCA Public Key Schemes Equivalent to Factoring $n = pq$

Kaoru Kurosawa, Wakaha Ogata, Toshihiko Matsuo, and Shuichi Makishima

Tokyo Institute of Technology,
2-12-1 O-okayama, Meguro-ku, Tokyo 152-8552, Japan
{kurosawa,wakaha,tossy,maxima}@ss.titech.ac.jp

Abstract. Indistinguishability against adaptive chosen ciphertext attack (IND-CCA2) is the strongest notion for security of public key schemes. In this paper, we present the first IND-CCA2 schemes whose securities are equivalent to factoring $n = pq$ under the random oracle model, where p and q are prime numbers. Our first scheme works for long messages and our second scheme is more efficient for short messages.

1 Introduction

Indistinguishability against adaptive chosen ciphertext attack (IND-CCA2) is the strongest notion of security for public key schemes. Bellare and Rogaway showed that a trapdoor one-way permutation f can be converted into a IND-CCA2 public key scheme in the random oracle model [1]. They further presented another IND-CCA2 scheme [2], called OAEP, which is more efficient than their first scheme for short messages.

RSA is believed to be a trapdoor one-way permutation. However, it is not known that inverting RSA is equivalent to factoring $n = pq$, where p and q are prime numbers.

On the other hand, Okamoto and Uchiyama showed a probabilistic public key scheme such that inverting the encryption function is equivalent to factoring a special modulus $n = p^2q$ [3]. Fujisaki, Okamoto and then Pointcheval showed some conversions of Okamoto and Uchiyama scheme into IND-CCA2 public key schemes in the random oracle model [4,5,6]. Paillier presented a trapdoor one-way permutation by modifying Okamoto and Uchiyama scheme [7].

Paillier presented a probabilistic public key scheme which is IND-CPA under the composite residuosity assumption [8, Sec.4], where IND-CPA stands for indistinguishability against chosen plaintext attack. Paillier also showed a variant of his scheme which is a trapdoor one-way permutation if and only if inverting RSA is hard [8, Sec.5]. Paillier and Pointcheval gave a conversion of Paillier's scheme [8, Sec.4] into a IND-CCA2 public key scheme in the random oracle model [9].

However, no IND-CCA2 scheme is known whose security is equivalent to factoring $n = pq$. In this paper, we present the first IND-CCA2 schemes whose

K. Kim (Ed.): PKC 2001, LNCS 1992, pp. 36–47, 2001.

securities are equivalent to factoring $n = pq$ in the random oracle model by using Kurosawa et al's public key cryptosystem. Our first scheme works for long messages. Our second scheme is more efficient for short messages.

Rabin's public key cryptosystem [10] is as hard as factorization. However, it is not uniquely deciphered because four different plaintexts produce the same cipher. Williams showed that this disadvantage can be overcome if the secret two prime numbers, p and q, satisfy $p = 3 \bmod 8, q = 7 \bmod 8$ [11]. Kurosawa et al. [12] showed a public key cryptosystem such that (i) inverting is equivalent to factoring $n = pq$, (ii) the decryption is unique and (iii) p and q are arbitrary prime numbers.

Related works: Cramer and Shoup showed an IND-CCA2 scheme in the standard model under the decision Diffie-Hellman problem [13]

2 Preliminaries

Let k be a security parameter. Let $n(k)$ denote the length of a plaintext, where $n(k)$ is bounded by some polynomial on k.

2.1 Definitions

Definition 1. *A* public key encryption scheme *with a plaintext length function $n(\cdot)$ is a triple of algorithms, $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$, where*

- *$\mathcal{G}$, the key generation algorithm, is a probabilistic algorithm that takes a security parameter k and returns pair of (pk, sk) of matching public and secret keys,*
- *$\mathcal{E}$, the encryption algorithm, is a probabilistic algorithm that takes a public key pk and a message $x \in \{0,1\}^n$ to produce a ciphertext y,*
- *$\mathcal{D}$, the decryption algorithm, is a deterministic algorithm that takes a secret key sk and a ciphertext y to produce a message $x \in \{0,1\}^n$ or a special symbol $\perp$ to indicate that the ciphertext was invalid.*

Our goal is to construct an encryption scheme which is indistinguishable secure (or semantically secure). We consider an adversary $A = (A_1, A_2)$ who runs in two stages. In the find-stage A_1 is given an encryption algorithm $\mathcal{E}$ and outputs a pair (x_0, x_1) of messages. It also outputs some string str, for example, its history and its inputs. In the guess-stage A_2 is given the outputs of A_1, (x_0, x_1) and str, and also y which is a ciphertext of a message x_b for random bit b. A_2 guesses a bit b' from x_0, x_1 and y, and outputs a guessing bit b'.

A simple A_2 who always outputs 0 (or 1) can succeed guessing b with probability $1/2$. This shows that the minimal probability with which any A_2 can outputs correct bit is $1/2$. We measure how well A is doing by the difference between $1/2$ and the probability in which A_2 can guess b correctly. Formally, we define the advantage of A as follows.

$$Adv_{A,\Pi}(k) \stackrel{\triangle}{=} |\Pr[(pk, sk) \leftarrow \mathcal{G}(1^k); (x_0, x_1, str) \leftarrow A_1(pk); b \leftarrow \{0,1\}; \\ y \leftarrow \mathcal{E}_{pk}(x_b); A_2(x_0, x_1, str, y) = b] - 1/2|$$

We consider two attack models, adaptive plaintext attack and adaptive ciphertext attack, in which an adversary can repeatedly use encryption and decryption oracle, respectively.

Definition 2 (IND-CPA). *Let $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ be an encryption scheme and let $A = (A_1, A_2)$ be an adversary who can use the encryption oracle. If*

$$Adv_{A,\Pi}(k) \geq \epsilon(k)$$

and A runs at most $t(k)$ steps, we say that A (t, ϵ)-breaks $\Pi(1^k)$ in the sense of IND-CPA.

If $Adv_{A,\Pi}(k)$ is negligible for any adversary A, we say that Π is secure in the sense of IND-CPA.

Definition 3 (IND-CCA2). *Let $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ be an encryption scheme and let $A = (A_1, A_2)$ be an adversary who can use the decryption oracle. If*

$$Adv_{A,\Pi}(k) \geq \epsilon(k)$$

and A runs at most $t(k)$ steps with at most q_D queries to decryption oracle, we say that A (t, ϵ, q_D)-breaks $\Pi(1^k)$ in the sense of IND-CCA2. If $Adv_{A,\Pi}(k)$ is negligible for any adversary A, we say that Π is secure in the sense of IND-CCA2.

Some literatures use an other notion IND-CCA1 in which an adversary $A = (A_1, A_2)$ can use the decryption oracle only in its find-stage: A_1. Since the secrecy in the sense of IND-CCA2 is stronger than that of IND-CCA1 (and IND-CPA) [14], we focus on the security in the sense of IND-CCA2.

2.2 Kurosawa et al's Public Key Cryptosystem [12]

Kurosawa et al. [12] showed a public key cryptosystem such that (i) inverting is equivalent to factoring $n = pq$, (ii) the decryption is unique and (iii) p and q are arbitrary prime numbers. Their scheme is described as follows.

Key generation algorithm $\mathcal{G}$: Choose two large primes p and q whose lengths are both $k/2$ bits. The secret key is a pair of p and q.
The public key is $pk = (N, c)$ such that

$$N = pq \text{ and } \left(\frac{c}{p}\right) = \left(\frac{c}{p}\right) = -1,$$

where $\left(\frac{c}{p}\right)$ denotes Legendre symbol.

Encryption algorithm $\mathcal{E}$: For a message $x \in Z_N^*$, let

$$Y_{pk}(x) = x + c/x \bmod N$$

$$U_{pk}(x) = \begin{cases} 0 & \text{if } (\frac{x}{N}) = 1 \\ 1 & \text{otherwise} \end{cases}$$

$$V_{pk}(x) = \begin{cases} 0 & \text{if } x < c/x \\ 1 & \text{otherwise,} \end{cases}$$

where $\left(\frac{x}{N}\right)$ denotes Jacobi symbol. Then the ciphertext is

$$Y_{pk}(x)||U_{pk}(x)||V_{pk}(x),$$

where $||$ denotes concatenation.

Decryption algorithm $\mathcal{D}$: Suppose that a receiver is given a ciphertext $y||u||v$. He first solves the following equations by using p and q.

$$x^2 - yx + c \equiv 0 \pmod{p},$$
$$x^2 - yx + c \equiv 0 \pmod{q}.$$

He then obtains four solutions x_1, x_2, x_3 and x_4 since $Y_{pk}(x)$ is a four-to-one function. Among the four roots, just one x_i satisfies $u = U_{pk}(x_i)$ and $v = V_{pk}(x_i)$. The receiver finally decides that such x_i is the message the sender sent.

3 Proposed Scheme for Long Messages

In this section, we present our first IND-CCA2 scheme whose security is equivalent to factoring $n = pq$ in the random oracle model, where p and q are arbitrary prime numbers. It works for long messages. We combine Kurosawa et al's scheme with Bellare and Rogaway's scheme of [1]. Note that the conversion of [1] requires a one-way permutation f while $Y_{pk}(x)$ of Sec. 2.2 is a four-to-one function.

Remember that k is a security parameter and $n(k)$ denotes the length of a plaintext. Let $k_0(k)$ be an integer valued function bounded by some polynomial on k. Let G be a mapping from k bit strings to n bit strings and let H be a mapping from $n + k$ bit strings to k_0 bit strings. They are treated as random oracles.

Then our scheme is described as follows.

Key generation algorithm $\mathcal{G}$: Choose two large primes p and q whose lengths are both $k/2$ bits. The secret key is a pair of p and q.
The public key is $pk = (N, c)$ such that

$$N = pq \text{ and } \left(\frac{c}{p}\right) = \left(\frac{c}{p}\right) = -1,$$

Encryption algorithm $\mathcal{E}$: Suppose that the input is a message x which is a n bit string. First, $\mathcal{E}$ chooses a random number $r \in Z_N^*$ such that $U_{pk}(r) = 0$ and $V_{pk}(r) = 0$, where

$$U_{pk}(r) = \begin{cases} 0 & \text{if } (\frac{r}{N}) = 1 \\ 1 & \text{otherwise} \end{cases}$$
$$V_{pk}(r) = \begin{cases} 0 & \text{if } r < c/r \\ 1 & \text{otherwise.} \end{cases}$$

Let

$$Y_{pk}(r) = r + c/r \bmod N.$$

The output which is the ciphertext of x is given as

$$x \oplus G(r)||Y_{pk}(r)||H(x||r).$$

Decryption algorithm $\mathcal{D}$: Suppose that an input is $z||y||s$. $\mathcal{D}$ first solves the following equations by using p and q.

$$r^2 - yr + c \equiv 0 \pmod{p},$$
$$r^2 - yr + c \equiv 0 \pmod{q}.$$

If there is no root, then it outputs $\perp$, which means that the input ciphertext is illegal. Otherwise, it obtains four solutions r_1, r_2, r_3 and r_4 since $Y_{pk}(r)$ is a four-to-one function. Among the four roots, just one r_i satisfies $U_{pk}(r_i) = 0$ and $V_{pk}(r_i) = 0$. let us denote such r_i by r without subscription. It computes

$$x = z \oplus G(r_i).$$

If $H(x||r_i) = s$ then it outputs x as the plaintext. Otherwise, outputs $\perp$.

See Fig.1.

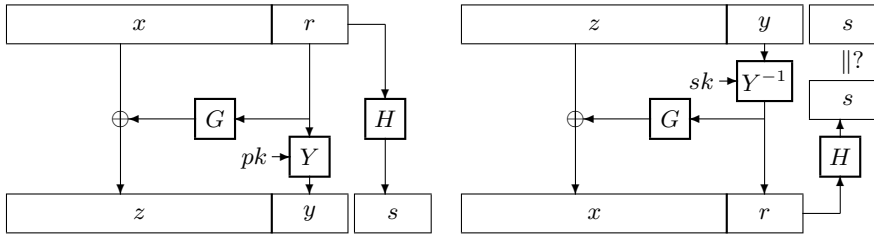

Fig. 1. Proposed scheme for long messages

Theorem 1. *Suppose that there exists an adversary A which $(t^{(A)}, \epsilon^{(A)}, q_D)$-breaks our first scheme in the sense of IND-CCA2 with at most q_G queries to G and at most q_H queries to H. Then there exists M which runs at most $t^{(M)}$ steps and can factor $N = pq$ with probability $\epsilon^{(M)}$, where*

$$t^{(M)} = t^{(A)} + (q_G + q_H + q_D)(T_Y(k) + \lambda(n+k)) + T_{Eu}(k)$$
$$\epsilon^{(M)} = \epsilon^{(A)}(1 - q_D 2^{-k_0})/2,$$

where $T_Y(k)$ denotes the time complexity of $Y_{pk}(x)$ and $T_{Eu}(k)$ denotes that of $\gcd(x, y)$.

The proof of Theorem 1 will be given in the final version.

4 Proposed Scheme for Short Messages

In this section, we present our second IND-CCA2 scheme whose security is equivalent to factoring $n = pq$ in the random oracle model, where p and q are arbitrary prime numbers. It is more efficient than our first scheme for short messages. We combine Kurosawa et al's scheme with Bellare and Rogaway's scheme of [2]. Note that [2] requires a one-way permutation f while $Y_{pk}(x)$ of Sec. 2.2 is four to one.

4.1 Scheme

Remember k is a security parameter. Let $k_0(\cdot)$ and $k_1(\cdot)$ be positive integer valued functions such that $k_0(k) + k_1(k) < k$ for all $k \geq 1$. Let $n(k) = k - k_0(k) - k_1(k)$ be the length of a plaintext.

Let G be a mapping from k_0 bit strings to $n + k_1$ bit strings and let H be a mapping from $n + k_1$ bit strings to k_0 bit strings. They are treaded as random oracles.

Then our scheme is described as follows.

Key generation algorithm: Choose two large primes p and q whose lengths are both $k/2$ bits. The secret key is a pair of p and q.

The public key is $pk = (N, c)$ such that

$$N = pq \text{ and } \left(\frac{c}{p}\right) = \left(\frac{c}{p}\right) = -1,$$

Encryption algorithm: Suppose that input message is $x \in \{0,1\}^n$. $\mathcal{E}$ at first chooses a random bit string r of length k_0, computes

$$s = (x||0^{k_1}) \oplus G(r), \quad t = r \oplus H(s), \quad w = s||t.$$

Let

$$U_{pk}(w) = \begin{cases} 0 & \text{if } (\frac{w}{N}) = 1 \\ 1 & \text{otherwise} \end{cases}$$

$$V_{pk}(w) = \begin{cases} 0 & \text{if } w < c/w \\ 1 & \text{otherwise.} \end{cases}$$

If $(U_{pk}(w), V_{pk}(w)) = (0,0)$, it outputs $y = Y_{pk}(w)$ as the ciphertext of x. Otherwise, it repeats choosing r untill w satisfies $U_{pk}(w) = V_{pk}(w) = 0$.

Decryption algorithm: Suppose that the input ciphertext is $y \in \{0,1\}^k$. $\mathcal{D}$ solves

$$\left.\begin{array}{ll} w^2 - yw + c \equiv 0 & (\text{mod } p) \\ w^2 - yw + c \equiv 0 & (\text{mod } q). \end{array}\right\} \tag{1}$$

If there is no solution it outputs $\perp$. Otherwise there are four solutions w_1, w_2, w_3 and w_4. Just one of them satisfies $U_{pk}(w) = V_{pk}(w) = 0$. It finds such w_i and sets $w_i = s||t$ where s and t are n and k_0 bits, respectively. It computes $x||z = s \oplus G(t \oplus H(s))$ where x is n bit string and z is the rest. If z is all zeros it outputs x, otherwise $\perp$.

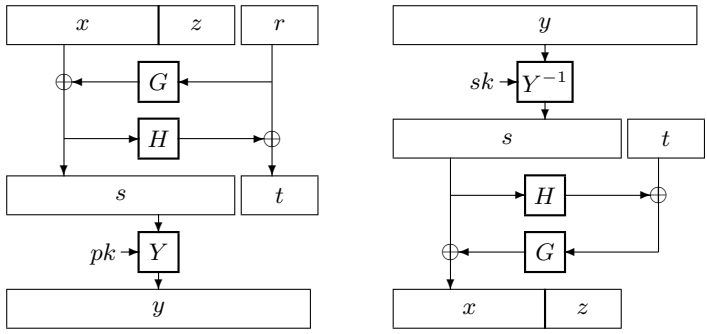

Fig. 2. Proposed scheme for short messages

See Fig.2.

Theorem 2. *Suppose that there exists an adversary A such that it* $(t^{(A)}, \epsilon^{(A)})$*-breaks our second scheme in the sense of IND-CPA with at most* q_G *queries to* G *and at most* q_H *queries to* H*. Then there exists* M *which runs at most* $t^{(M)}$ *steps and can factor* $N = pq$ *with probability* $\epsilon^{(M)}$*, where*

$$t^{(M)} = t^{(A)} + q_G\ q_H(T_Y(k) + \lambda k) + T_{Eu}(k)$$
$$\epsilon^{(M)} = \epsilon^{(A)}\ (1 - q_G 2^{-k_0} - q_H 2^{-(n+k_1)})/2 - q_G 2^{-k}$$

for some constant λ *where* $T_Y(k)$ *denotes the time complexity of* $Y_{pk}(x)$ *and* $T_{Eu}(k)$ *denotes that of* $\gcd(x, y)$*.*

Theorem 3. *Suppose that there exists an adversary B such that it* $(t^{(B)}, \epsilon^{(B)}, q_D)$*-breaks our second scheme in the sense of IND-CCA2 with at most* q_G *queries to* G *and at most* q_H *queries to* H*. Then there exists A which* $(t^{(A)}, \epsilon^{(A)})$*-breaks our second scheme in the sense of IND-CPA with at most* q_G *queries to* G *and at most* q_H *queries to* H*, where*

$$t^{(A)} = t^{(B)} + q_G q_H q_D(T_Y(k) + \lambda k)$$
$$\epsilon^{(A)} = \epsilon^{(B)}(1 - q_D 2^{-k_1}) - q_D 2^{-k_1}/2$$

for some constant λ*.*

The proof of Theorem 2 and 3 are given in the next subsections. From Theorem 2 and 3, we straightforward obtain the following corollary.

Corollary 1. *If there exists an adversary B such that it* $(t^{(B)}, \epsilon^{(B)}, q_D)$*-breaks our public encryption scheme in the sense of IND-CCA2 with at most* q_G *queries to* G *and at most* q_H *queries to* H*, then there exists* M *which runs at most* $t^{(M)}$ *steps and can factor* $N = pq$ *with probability* $\epsilon^{(M)}$*, where*

$$t^{(M)} = t^{(B)} + q_G q_H(q_D + 1)(T_Y(k) + \lambda k) + T_{Eu}(k)$$
$$\epsilon^{(M)} = \epsilon^{(B)}(1 - q_D 2^{-k_1})(1 - q_G 2^{-k_0} - q_H 2^{-(n+k_1)})/2$$
$$-q_D 2^{-k_1}(1 - q_G 2^{-k_0} - q_H 2^{-(n+k_1)})/4 - q_G 2^{-k}$$

4.2 Proof of Theorem 2

First, we construct M which factors its input N efficiently using $A = (A_1, A_2)$. This is done by finding a random m' with $(m'/N) = -1$ and then using the attacker to extract a preimage m of $y = f(m')$ inequivalent to m' by using the Bellare/Rogaway inversion algorithm. Since the latter does not use the permutation property, our method works.

Firstly, M randomly chooses c from Z_N^* which satisfies $(\frac{c}{N}) = 1$ and sets $pk = (N, c)$. M chooses randomly $\hat{m}$ from Z_N^* which satisfies $U_{pk}(\hat{m}) = 1$ and $V_{pk}(\hat{m}) = 0$ and computes $\hat{y} = Y_{pk}(\hat{m})$. Then it runs $A_1(pk)$. After this, A_1 will make G-queries and/or H-queries. To answer them M prepares two empty lists, G-list and H-list, and performs the following.

G-query for g: If G-list includes an entry $(g, G(g))$, return $G(g)$. Otherwise, for all entry $(h, H(h))$ in H-list compute

$$m = h||(g \oplus H(h)). \tag{2}$$

If there exists h such that

$$\hat{y} = Y_{pk}(m),\ U_{pk}(m) = V_{pk}(m) = 0, \tag{3}$$

then obtain p, q from $\gcd(N, m - \hat{m})$. Choose $G(g)$ which is a random bits string with length $n+k_1$, add an entry $(g, G(g))$ into G-list, and return $G(g)$.

H-query for h: If there is an entry $(h, H(h))$ in H-list, return $H(h)$. Otherwise, choose $H(h)$ which is a random bits string with length k_0 and add an entry $(h, H(h))$ into H-list. For all entry $(g, G(g))$ in G-list computes Eq.(2). If there exists g which satisfies Eq.(3), then obtain p, q from $\gcd(N, m - \hat{m})$. Finally, return $H(h)$.

Then A_1 will outputs (x_0, x_1, str).

Next, M chooses a random bit b, and runs $A_2(x_0, x_1, str, \hat{y})$. If A_2 makes G-queries and H-queries, M answers in the following ways.

G-query for g: If there exists an entry $(g, G(g))$ in G-list, return $G(g)$. Otherwise, compute Eq.(2) for all entry $(h, H(h))$ in H-list. If there exists h which satisfies Eq.(3), then obtain p, q from $\gcd(N, m - \hat{m})$ and return $G(g) = (x_b||0^{k_1}) \oplus h$. Otherwise, choose $G(g)$ which is a random bits string with length $n + k_1$, add an entry $(g, G(g))$ into G-list, and return $G(g)$.

H-query for h: If there exists an entry $(h, H(h))$, return $H(h)$. Otherwise, choose $H(h)$ which is a random bits string with length k_0 and add an entry $(h, H(h))$ into H-list. For all entry $(g, G(g))$ in G-list compute Eq.(2). If there exists g which satisfies Eq.(3), then obtain p, q from $\gcd(N, m - \hat{m})$. Finally return $H(h)$.

A_2 will outputs a bit b'. If M finds p and q then it outputs them, otherwise outputs $\perp$.

If (N, c) is a legal public key of the proposed scheme and there is a pair of (g, h) which satisfies Eq.(2) and (3), $\gcd(N, m - \hat{m})$ always presents p or q. For randomly chosen c, the probability with which (N, c) is a legal public key is $1/2$. Therefore we can estimate $\epsilon^{(M)}$ similarly with the proof of Theorem 3 in [2] without the factor $1/2$.

The running time of M is also similar with the proof of Theorem 3 in [2] except M needs additional time to compute p and q using gcd. When (N, c) is not a legal public key, it is possible that A never finish. In such a case M waits during $t^{(A)}$ and halts A.

4.3 Proof of Theorem 3

First, we show how to construct A_1/A_2 which uses B_1/B_2 as subroutine. This is done by showing that the chosen-ciphertext attacker is answered invalid plaintext with high probability for a decryption query unless the attacker has previously asked the random oracles G and H the queries corresponding to an encryption of the plaintext, the knowledge of which allows the recovery of the plaintext (i.e. the attacker is already aware of the plaintext of the decryption he is asking for).

(A_1 : find stage)

On input pk A_1 initializes G-list and H-list with empty and runs $B_1(pk)$. After this, B_1 will make G-query, H-query and decryption query. A_1 answers these queries in the following ways:

G-query for g: A_1 makes the same query to G oracle, obtains $G(g)$, gives it to B_1, and then adds an entry $(g, G(g))$ into G-list.

H-query for h: A_1 answers to it in similar way to the case of G-query and adds an entry into H-list.

The most troublesome task is to answer decryption queries for y. To simulate the decryption oracle without knowing the secret key, A_1 runs Dec-simulator :

Dec-simulator: On input y, for all entry $(h, H(h))$ and $(g, G(g))$ in H-list and G-list, compute $m = h||(g \oplus H(h))$ and check $y = Y_{pk}(m)$ and $U_{pk}(m) = V_{pk}(m) = 0$. If the check is not passed, return $\perp$ which means that y is an illegal ciphertext. Otherwise set $x||z = h \oplus G(g)$ where x is n bits and z is the rest. If $z = 0^{k_1}$ then return x, otherwise $\perp$.

Finally, B_1 outputs (x_0, x_1, str). A_1 outputs $(x_0, x_1, str||G\text{-list}||H\text{-list})$.

(A_2 : guess stage)

On input $(x_0, x_1, str||G\text{-list}||H\text{-list}, y)$ A_2 runs B_2 on input (x_0, x_1, str, y). After this, B_2 will make G-query, H-query and decryption query. A_2 answers these queries in the same way as A_1. Finally, A_2 outputs a bit b' or $\perp$ which B_2 outputs.

We will estimate the probability $Adv_{A,\Pi}(k)$. Let us denote the event "Dec-simulator can simulate decryption oracle correctly for all queries" with $success_D$. Clearly,

$$\Pr(b' = b) \geq \Pr(b' = b | success_D) \times \Pr(success_D).$$

From the definition of $Adv_{A,\Pi}$ and the hypothesis of the theorem,

$$\epsilon^{(B)} = |\Pr(b' = b|success_D) - 1/2|.$$

Wlog., we assume that

$$\epsilon^{(B)} = \Pr(b' = b|success_D) - 1/2.$$

Then

$$\begin{aligned}
\Pr(b' = b|success_D) &= 1/2 + \epsilon^{(B)}, \\
Adv_{A,\Pi}(k) &= |\Pr(b' = b) - 1/2| \\
&\geq |\Pr(b' = b|success_D) \times \Pr(success_D) - 1/2| \\
&= |(1/2 + \epsilon^{(B)})\Pr(success_D) - 1/2| \\
&= |\epsilon^{(B)}\Pr(success_D) - \frac{1 - \Pr(success_D)}{2}|
\end{aligned}$$

Now then, we need to estimate $\Pr(succeed_D)$. Assume that B_1 or B_2 makes a query for a ciphertext y. We consider three cases.

1. If there exists no m such that $y = Y_{pk}(m)$, then both Dec-simulator and decryption oracle will output $\perp$, that is, Dec-simulator can simulate correctly.
2. If there is $m = s||t$ such that $y = Y_{pk}(m)$ and $U_{pk}(m) = V_{pk}(m) = 0$, and there are $(g, G(g))$ and $(h, H(h))$ in G-list and H-list such that $h = s, g = t \oplus H(h)$, Dec-simulator returns a correct plaintext (or $\perp$) with probability one.
3. If there is $m = s||t$ such that $y = Y_{pk}(m)$ and $U_{pk}(m) = V_{pk}(m) = 0$, but there is no $(g, G(g))$ and/or $(h, H(h))$ in G-list and/or H-list such that $h = s, g = t \oplus H(h)$, Dec-simulator returns always outputs $\perp$. On the other hands, the legal decryption oracle will make H-query for $h = s$ and G-query for $g = t \oplus H(h)$, and check whether z is all zeros where $x||z = h \oplus G(g)$. Since $G(g)$ is random, $h \oplus G(g)$ is random. Then the decryption oracle outputs a message with probability 2^{-k_1}.

Consequently,

$$\begin{aligned}
\Pr(success_D \text{ for one query}) &\geq 1 - 2^{-k_1}, \\
\Pr(success_D) &\geq 1 - q_D 2^{-k_1},
\end{aligned}$$

and

$$Adv_{A,\Pi}(k) \geq \epsilon^{(B)}(1 - q_D 2^{-k_1}) - \frac{q_D 2^{-k_1}}{2}.$$

Finally, we will estimate the time complexity. To answer one decryption query A needs to compute $Y_{pk}(m)$ and to compare for every h and g. Then, it is clear that

$$t^{(A)}(k) = t^{(B)}(k) + q_G q_H q_D(T_Y(k) + \lambda k).$$

5 Discussion

In this section, we discuss about some variations of our schemes.

First, we can reduce the running time of encryption four times in average if we add two bits (U_{pk}, V_{pk}) to the ciphertexts. In this case, we do not have to choose random numbers such that $U_{pk} = V_{pk} = 0$ while the ciphertext becomes two bits longer.

Next, we can replace $Y_{pk}(r)$ with $r^2 \bmod N$. In this case, however, the ciphertexts are not uniquely decrypted with small probabilities such as

$$P_1 = 3/2^{k_0} \text{ in the scheme of Sec. 3,}$$

$$P_2 = 3/2^{k_1} \text{ in the scheme of Sec. 4.}$$

References

1. Bellare, M., Rogaway, P.: Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. Proc. of the 1st CCS (1993) 62–73
2. Bellare, M., Rogaway, P.: Optimal Asymmetric Encryption — How to encrypt with RSA. Proc. of Eurocrypt'94, Lecture Notes in Computer Science, **950** (1994) 92–111
3. Okamoto, T., Uchiyama, S.: A New Public-Key Cryptosystem as Secure as Factoring. Proc. of Eurocrypt'99, Lecture Notes in Computer Science, **1403** (1998) 308–318
4. Fujisaki, E., Okamoto, T.: How to Enhance the Security of Public-Key Encryption at Minimum Cost. Proc. of PKC'99, Lecture Notes in Computer Science, **1560** (1999) 53–68
5. Fujioka, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. Proc. of Crypto'99, Lecture Notes in Computer Science, **1666** (1999) 537–554
6. Pointcheval, D.: Chosen-Ciphertext Security for any One-Way Cryptosystem. Proc. of PKC 2000, Lecture Notes in Computer Science, **1807** (2000) 129–146
7. Paillier, P.: A Trapdoor Permutation Equivalent to Factoring. Proc. of PKC'99, Lecture Notes in Computer Science, **1560** (1999) 217–222
8. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. Proc. of Eurocrypt'99, Lecture Notes in Computer Science, **1592** (1999) 223–238
9. Paillier, P., Pointcheval, D.: Deficient Public-Key Cryptosystems Provably Secure Against Active Adversaries. Proc. of Asiacrypt'99, Lecture Notes in Computer Science, **1716** (1999) 165–179
10. Rabin, M.O.: Digitalized signatures and public key cryptosystems as intractable as factorization. MIT/LCS/TR-212, Technical Report MIT (1979)
11. Williams, H.C.: A modification of the RSA public-key encryption procedure. IEEE, IT, **IT-26** No.6 (1980) 726–729
12. Kurosawa, K., Itoh, T., Takeuchi, M.: Public key cryptosystem using a reciprocal number with the same intractability as factoring a large number. CRYPTOLOGIA, **XII** (1988) 225–233

13. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. Proc. of Crypto'98, Lecture Notes in Computer Science, **1462** (1998) 13–25
14. Bellare, M., Desai, A., Pointcheval, D., Rogaway, P.: Relations among notations of security for public key encryption schemes. Crypto'98, Lecture Notes in Computer Science, **1462** (1998) 26–45