Key Size Evaluation of Provably Secure RSA-based Encryption Schemes

Takako Nakashima and Tatsuaki Okamoto

Graduate School of Humanities and Sciences, Ochanomizu University[1]
and
NTT Information Sharing Platform Laboratories

(Received April 10, 2006)
(Revised July 18, 2006)

abstract[2]

Recently studies on asymmetric encryption schemes with proven security are very active. Among them, RSA-based schemes are practically most important. As such we have, the most popular OAEP, OAEP+, SAEP, SAEP+, REACT and so on. These schemes are proven semantically secure against adaptive chosen-ciphertext attack (denoted IND-CCA , which is the strongest security) under the random oracle model, on the RSA assumption. To assure the practical security theoretically, (for instance, to establish the security equivalent to the complexity of 1024 bits-long integer factorization), we must quantitatively estimate the exact key size needed for each scheme through the reduction. But this is not well studied until now. Therefore in this paper we estimate the exact key size needed by each scheme in order to theoretically guarantee security, based on the precise evaluation of the reduction efficiency. We compare the results among the schemes, and conclude that REACT is assured theoretical security with the shortest (almost minimum necessary) key size.

keywords Public-key cryptosystem, RSA-based encryption scheme, OAEP, semantic security, proven security, real security parameter

1 Introduction

The security of the public-key encryption schemes is classified to 9 levels by the combination of two aspects, one the achieved levels (one-wayness OW, semantic security IND, non-malleability NM) and the other the kinds of attack (chosen plaintext attack CPA, chosen-ciphertext attack CCA1, adaptive chosen-ciphertext attack CCA2). In [1] it was shown that the strongest security NM-CCA2 is equivalent to IND-CCA2. Thus an encryption scheme has the strongest security if it is shown to be IND-CCA2 secure. We say that such an encryption scheme has proven security.

The most fundamental RSA or Rabin encryption schemes are not IND-CCA2 secure because they are deterministic, hence never indistinguishable. The OAEP system [3] based on deterministic encryption function such as RSA, satisfies IND-CCA2 under the random oracle model [2]. By replacing the ideal random function in this model by a practical hash function, we can construct an encryption scheme with established security in a practical sense.

Besides OAEP, there proposed are schemes OAEP+, SAEP, SAEP+, REACT etc. which are also proven to have the strongest security IND-CCA2 under the random oracle model. It is not known, however, which length of the key theoretically assures their security in the practical use. Thus in this paper, we concretely evaluate the key lengths which assure the proven theoretical security to those practically important encryption schemes based on RSA, such as RSA – OAEP (that is, the RSA cryptosystem strengthened by OAEP) etc., and compare the results to see which scheme is most favorable.

[1] Currently The Japan Research Institute, Limited.

[2] Translator's note: This is the translation of master's thesis of the first author, who was my former student, written in Japanese and submitted on January 2002, under the substantial supervision of the second author. Although its essential part is already published under the same title in the Proceedings of SCIS2002 9A-3, it is also written in Japanese, and there are repeated demands for citation of this work in English literature. So I decided to translate it. Considering the general character of this journal, I preferred not the proceeding version but the original master thesis which contains detailed explanation of the employed notion and cited results, hence more readable to non-specialists. I am grateful to Dr. G. Hanaoka for encouraging this translation and giving valuable comments for the translated manuscript. (Akira Kaneko)

2 public-key encryption schemes

2.1 Definitions

A public-key encryption scheme consists of the following three algorithms:

- **Key generation algorithm** $\mathcal{K}$: On the input of 1^k, $\mathcal{K}$ returns $(\mathsf{pk}, \mathsf{sk})$. Here k is the security parameter, meaning the key length, $1^k := \overbrace{1 \cdots 1}^{k}$, pk is the public key, and sk the private key. This is a randomized algorithm, that is, the output changes randomly.
- **Encryption algorithm** $\mathcal{E}$: Given a plaintext $m \in \{0,1\}^*$ and a public key pk, $\mathcal{E}$ returns the encryption c of m. This may be deterministic or randomized.
- **Decryption algorithm** $\mathcal{D}$: Given a cipher text c and a private key sk, $\mathcal{D}$ returns either the plaintext m or "Reject". This latter implies that the ciphertext is invalid and there is no plaintext corresponding to it. This is a deterministic algorithm.

We assume that $\mathcal{K}$, $\mathcal{E}$, $\mathcal{D}$ all work in polynomial time.

2.1.1 Variety of attacks

The attack of the adversary $\mathcal{A}$ is divided to the two steps $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ corresponding to before or after she gets the ciphertext c^*, respectively. These are a pair of randomized algorithms. If both $\mathcal{A}_1$ and $\mathcal{A}_2$ are polynomial-time, $\mathcal{A}$ is said to be polynomial-time.

The kind of attack of the adversary $\mathcal{A}$ is classified to two big subclasses, passive and active attack. A passive attack tries to decrypt the target ciphertext c^* from itself and the public information. An active attack makes several queries to gain further information, for example, sending another ciphertext and obtaining its decryption.

The passive attack contains chosen-plaintext attack CPA. Here the adversary $\mathcal{A}$ can encrypt arbitrarily chosen plaintext. For public-key cryptosystem, this attack is inevitable because $\mathcal{E}$, pk are publicized. The active attack contains chosen-ciphertext attack CCA. Here the decryption oracle plays an important role. He answers to any query of ciphertext, returns its decrypted plaintext in an instant. According to the timing where $\mathcal{A}$ can use the oracle, CCA is divided to two subclasses.

Non-adaptive chosen-ciphertext attack, CCA1 admits $\mathcal{A}_1$ to make query to the decryption oracle, but does not to $\mathcal{A}_2$, namely, after $\mathcal{A}$ obtains the target ciphertext c^*, she cannot use the decryption oracle. On the contrary, adaptive chosen-ciphertext attack, CCA2 admits not only to $\mathcal{A}_1$ but also to $\mathcal{A}_2$ the query to the decryption oracle. But of course query of the target ciphertext c^* itself is prohibited. The last one is the strongest kind of attack.

Remark 2.1 It may be imagined that such an attack as CCA might be very artificial and non-realistic. But Bleichenbacher's attack [4] is of this kind. In that attack the adversary $\mathcal{A}$ can, for every query of the ciphertext c obtain a one-bit information of the decrypted result. If this attack could be generalized, $\mathcal{A}$ could obtain the whole decrypted result. Such a generalization is still non-realistic. Since CCA2 is, however, considered to be the strongest class of attacks, a cryptosystem which is proven secure against CCA2 may be secure against any type of attacks in the future.

2.1.2 Security achievement level

Security achievement level of encryption schemes is classified to secrecy and non-malleability. Secrecy means the extent of hiding transported information. Its level consists of one-wayness, partial one-wayness and indistinguishability. One-wayness is a basic requirement for any cryptosystem, and means the difficulty of inferring the plaintext from the ciphertext.

Definition 2.2 [One-wayness] A public-key encryption scheme is called one-way, if there is no adversary of polynomial-time computational power who cannot infer the plaintext from the given ciphertext with non-negligible probability. More precisely, it is said to be (t, ε)-OW if for any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ whose computation time is limited to t, the success probability $\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A})$ of the following is less than ε:

$$\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A}) = \Pr_{m \xleftarrow{R} \mathcal{M}}[(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k) : \mathcal{A}(\mathcal{E}_{\mathsf{pk}}(m)) \stackrel{?}{=} m].$$

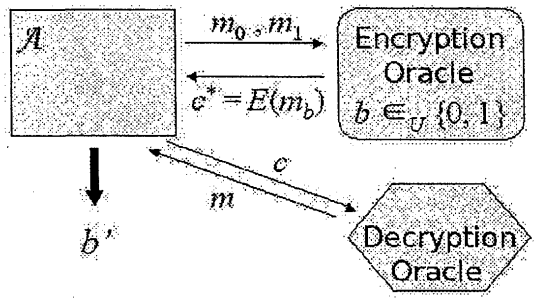

Figure 1: Semantic security against chosen-ciphertext attack.

A public-key encryption scheme is said to be partially one-way if it is difficult to infer some partial information (e.g. some bit(s)) of the plaintext from the ciphertext. It is said to be semantically secure if any bit or partial information of the plaintext is difficult to be inferred from the ciphertext. More precisely,

Definition 2.3 [Semantic security] A public-key encryption scheme in which it is difficult to know any partial information (other than the length) of the plaintext from the ciphertext, is said to be semantically secure or indistinguishable. More precisely, it is called (t, ε)-IND if for any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ whose computation time is limited to t, the following advantage $\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A})$ is less than ε:

$$\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A}) = 2 \times \Pr_{b \xleftarrow{R} \{0,1\}} \left[\begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k),\ (m_0, m_1, \mathsf{state}) \leftarrow \mathcal{A}_1(\mathsf{pk}), \\ c = \mathcal{E}_{\mathsf{pk}}(m_b) : \mathcal{A}_2(c, \mathsf{state}) \stackrel{?}{=} b \end{array} \right] - 1.$$

Here the length of m_0 and m_1 is the same and fixed by $\mathcal{A}_1$ from the space of plaintexts $\mathcal{M}$. "state" denotes the information possessed by $\mathcal{A}_1$, such as pk. It is passed to $\mathcal{A}_2$.

As is seen from this definition, deterministic encryption schemes, such as the simple RSA or Rabin encryption, do not satisfy the indistinguishability. In fact, in case of deterministic encryption scheme, for a pair of plaintexts m_i, $i = 0, 1$, their encryption $c_i = \mathcal{E}_{\mathsf{pk}}(m_i)$ are uniquely determined. Hence, by checking the ciphertext c equal to either of c_0, c_1, indistinguishability is obviously broken. Thus an encryption scheme with indistinguishability should be randomized.

In the sequel we shall abbreviate (t, ε)-OW to OW and (t, ε)-IND to IND.

The non-malleability expresses the rigidity against manipulation of ciphertext to result intended modification of the original plaintext, such as bit reversion. This formalizes the impossibility of modification of the ciphertext $c = \mathcal{E}_{\mathsf{pk}}(m)$ to produce a relation f, that is, to work out $\widetilde{c} = \mathcal{E}_{\mathsf{pk}}(\widetilde{m})$ satisfying $\widetilde{m} = f(m)$. We denote this property by NM. Non-malleability is known to be equivalent to the semantic security under the adaptive chosen-ciphertext attack (see below). We therefore omit to give its precise definition. They are not, however, equivalent in general.

2.1.3 Encryption schemes with proven security

As is seen from §2.1.1 and §2.1.2, the strongest security in public-key encryption schemes is the non-malleability against the adaptive chosen-ciphertext attack (NM-CCA2). This is shown to be equivalent to the semantic security against the adaptive chosen-ciphertext attack (IND-CCA2) in [1]. We call an encryption scheme proven secure under this setting "an encryption scheme with proven security". Since we only treat such schemes in this paper, we simply abbreviate CCA2 to CCA.

The definition of semantic security against adaptive chosen-ciphertext attack can be understood as in Figure 1. First, the adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ can query the decryption oracle at any time. $\mathcal{A}_1$ chooses two plaintexts m_0, m_1, and receives $c = \mathcal{E}(m_b)$ which is the encryption of either one (where $b \xleftarrow{R} \{0,1\}$). Since $\mathcal{E}$ is a randomized algorithm, one cannot know which of m_0, m_1 is c the encryption of. Thus with the aid of the decryption oracle $\mathcal{A}_2$ infers this, and as a result outputs $b' \in \{0,1\}$. If $b' = b$, $\mathcal{A}$ is said to have succeeded in inferring under the semantic security. The success probability $\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A})$ for that is

$$\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}) = \Pr_{b \xleftarrow{R} \{0,1\}} \left[\begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k),\ (m_0, m_1, \mathsf{state}) \leftarrow \mathcal{A}_1(\mathsf{pk}), \\ c = \mathcal{E}_{\mathsf{pk}}(m_b) : \mathcal{A}_2(c, \mathsf{state}) \stackrel{?}{=} b \end{array} \right].$$

Since the definition of the semantic security is simpler than that of the non-malleability, we often prove the security of an encryption scheme by showing IND-CCA.

2.2 Random oracle model

By assuming the existence of the random oracle model proposed in [2], an encryption scheme with proven security and yet effective was first realized by [3]. This is the so called OAEP (Optimal Asymmetric Encryption Padding). It was adopted in PKCS (Public-Key Cryptography Standards) #1 Version 2 as the standard way to use the RSA cryptosystem, and employed in SSL (Secure Sockets Layer). There were encryption schemes with proven security before OAEP, but they were ineffective and far from practical.

The random oracle model starts by assuming an ideal random function. First consider a hash function (hereafter abbreviated as Hash) used frequently in the real encryption schemes. Its outputs seem apparently random. But it runs by a uniquely defined deterministic algorithm whose design is publicized. Thus an input x produces a uniquely defined output $\mathsf{Hash}(x)$, hence never random information theoretically. This produces a difficulty in proving the security of an encryption scheme employing such hash functions. A random function is the idealization of hash functions which returns information theoretically random values. It cannot be realized. It consists of a big table containing the correspondence of the input and the output, any output value is chosen randomly. We assume the existence of a virtual server (oracle) which answers, on the query with an input value, the output value instantaneously. This is called a random oracle, and a model using such is called a random oracle model.

The actual hash function Hash is shared by all the participants including the adversary $\mathcal{A}$, by publicizing its design, thus enabling for all the participants to share the pair of input and output. On the other hand, in the random oracle model, anyone can query the oracle, which returns the same value to everybody on the same query. Thus all the participants can share the pair of input and output just as using Hash.

The proof of security of OAEP etc. is made under the random oracle model. Since the existence of random oracle model is non-realistic, practical encryption schemes are constructed by replacing the random oracle by practical hash functions. Then a scheme thus obtained is no more proven secure. But it can be considered as an approximation of an encryption scheme proven secure, thereby with a kind of assured security. In this way a random oracle model plays a role of bridge between the theory and practice in the study of security of cryptography.

3 RSA-based encryption schemes with proven security

In this section, we consider the RSA-based encryption schemes obtained by combining the RSA function with the five encryption schemes, OAEP, OAEP+, SAEP, SAEP+, REACT, which are proven secure under the random oracle model. We briefly give their descriptions and then evaluate their security.

In either of these encryption schemes, the key generation algorithm computes the two primes p, q and their product $N = pq$ (N is assumed to be k bits), and finally outputs (e, N) as the public key pk, (p, q) or d as the private key sk. Precise conditions on the key generation are assumed to be common to all RSA or Rabin functions.

The evaluation of the security is made by showing the semantic security against the adaptive chosen-ciphertext attack. Here we assume that the adversary makes the query q_G, q_H and $q_{H'}$ times to each random oracle G, H and H', respectively. Also, we assume that in the adaptive chosen ciphertext attack, the number of query made by the adversary to the decryption oracle is q_D. If the adversary $\mathcal{A}$ breaks the semantic security within the execution time t and the advantage $\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A}) \geq \varepsilon$, she is called an adversary breaking the semantic security in (t, ε)-CCA.

3.1 Reduction of the encryption scheme to the RSA problem

The encryption schemes considered in this paper are proven secure by showing that, on the assumption of the existence of an adversary $\mathcal{A}$ who breaks the semantic security by the adaptive chosen-ciphertext attack within time t and with the advantage at least ε, we can construct $\mathcal{B}$ who solves the RSA problem (or the factorization in case of encryption schemes based on the Rabin function) within time t' and with the success probability ε'. Here t', ε' can be expressed in terms of t, ε. Namely, if the RSA problem (or the factorization) cannot be solved within time t', and with the minimum success probability ε', then the

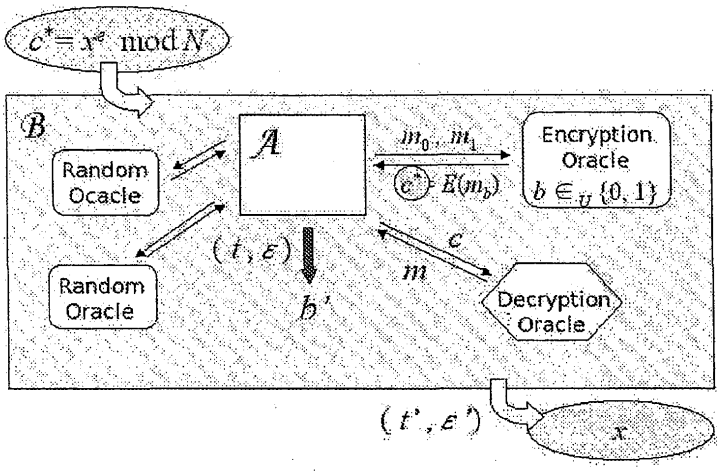

Figure 2: Reduction from cryptosystem to RSA problem.

semantic security of the corresponding encryption scheme cannot be broken within time t, and with the advantage not less than ε, thus the security in IND–CCA is shown.

Breaking semantic security means to find which of the plaintexts m_0, m_1 gives the object ciphertext c with a probability greater than $\frac{1}{2}$. An adversary $\mathcal{A}$ with the advantage ε is thought to have some information for the plaintext corresponding to c. $\mathcal{B}$, who wish to solve the RSA problem $c^\star = x^e \bmod N$ passes $c^\star$ as the target ciphertext to $\mathcal{A}$ who can break the semantic security. $\mathcal{B}$ himself simulates the random oracle or the decryption oracle to answer $\mathcal{A}$, and tries to obtain the solution x of the RSA problem of $c^\star$ from the response of $\mathcal{A}$. Then t' is the running time in which $\mathcal{B}$ executes these simulations and get the solution of the RSA problem, and ε' is its success probability. Figure 2 explains this.

3.2 RSA–OAEP

Let G and H be the following random functions, where $k = n + k_0 + k_1$.

$$G : \{0,1\}^{k_0} \longrightarrow \{0,1\}^{k-k_0}, \quad H : \{0,1\}^{k-k_0} \longrightarrow \{0,1\}^{k_0}.$$

Then the encryption and the decryption algorithms of this scheme are as follows:

$\mathcal{E}_{\mathsf{pk}}(m; r)$: Given a plaintext $m \in \{0,1\}^n$ and a random number $r \xleftarrow{R} \{0,1\}^{k_0}$, it computes

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

and outputs the ciphertext $c = (s \,||\, t)^e \bmod N$.

$\mathcal{D}_{\mathsf{sk}}(c)$: It computes $(s, t) = c^d \bmod N$, then

$$r = t \oplus H(s), \quad M = s \oplus G(r).$$

If $[M]_{k_1} = 0^{k_1}$ it returns $[M]^n$, otherwise returns "Reject". Here $[M]_{k_1}$ and $[M]^n$ denote the lower k_1 bits and the upper n bits of M, respectively.

Remark 3.1 If f is a general encryption function, the f–OAEP based thereon cannot be proven secure under the mere assumption of f being one-way ([9]). It is, however, proven secure for partial one-way function f as is shown in the following theorem ([7]). This is limited to deterministic encryption functions, and does not apply to randomized encryption functions such as ElGamal encryption.

Theorem 3.2 (Security of f–OAEP) *Assume that there exists an adversary $\mathcal{A}$ who breaks the semantic security of k bits f-OAEP by the q_G, q_H and q_D times of query against G, H and the decryption oracle, respectively, with (t, ε)-CCA. Then there exists $\mathcal{B}$ who can output the set S containing the solution*

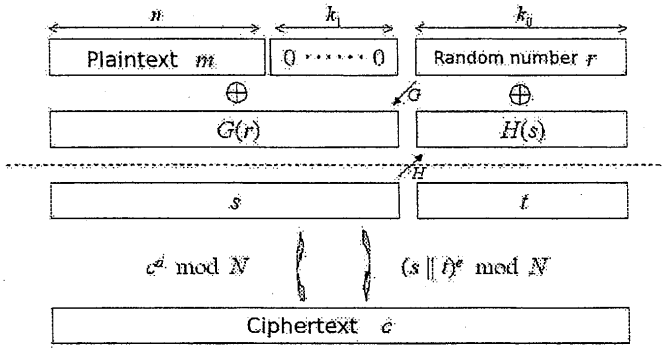

Figure 3: RSA–OAEP

of the partial one-wayness of the function f within time t'_s and with the success probability not less than ε'_s. Here ε'_s, t'_s are specified, respectively, by

$$\varepsilon'_s \geq \varepsilon - \frac{2q_Dq_G + q_D + q_G}{2^{k_0}} - \frac{2q_D}{2^{k_1}},$$

$$t'_s \leq t + q_Gq_HT_f + (q_G + q_H + q_D)T_{\mathsf{Hash}},$$

where T_f denotes the time needed for the computation of f, and T_{Hash} denotes the time complexity of the hash functions which we assume to be common and of $\mathcal{O}(k)$ (see §4.1 below).

In this way, OAEP is proven secure based on the partial one-wayness of f. Notice that the partial one-wayness of the RSA function is equivalent with its one-wayness by the property of the RSA function. Thus RSA–OAEP can be proven secure based on the one-wayness of the RSA function. The precise evaluation in this case is as follows:

Theorem 3.3 (Security of RSA–OAEP) ([7]) *Let $k > 2k_0$ and assume that there exists an adversary $\mathcal{A}$ who, by the q_G, q_H and q_D times query to G, H and the decryption oracle, respectively, breaks the semantic security of the k bits RSA–OAEP with (t, ε)-CCA. Then, there exists $\mathcal{B}$ who solves the RSA problem within time t' and with success probability not less than ε'. Here, ε', t' are specified, respectively, by.*

$$\varepsilon' \geq \varepsilon^2 - 2\varepsilon\Big(\frac{2q_Dq_G + q_D + q_G}{2^{k_0}} + \frac{2q_D}{2^{k_1}} + \frac{32}{2^{k-2k_0}}\Big),$$

$$t' \leq 2t + 2q_Gq_HT_{\mathsf{RSA}} + 2(q_G + q_H + q_D)T_{\mathsf{Hash}} + q_H^2T_L,$$

where T_{RSA} denotes the encryption time of the RSA function, T_L denotes the time complexity of the lattice reduction algorithm needed to reduce the one-wayness of the RSA function to its partial one-wayness and is estimated by $\mathcal{O}(k^3)$.

3.3 RSA–OAEP+

Let G, H', H be the following three random functions, respectively, where $k = n + k_0 + k_1$.

$$G : \{0,1\}^{k_0} \longrightarrow \{0,1\}^n, \quad H' : \{0,1\}^{k-k_1} \longrightarrow \{0,1\}^{k_1}, \quad H : \{0,1\}^{k-k_0} \longrightarrow \{0,1\}^{k_0}.$$

Then the encryption and the decryption algorithms of this scheme are as follows:

$\mathcal{E}_{\mathsf{pk}}(m; r)$: Given a plaintext $m \in \{0,1\}^n$ and a random number $r \xleftarrow{R} \{0,1\}^{k_0}$, it computes

$$s = (m \oplus G(r)) \,||\, H'(r \,||\, m), \quad t = r \oplus H(s),$$

and outputs the ciphertext $c = (s \,||\, t)^e \bmod N$.

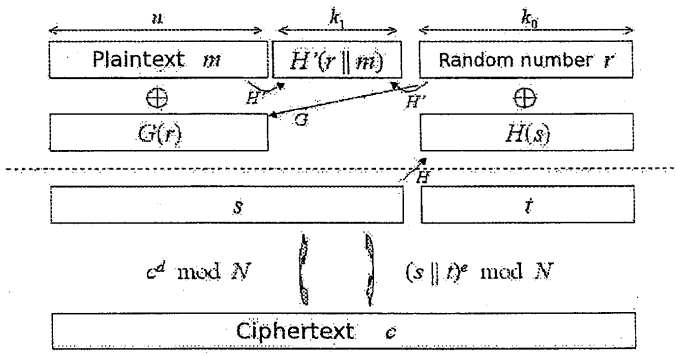

Figure 4: RSA–OAEP+

$\mathcal{D}_{\mathsf{sk}}(c)$: It computes $(s,t) = c^d \bmod N$, and

$$r = t \oplus H(s), \quad m = [s]^n \oplus G(r).$$

If $[s]_{k_1} = H'(r \,||\, m)$, then it returns m, otherwise returns "Reject".

Theorem 3.4 (Security of RSA–OAEP+) ([9]) *Assume that there exists an adversary $\mathcal{A}$ who, by the q_G, q_H, $q_{H'}$, q_D times query to G, H, H' and the decryption oracle, respectively, breaks the semantic security of the k bits RSA–OAEP+ with the advantage (t,ε)-CCA. Then there exists $\mathcal{B}$ who solves the RSA problem within time t' and with the success probability not less than ε'. Here, ε', t' are given, respectively, by*

$$\varepsilon' \geq \frac{\varepsilon}{2} - \frac{q_{H'} + q_D}{2^{k_1}} - \frac{(q_D+1)q_G}{2^{k_0}},$$

$$t' \leq t + q_G q_H T_{\mathsf{RSA}} + (q_G + q_H + q_{H'} + q_D) T_{\mathsf{Hash}},$$

T_{Hash} denoting the time complexity of the hash functions.

3.4 Rabin–SAEP

Let H be a random function as follows, where $k = n + k_0 + k_1$.

$$H : \{0,1\}^{k_0} \longrightarrow \{0,1\}^{k-k_0}.$$

The encryption and the decryption algorithms of this scheme are as follows:

$\mathcal{E}_{\mathsf{pk}}(m;r)$: It receives a plaintext $m \in \{0,1\}^n$ and a random number $r \xleftarrow{R} \{0,1\}^{k_0}$, computes

$$s = (m \,||\, 0^{k_1}) \oplus H(r),$$

then outputs the ciphertext $c = (s \,||\, r)^2 \bmod N$.

$\mathcal{D}_{\mathsf{sk}}(c)$: It calculates (s,r) from c by the same technique as the decryption of the Rabin encryption, then computes

$$M = s \oplus H(r).$$

If $[M]_{k_1} = 0^{k_1}$, then it returns $[M]^n$, otherwise returns "Reject".

SAEP is the abbreviation of Simple-OAEP as can be imagined from its description. In the proof of security of SAEP and SAEP+, Coppersmith's algorithm plays an important role. We therefore introduce here a theorem by Coppersmith.

Theorem 3.5 (Coppersmith) ([5]) *Let N be an integer, and let $f(x) \in \mathbf{Z}_N[x]$ be a monic polynomial of degree d. Then, there exists an efficient algorithm which allows to find all $x_0 \in \mathbf{Z}$ satisfying $f(x_0) = 0 \bmod N$ and $|x_0| < N^{1/d}$.*

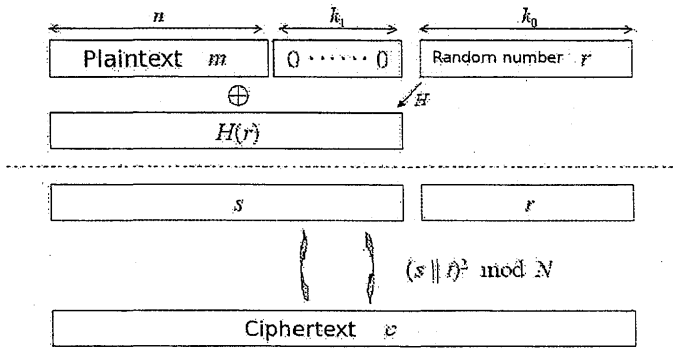

Figure 5: Rabin–SAEP

In the sequel we shall denote by $T_{\mathsf{C}}(k,d)$ the execution time of Coppersmith's algorithm to find the roots of a polynomial $f(x) \in \mathbf{Z}_N[x]$ of degree d, where $k = \log_2 N$ is the bit length of N. The complexity of this algorithm is not given by Coppersmith himself in a concrete form available to us. For the case $d = 2, 4$ which we need later, it was shown by Uchiyama [10] that $T_{\mathsf{C}}(k,d) = \mathcal{O}(k^3)$.

Theorem 3.6 (Security of Rabin–SAEP) ([5]) *Let $n < k/4$ and $n + k_1 < k/2$. Assume that there exists an adversary $\mathcal{A}$ who, by the q_H resp. q_D times query to H resp. the decryption oracle, breaks the semantic security of k bits Rabin–SAEP with the advantage (t,ε)-CCA. Then there exists $\mathcal{B}$ who can completely factorize k bits integer N within time t' with the success probability not less than ε'. Here ε', t' are given, respectively, by*

$$\varepsilon' \geq \frac{\varepsilon}{12}\Big(1 - \frac{2q_D}{2^{k_0}} - \frac{2q_D}{2^{k_1}}\Big),$$

$$t' \leq t + (q_D+1)q_H T_{\mathsf{C}}(N,2) + q_D T_{\mathsf{C}}(N,4) + (q_H+q_D)T_{\mathsf{Hash}},$$

T_{Hash} denoting the time complexity of the hash functions.

Remark 3.7 SAEP is not shown secure with general RSA function. It is shown as secure as Rabin–SAEP only when small e's such as $e = 3$ are used. In case $e = 3$, it can be proved secure for $n < k/9$ and $n + k_1 < k/3$. For typical k about the size of N in RSA function, the proof cannot apply because of this restriction on the length n of the plaintext, which makes the secure use of this encryption scheme difficult.

3.5 RSA–SAEP+

Let G and H be the two random functions as follows, where $k = n + k_0 + k_1$.

$$G : \{0,1\}^{k-k_1} \longrightarrow \{0,1\}^{k_1}, \quad H : \{0,1\}^{k_0} \longrightarrow \{0,1\}^{k-k_0}.$$

The encryption and the decryption algorithms of this scheme are as follows:

$\mathcal{E}_{\mathsf{pk}}(m;r)$: Given a plaintext $m \in \{0,1\}^n$ and a random number $r \xleftarrow{R} \{0,1\}^{k_0}$, it computes

$$s = (m \,||\, G(m \,||\, r)) \oplus H(r),$$

and outputs the ciphertext $c = (s \,||\, r)^e \bmod N$.

$\mathcal{D}_{\mathsf{sk}}(c)$: It computes $(s,r) = c^d \bmod N$, then

$$M = s \oplus H(r).$$

If $[M]_{k_1} = G([M]^n \,||\, r)$, it returns $[M]^n$, otherwise returns "Reject".

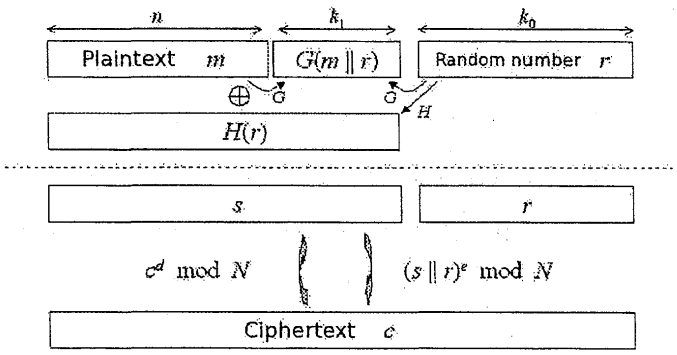

Figure 6: RSA–SAEP+

Remark 3.8 In the proof of security of RSA–SAEP+, things go like the case of RSA–OAEP. That is, first the security of f–SAEP+ is proven for the partially one-way function f, then so is the security of RSA–SAEP+ employing the one-wayness of the RSA function and its property.

Theorem 3.9 (Security of f–SAEP+) ([5]) *Assume that there exists an adversary $\mathcal{A}$ who, by q_G, q_H and q_D times query to G, H and the decryption oracle, respectively, breaks the semantic security of k bits f–SAEP+ with (t, ε)-CCA. Then there exists $\mathcal{B}$ who can output a set S containing the solution of the partial one-wayness of the function f within time t'_s and with the success probability not less than ε'_s. Here ε'_s, t'_s are given by*

$$\varepsilon'_s \geq \frac{\varepsilon}{2} - \frac{1}{2}\left(\frac{q_D}{2^{k_0}} + \frac{q_D}{2^{k_1}}\right),$$

$$t'_s \leq t + q_G T_f + (q_G + q_H + q_D) T_{\mathsf{Hash}}.$$

Theorem 3.10 (Security of RSA–SAEP+) ([5]) *Let $n + k_1 < k/2$. Assume that there exists an adversary $\mathcal{A}$ who, by the q_G, q_H, q_D times query to G, H and the decryption oracle, respectively, breaks the semantic security of the k bits RSA–SAEP+ with the setting (t, ε)-CCA. Then there exists $\mathcal{B}$ who can solve the RSA problem within time t' and with the success probability not less than ε'. Here ε', t' are estimated, respectively, as*

$$\varepsilon' \geq \frac{\varepsilon^2}{4} - \frac{\varepsilon}{2}\left(\frac{q_D}{2^{k_0}} + \frac{q_D}{2^{k_1}} + \frac{32}{2^{k-2k_0}}\right),$$

$$t' \leq 2t + 2q_G T_{\mathsf{RSA}} + 2(q_G + q_H + q_D) T_{\mathsf{Hash}} + q_H^2 T_L,$$

where T_L denoets the same quantity as in Theorem 3.3.

3.6 Rabin–SAEP+

The technique of encryption and decryption is the same as RSA–SAEP+: The only difference in case of Rabin–SAEP+ is that Rabin function is employed in $\mathcal{E}$, $\mathcal{D}$.

Theorem 3.11 (Security of Rabin–SAEP+) ([5]) *Let $n + k_1 < k/2$. Assume that there exists an adversary $\mathcal{A}$ who, by the q_G, q_H, q_D times query to G, H and the decryption oracle, respectively, breaks the semantic security of the k bits Rabin–SAEP+ with (t, ε)-CCA. Then there exists $\mathcal{B}$ who can completely factorize a k bits integer N within time t' and with the success probability not less than ε'. Here, ε', t' are given, respectively, by*

$$\varepsilon' \geq \frac{\varepsilon}{12} - \frac{q_D}{2^{k_0}} - \frac{q_D}{2^{k_1}},$$

$$t' \leq t + q_G T_{\mathsf{Rabin}} + q_H T_{\mathsf{C}}(N, 2) + (q_G + q_H + q_D) T_{\mathsf{Hash}}.$$

3.7 RSA–REACT

Let G, H be the following random functions:

$$G : \mathbf{Z}_N \longrightarrow \{0,1\}^{k_1}, \quad H : \{0,1\}^* \longrightarrow \{0,1\}^{k_2}.$$

The encryption and the decryption algorithms of this scheme are as follows:

$\mathcal{E}_{\mathsf{pk}}(m;r)$: Given a plaintext $m \in \{0,1\}^{k_1}$ and a random number $r \stackrel{R}{\leftarrow} \mathbf{Z}_N$, it computes

$$c_1 = r^e \bmod N, \quad c_2 = G(r) \oplus m,$$

then makes

$$c_3 = H(r, m, c_1, c_2),$$

and outputs $C = (c_1, c_2, c_3)$ as the ciphertext.

$\mathcal{D}_{\mathsf{sk}}(c_1, c_2, c_3)$: It first computes $r = c_1^d \bmod N$, then gets

$$m = G(r) \oplus c_2.$$

If $c_3 = H(r, m, c_1, c_2)$, then it returns m, and otherwise returns "Reject".

Theorem 3.12 (Security of RSA–REACT) ([8]) *Assume that there exists an adversary $\mathcal{A}$ who, by the q_G, q_H, q_D times query to the G, H and the decryption oracle, respectively, breaks the k bits RSA–REACT with (t,ε)-CCA. Then there exists $\mathcal{B}$ which solves k bits RSA problem within time t' and with success probability not less than ε'. Here, ε', t' are given, respectively, by*

$$\varepsilon' \geq \varepsilon - \frac{q_D}{2^{k_2}},$$

$$t' \leq t + q_G T_{\mathsf{XOR}} + (q_G + q_H) T_{\mathsf{RSA}} + (q_G + q_H + q_D) T_{\mathsf{Hash}},$$

T_{XOR} being the time needed for the XOR operations.

Remark 3.13 REACT (Rapid Enhanced-security Asymmetric Cryptosystem Transform) proposed in [8] applies not only to the deterministic encryption functions such as RSA, but also to the randomized encryption functions such as ElGamal. Also, REACT admits the use of the symmetric cryptosystem for $\mathcal{E}^{\mathsf{sym}}$ with $K = G(r)$ as the secret key to compute $c_2 = \mathcal{E}_K^{\mathsf{sym}}(m)$. ($T_{\mathsf{XOR}}$ in the above evaluation then corresponds to the time to compute c_2.) The evaluation of the reduction from RSA–REACT to RSA problem when $\mathcal{E}^{\mathsf{sym}}$ is employed becomes a little more complicated (see [8]). Here for the sake of simplicity, we adopted XOR operation.

4 Evaluation of the efficiency of reduction

Now we evaluate the efficiency of reduction in each security proof introduced in the preceding section.

4.1 Basic strategy of evaluation

Let $T = \frac{t}{\varepsilon}$ be the computational complexity of the adversary $\mathcal{A}$ attacking each cryptosystem with (t,ε)-CCA. Let $T' = \frac{t'}{\varepsilon'}$ be the computational complexity of $\mathcal{B}$ solving the RSA problem (or computing the factorization) within time t' and with success probability ε'. By finding relation between T' and T, we can evaluate the efficiency of the reduction from each encryption scheme to RSA or factorization problem. Before doing this, we give several remarks.

In the evaluations below, we let T_{Hash}, T_{RSA}, T_{Rabin} and $T_{\mathsf{C}}(k,d)$ denote the computation time for hash operation, RSA encryption, Rabin encryption, and Coppersmith algorithm, respectively, for k bits data. Precisely speaking, T_{Hash} depends also on the output length. But here we assume that it depends only on the input length. These are respectively [3],

$$T_{\mathsf{Hash}} = \mathcal{O}(k), \quad T_{\mathsf{RSA}} = \mathcal{O}(k^2), \quad T_{\mathsf{Rabin}} = \mathcal{O}(k^2), \quad T_{\mathsf{C}}(N,d) = \mathcal{O}(k^3).$$

[3] Translator's note: The RSA encryption key e is assumed to be short following the practical setting.

Here $d = 2, 4$, and k is of size N. We also denote by T_{Dec} the decryption time. This is

$$T_{\mathsf{Dec}} = \mathcal{O}(k^3)$$

common to all the schemes. These are employed to find the ratio of the computation time. In the sequel we *assume* that

$$T_{\mathsf{Hash}} = k, \quad T_{\mathsf{XOR}} = k, \quad T_{\mathsf{RSA}} = k^2,$$
$$T_{\mathsf{Rabin}} = k^2, \quad T_{\mathsf{C}}(N, d) = k^3, \quad T_{\mathsf{Dec}} = k^3, \quad T_L = k^3.$$

The oracles employed in the random oracle model all return answers instantaneously. In practice, however, there is no such oracle, as practical random oracles are realized by hash functions such as SHA. Thus we consider that

$$T_G = T_H = T_{H'} = T_{\mathsf{Hash}}.$$

On the other hand, we understand that the decryption oracle is realized through the decryption by the decryptor with the secret key. Hence, the computation time T_D of the decryption oracle is considered to be

$$T_D = T_{\mathsf{Dec}}.$$

In the evaluation of efficiency, we take the worst case computation time of reduction from each encryption scheme to RSA or factorization problem. This means that each evaluation formula for t' is the worst. For example, for RSA–OAEP, it means that $t' \leq 2t + 2q_G q_H T_{\mathsf{RSA}} + 2(q_G + q_H + q_D)k + q_H^2 k^3$ becomes biggest, that is, the term $q_H^2 k^3$ which influences t' most, becomes biggest. Since this is the case when q_H becomes biggest, this amounts to assuming that the adversary $\mathcal{A}$ continuously queries the H oracle during the time t. On the other hand, we assume that the query to G oracle is minimal, that is, only once. Namely, in RSA–OAEP, we assume that

$$t = q_H T_H = q_H T_{\mathsf{Hash}}.$$

Thus we can evaluate as

$$q_H = \frac{t}{T_{\mathsf{Hash}}}, \quad q_G = 1.$$

Discussing similarly for the other encryption schemes, we can obtain the following estimates:

$$\begin{aligned}
\text{RSA–OAEP+} &: q_G = q_H = \frac{t}{2T_{\mathsf{Hash}}}, \quad q_{H'} = 1, \\
\text{Rabin–SAEP} &: q_H = \frac{t}{T_{\mathsf{Hash}}}, \\
\text{RSA–SAEP+} &: q_H = \frac{t}{T_{\mathsf{Hash}}}, \quad q_G = 1, \\
\text{Rabin–SAEP+} &: q_H = \frac{t}{T_{\mathsf{Hash}}}, \quad q_G = 1, \\
\text{RSA–REACT} &: q_G = q_H = \frac{t}{2T_{\mathsf{Hash}}}.
\end{aligned}$$

On the other hand, we may assume that $\mathcal{A}$ can continuously query the decryption oracle during the time t, hence, $t = q_D T_D = q_D T_{\mathsf{Dec}}$, that is,

$$q_D = \frac{t}{T_{\mathsf{Dec}}}.$$

The values of the parameters k_0, k_1, k_2 are assumed to be sufficiently bigger than $\log_2 q_D q_G$ etc., hence $\frac{q_D q_G}{2^{k_0}}$, for example, is small enough to be neglected. Also, the terms of $\mathcal{O}(k)$ such as $(q_G + q_H + q_D)k$ in the evaluation of reduction time from the encryption schemes to RSA or factorization problems, are small enough in comparison with terms of $\mathcal{O}(k^2)$ or of $\mathcal{O}(k^3)$ like $q_G T_{\mathsf{RSA}}$ or $q_H T_{\mathsf{C}}(k, 2)$. Hence they are neglected in the evaluation below.

Following these strategies, we now actually seek relations between $T = \frac{t}{\epsilon}$ and $T' = \frac{t'}{\epsilon'}$ for each encryption scheme.

4.2 Evaluation of efficiency of reduction for RSA–OAEP

$$
\begin{aligned}
T' &\le \frac{2t + 2q_G q_H T_{\mathsf{RSA}} + 2(q_G+q_H+q_D)k + q_H^2 k^3}{\varepsilon^2 - 2\varepsilon\left(\frac{2q_D q_G + q_D + q_G}{2^{k_0}} + \frac{2q_D}{2^{k_1}} + \frac{32}{2^{k-2k_0}}\right)} \\
&= \frac{1}{\varepsilon^2}\{2t + 2q_G q_H T_{\mathsf{RSA}} + q_H^2 k^3\} \\
&= \frac{1}{\varepsilon^2}\Big\{2t + 2\cdot 1\cdot \frac{t}{T_{\mathsf{Hash}}} T_{\mathsf{RSA}} + \Big(\frac{t}{T_{\mathsf{Hash}}}\Big)^2 k^3\Big\} \\
&= \frac{2}{\varepsilon}\Big(1 + \frac{T_{\mathsf{RSA}}}{T_{\mathsf{Hash}}}\Big)T + \frac{k^3}{T_{\mathsf{Hash}}^2} T^2 \\
&= \frac{2}{\varepsilon}(1+k)T + kT^2 = kT^2 + \frac{2}{\varepsilon}(1+k)T.
\end{aligned}
$$

4.3 Evaluation of efficiency of reduction of RSA–OAEP+

$$
\begin{aligned}
T' &\le \frac{t + q_G q_H T_{\mathsf{RSA}} + (q_G + q_H + q_{H'} + q_D)k}{\frac{\varepsilon}{2} - \frac{q_{H'}+q_D}{2^{k_1}} - \frac{(q_D+1)q_G}{2^{k_0}}} = \frac{2}{\varepsilon}(t + q_G q_H T_{\mathsf{RSA}}) \\
&= \frac{2}{\varepsilon}\Big\{t + \Big(\frac{t}{2T_{\mathsf{Hash}}}\Big)^2 T_{\mathsf{RSA}}\Big\} = 2T + \frac{t\cdot T_{\mathsf{RSA}}}{2T_{\mathsf{Hash}}^2} T = 2T + \frac{T_{\mathsf{RSA}}}{2T_{\mathsf{Hash}}^2}\varepsilon T^2 \\
&= 2T + \frac{1}{2}\varepsilon T^2 = \frac{1}{2}\varepsilon T^2 + 2T.
\end{aligned}
$$

4.4 Evaluation of efficiency of reduction of Rabin–SAEP

Since we assumed that $T_{\mathsf{C}}(k,d) = \mathcal{O}(k^3) = k^3$ irrespective of the value of d, we shall write T_{C} for both $T_{\mathsf{C}}(k,2)$ and $T_{\mathsf{C}}(k,4)$.

$$
\begin{aligned}
T' &\le \frac{t + (q_D+1)q_H T_{\mathsf{C}}(k,2) + q_D T_{\mathsf{C}}(k,4) + (q_H+q_D)k}{\frac{\varepsilon}{12}\left(1 - \frac{2q_D}{2^{k_0}} - \frac{2q_D}{2^{k_1}}\right)} \\
&= \frac{12}{\varepsilon}\{t + (q_D+1)q_H T_{\mathsf{C}} + q_D T_{\mathsf{C}}\} = \frac{12}{\varepsilon}\{t + q_D q_H T_{\mathsf{C}} + (q_H+q_D)T_{\mathsf{C}}\} \\
&= \frac{12}{\varepsilon}\Big\{t + \frac{t}{T_{\mathsf{Dec}}}\cdot\frac{t}{T_{\mathsf{Hash}}} T_{\mathsf{C}} + \Big(\frac{t}{T_{\mathsf{Hash}}} + \frac{t}{T_{\mathsf{Dec}}}\Big)T_C\Big\} \\
&= 12\Big\{1 + \frac{t\cdot T_{\mathsf{C}}}{T_{\mathsf{Dec}}\cdot T_{\mathsf{Hash}}} + \Big(\frac{1}{T_{\mathsf{Hash}}} + \frac{1}{T_{\mathsf{Dec}}}\Big)T_{\mathsf{C}}\Big\}T \\
&= 12\Big\{1 + \Big(\frac{1}{T_{\mathsf{Hash}}} + \frac{1}{T_{\mathsf{Dec}}}\Big)T_{\mathsf{C}}\Big\}T + \frac{12T_{\mathsf{C}}}{T_{\mathsf{Dec}}\cdot T_{\mathsf{Hash}}}\varepsilon T^2 \\
&= 12(1 + k^2 + 1)T + \frac{12}{k}\varepsilon T^2 = \frac{12}{k}\varepsilon T^2 + 12(k^2+2)T.
\end{aligned}
$$

4.5 Evaluation of efficiency of reduction of RSA–SAEP+

$$
\begin{aligned}
T' &\le \frac{2t + 2q_G T_{\mathsf{RSA}} + 2(q_G + q_H + q_D)k + q_H^2 k^3}{\frac{\varepsilon^2}{4} - \frac{\varepsilon}{2}\left(\frac{q_D}{2^{k_0}} + \frac{q_D}{2^{k_1}} + \frac{32}{2^{k-2k_0}}\right)} \\
&= \frac{4}{\varepsilon^2}\{2t + 2q_G T_{\mathsf{RSA}} + q_H^2 k^3\} \\
&= \frac{4}{\varepsilon^2}\Big\{2t + 2\cdot 1\cdot T_{\mathsf{RSA}} + \Big(\frac{t}{T_{\mathsf{Hash}}}\Big)^2 k^3\Big\} \\
&= \frac{8}{\varepsilon}T + \frac{8}{\varepsilon^2}T_{\mathsf{RSA}} + \frac{4k^3}{T_{\mathsf{Hash}}^2}T^2 = \frac{8}{\varepsilon}T + \frac{8}{\varepsilon^2}k^2 + 4kT^2 = 4kT^2 + \frac{8}{\varepsilon}T + \frac{8k^2}{\varepsilon^2}.
\end{aligned}
$$

4.6 Evaluation of efficiency of reduction of Rabin–SAEP+

$$
\begin{aligned}
T' &\leq \frac{t + q_G T_{\mathsf{Rabin}} + q_H T_{\mathsf{C}}(k,2) + (q_G + q_H + q_D)k}{\frac{\varepsilon}{12} - \frac{q_D}{2^{k_0}} - \frac{q_D}{2^{k_1}}} \\
&= \frac{12}{\varepsilon}(t + q_G T_{\mathsf{Rabin}} + q_H T_{\mathsf{C}}) \\
&= \frac{12}{\varepsilon}\Big(t + 1 \cdot T_{\mathsf{Rabin}} + \frac{t}{T_{\mathsf{Hash}}} T_{\mathsf{C}}\Big) = 12\Big(1 + \frac{T_{\mathsf{C}}}{T_{\mathsf{Hash}}}\Big)T + \frac{12}{\varepsilon} T_{\mathsf{Rabin}} \\
&= 12(1+k^2)T + \frac{12}{\varepsilon}k^2 = 12(k^2+1)T + \frac{12k^2}{\varepsilon}.
\end{aligned}
$$

4.7 Evaluation of efficiency of reduction of RSA–REACT

$$
\begin{aligned}
T' &\leq \frac{t + q_G T_{\mathsf{XOR}} + (q_G + q_H) T_{\mathsf{RSA}} + (q_G + q_H + q_D)k}{\varepsilon - \frac{q_D}{2^{k_2}}} \\
&= \frac{1}{\varepsilon}\{t + (q_G + q_H) T_{\mathsf{RSA}}\} = \frac{1}{\varepsilon}\Big(t + \frac{t}{T_{\mathsf{Hash}}} T_{\mathsf{RSA}}\Big) = \Big(1 + \frac{T_{\mathsf{RSA}}}{T_{\mathsf{Hash}}}\Big)T \\
&= (1+k)T = (k+1)T.
\end{aligned}
$$

5 Evaluation of key length

In this section, under the setting that all encryption schemes have the same level of security, we evaluate their key lengths, that is, the size of the RSA or factorization problems to which they are reduced.

5.1 Fundamental strategy for evaluation

If the computation time T of the adversary $\mathcal{A}$ defined in §4 is practically intractable, the encryption scheme will be secure. We evaluate the length of the secure key under this hypothesis. Here we choose as this computation time T the one corresponding to the factorization of 1024 or 2048 bits integers, which are now admitted as practically infeasible. The computation time necessary to factorize the integer N of the public key of RSA cryptosystem by the general number field sieve method, now thought to be the fastest, is

$$\exp\Big\{\Big(\frac{64}{9}\Big)^{1/3}(\log N)^{1/3}(\log\log N)^{2/3}\Big\}.$$

Thus we apply this to $N = 2^{1024}$, 2^{2048}, and let the computation time T represent this.

By substituting this T to the evaluation formulas deduced in §4, we find the computation time T' to solve the RSA or factorization problem reduced from each encryption scheme, and then consider the intractability of the computation time T' thus obtained. Here we are assuming that there is no means other than factorization to solve the RSA problem, and seek the bit length of the integer of which the computation time for the factorization is equivalent with T'.

The size k thus obtained is the key length minimum necessary for each encryption scheme to have the same level of security as the factorization of 1024 or 2048 bits integers (that is, the computation time T). In fact, if $k^\star > k$, then assuming the existence of the adversary $\mathcal{A}$ who breaks the encryption scheme of key length $k^\star$ by computation time T. it would break the $k^\star$ bits RSA problem by computation time T'. Then by the assumption that there is no other way than factorization to solve the RSA problem, it would follow that $\mathcal{A}$ succeeds in factoring $k^\star$ bits integer with computation time T'. Since, however, T' is the computation time for the factorization of size k integers, the factorization of size $k^\star > k$ would need computation time bigger than T'. This is a contradiction. Thus if $k^\star > k$, the existence of such adversary $\mathcal{A}$ is disproven, and the security of the encryption scheme is proven.

5.2 Results

We computed the necessary key length of the RSA or factorization problem reduced from each encryption scheme to have the same security as the 1024, 2048 bits integer factorization, according to the evaluation of the efficiency of the reduction made in §4. These are listed respectively in table 1 or 2.

value of ε	1	2^{-60}	2^{-80}	2^{-120}	2^{-160}
RSA–OAEP	6221	6221	6223	9443	14285
RSA–OAEP+	5179	1874	1190	1052	1052
Rabin–SAEP	4647	1857	1856	1856	1856
RSA–SAEP+	6387	6387	6477	15568	30000
Rabin–SAEP+	1856	1856	1856	3536	6447
RSA–REACT	1330	1330	1330	1330	1330

Table 1: Key length necessary to have security equivalent to 1024 bits factorization, with respect to our reduction estimation.

value of ε	1	2^{-60}	2^{-80}	2^{-120}	2^{-160}
RSA–OAEP	12452	12452	12452	12985	18761
RSA–OAEP+	10725	5196	3867	2092	2090
Rabin–SAEP	9773	4663	3470	3351	3351
RSA–SAEP+	12704	12704	12704	15568	30000
Rabin–SAEP+	3351	3351	3351	3545	6447
RSA–REACT	2547	2547	2547	2547	2547

Table 2: Key length necessary to have security equivalent to 2048 bits factorization, with respect to our reduction estimation.

6 dependency of the key length on ε

In §5, for preassigned complexity T. we found the key length for each encryption scheme against which there is no adversary $\mathcal{A}$ breaking its semantic security with (t, ε)-CCA. But the evaluated key length varies with ε. Here we consider this problem.

As is seen from table 1, table 2, we can classify the dependency of the key length on the value of ε needed by each encryption scheme. The first group consists of those which need longer key as ε becomes smaller, like RSA–OAEP, RSA–SAEP+, Rabin–SAEP+. The second group, on the contrary, consists of those for which the necessary key length becomes shorter as ε becomes smaller, such as RSA–OAEP+, Rabin–SAEP. The third group consists of those for which the necessary key length does not depend on ε, like RSA–REACT. Here we consider the first two groups in which the key length varies depending on ε.

6.1 The case where necessary key length decreases with ε

We first consider RSA–OAEP+, Rabin–SAEP to which the key length assuring the security becomes shorter as ε becomes smaller. Note that in general, the transformation of (t, ε)-CCA to $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA is not necessarily guaranteed. In fact, if we transform an adversary $\mathcal{A}$ which breaks the semantic security with (t, ε)-CCA to $(\frac{t}{c}, \varepsilon^*)$-CCA by simply stopping with execution time $\frac{t}{c}$, then by the assumption that $\mathcal{A}$ is a blackbox, it may be possible that $\varepsilon^* < \frac{\varepsilon}{c}$. Now that the transformation from (t, ε)-CCA to $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA is unexpectable, in order to assure the non-existence of the adversary who breaks with all (t, ε)-CCA by the computation time $T = \frac{t}{\varepsilon}$, we need to adopt the key length in case it is the longest ($\varepsilon = 1$). For example, for RSA–OAEP+ in table 1 with the key length 1190 we can assure the non-existence of the adversary who breaks the semantic security with $(\frac{T}{2^{80}}, \frac{1}{2^{80}})$-CCA, but cannot with $(\frac{T}{2^{60}}, \frac{1}{2^{60}})$-CCA or with $(T, 1)$-CCA. On the other hand, with the 5179 bits key length we can assure the non-existence of the adversary who breaks all $(\frac{T}{c}, \frac{1}{c})$-CCA ($c \geq 1$) with the same computation time T.

6.2 The case where the key length increases as ε becomes smaller

We next consider those encryption schemes, such as RSA–OAEP, RSA–SAEP+, Rabin–SAEP+, for which the key length assuring the security increases when ε becomes smaller. Note that the c times repetition of $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA adversary obviously gives rise to (t, ε)-CCA adversary. Thus, for these encryption schemes, when ε is small, we may give better (shorter) estimate for the key length to disprove the existence of an adversary breaking semantic security with $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA, by repeating the reduction of each encryption scheme to the RSA problem c times, than the directly evaluated key length necessary to disprove the existence of an adversary who breaks the semantic security with (t, ε)-CCA.

6.2.1 Examples

Now we explain the above principle by the example of RSA–OAEP. First, let $\varepsilon \ll 1$, and consider an adversary who repeats c times ($c > 1$) the reduction from RSA–OAEP to the set S containing the solution of the partial one-wayness of the RSA problem. We shall show that in this case, repeating c times the reduction from $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA is really more efficient than doing the reduction only once from (t, ε)-CCA.

The reduction from RSA–OAEP to the RSA problem, as mentioned in §3.2, is executed as follows: First, using the adversary $\mathcal{A}$ who breaks (t, ε)-CCAwith success probability $\varepsilon'_{\mathsf{s}}$, and time t'_{s}. we reduce the problem to the one finding the set S containing the solution $s^\star$ of the partial one-wayness of the RSA problem (where $chc = (s^\star \,||\, t^\star)^e$), and then using the property of RSA function (random reducibility), we end up by finding the solution of the RSA problem $(s^\star \,||\, t^\star)$. Here,

$$\varepsilon'_{\mathsf{s}} \geq \varepsilon - \frac{2q_D q_G + q_D + q_G}{2^{k_0}} - \frac{2q_D}{2^{k_1}},$$

$$t'_{\mathsf{s}} \leq t + q_G q_H T_{\mathsf{RSA}} + (q_G + q_H + q_D)k.$$

As mentioned in §4, we assume that $\frac{q_D q_G}{2^{k_0}}$ etc. are small enough compared to ε, hence can be neglected. Also, $(q_G + q_H + q_D)k = \mathcal{O}(k)$ is small enough compared to $q_G q_H T_{\mathsf{RSA}} = \mathcal{O}(k^2)$, and negligible. Then, we can write as

$$\varepsilon'_{\mathsf{s}} \geq \varepsilon, \quad t'_{\mathsf{s}} \leq t + q_G q_H T_{\mathsf{RSA}}.$$

From this, we see that if the adversary $\mathcal{A}$ breaks the semantic security of RSA–OAEP with $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA, then the success probability $\varepsilon'^{[1]}_{\mathsf{s}}$ and the time $t'^{[1]}_{\mathsf{s}}$ with which $\mathcal{B}$ finds the set $S^{[1]}$, is evaluated as

$$\varepsilon'^{[1]}_{\mathsf{s}} \geq \frac{\varepsilon}{c}, \quad t'^{[1]}_{\mathsf{s}} \leq \frac{t}{c} + q_G q_H T_{\mathsf{RSA}}.$$

Then, $\mathcal{B}$, by repeating c times the reduction to this set $S^{[1]}$, will obtain the set $S^{[c]}$ with success probability $\varepsilon'^{[c]}_{\mathsf{s}}$ and the computation time $t'^{[c]}_{\mathsf{s}}$ as follows:

$$\varepsilon'^{[c]}_{\mathsf{s}} = c \cdot \varepsilon'^{[1]}_{\mathsf{s}} \geq \varepsilon, \quad t'^{[c]}_{\mathsf{s}} = c \cdot t'^{[1]}_{\mathsf{s}} \leq t + c \cdot q_G q_H T_{\mathsf{RSA}}.$$

Here, too, we are assuming the worst case of the reduction, hence, the adversary $\mathcal{A}$ adopts the strategy with which the reduction time to $\mathcal{B}$ becomes the worst. In order to make $t'^{[c]}_{\mathsf{s}}$ the biggest (that is, to make $q_G q_H$ the biggest), $\mathcal{A}$ is assumed to make as many queries as possible to the oracles G, H within time $\frac{t}{c}$. Thus in view of $\frac{t}{c} = q_G T_G + q_H T_H = (q_G + q_H) T_{\mathsf{Hash}}$ we assume that $q_G = q_H = \frac{t}{2cT_{\mathsf{Hash}}}$. Then the time $t'^{[c]}_{\mathsf{s}}$ can be written as

$$t'^{[c]}_{\mathsf{s}} = t + c \cdot \left(\frac{t}{2cT_{\mathsf{Hash}}}\right)^2 T_{\mathsf{RSA}} = t + \frac{t^2}{4c} \cdot \frac{T_{\mathsf{RSA}}}{T^2_{\mathsf{Hash}}}.$$

On the other hand, the success probability $\widetilde{\varepsilon}'_{\mathsf{s}}$ and the computation time $\widetilde{t}'_{\mathsf{s}}$ of $\mathcal{B}$ who, employing the adversary $\widetilde{\mathcal{A}}$ of (t, ε)-CCA, finds the set $\widetilde{S}$, are evaluated by

$$\widetilde{\varepsilon}'_{\mathsf{s}} = \varepsilon, \quad \widetilde{t}'_{\mathsf{s}} \leq t + q_G q_H T_{\mathsf{RSA}}.$$

As before, the adversary $\widetilde{\mathcal{A}}$ of (t, ε)-CCA, in order to make the reduction time $\widetilde{t}'_{\mathsf{s}}$ biggest, takes, the strategy $t = q_G T_G + q_H T_H = (q_G + q_H) T_{\mathsf{Hash}}$. Namely, we assume that $q_G = q_H = \frac{t}{2T_{\mathsf{Hash}}}$. Therefore, this time $\widetilde{t}'_{\mathsf{s}}$ can be written as

$$\widetilde{t}'_{\mathsf{s}} = t + \left(\frac{t}{2T_{\mathsf{Hash}}}\right)^2 T_{\mathsf{RSA}} = t + \frac{t^2}{4} \cdot \frac{T_{\mathsf{RSA}}}{T^2_{\mathsf{Hash}}}.$$

Let now $\mathsf{Red}^{[c]}$ denote the c-times repetition of the reduction of finding the set containing the solution of the partial one-wayness employing $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA. and $\widetilde{\mathsf{Red}}$ the same with only one reduction employing (t, ε)-CCA. If we compare the reduction efficiency of $\mathsf{Red}^{[c]}$ with $\widetilde{\mathsf{Red}}$, we have $\varepsilon'^{[c]}_{\mathsf{s}} = \widetilde{\varepsilon}'_{\mathsf{s}}$, but $t'^{[c]}_{\mathsf{s}} < \widetilde{t}'_{\mathsf{s}}$. Hence $\mathsf{Red}^{[c]}$ is more efficient than $\widetilde{\mathsf{Red}}$ in reduction. Since the efficiency of reduction from the set containing the solution of partial one-wayness to the solution of RSA problem is the same, $\mathsf{Red}^{[c]}$ has better efficiency for $\mathcal{B}$ to solve the RSA problem. Thus the key length necessary to theoretically assure

value of ε	1	2^{-60}	2^{-80}	2^{-120}	2^{-160}
RSA–OAEP	6221	< 6221	< 6221	6221	6221

Table 3: Key length necessary to assure the same security as the 1024 bits integer factorization with respect to our reduction estimation, in case of repetition of reduction.

	1024 bits	2048 bits
RSA–OAEP	6221	12452
RSA–OAEP+	5179	10725
Rabin–SAEP	4647	9773
RSA–SAEP+	6387	12704
Rabin–SAEP+	1856	3351
RSA–REACT	1330	2547
Case where reduction is ideal	1024	2048

Table 4: recommended key length to assure the same security as 1024/2048 bits integer factorization, with respect to our reduction estiamtion

the security of RSA–OAEP, can be made shorter in the case of c times repetition of the reduction to the set containing solution of partial one-wayness employing $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA, than in the case of once reduction employing (t, ε)-CCA. Therefore the key length of RSA–OAEP in table 1 can be transformed as in the table 3.

As is seen from table 3, the key length needed to assure the security of RSA–OAEP is the longest when $\varepsilon = 1$. By the same argument as in the case of §6.1 where the key length becomes shorter as ε becomes smaller, this implies that in order to guarantee the non-existence of any adversary who breaks the semantic security of RSA–OAEP of computational complexity T with (t, ε)-CCA, we should adopt the longest key length corresponding to $\varepsilon = 1$. The same argument applies to RSA–SAEP+, Rabin–SAEP+, to conclude that we can adopt the key length for the case $\varepsilon = 1$.

Remark 6.1 Here in order to let $t_s'^{[c]}$, $\widetilde{t'}_s$ have the worst values, we discussed assuming that $q_G = q_H = \frac{t}{2\alpha T_{\mathsf{Hash}}}$ ($\alpha = c, 1$, respectively). Comparing with $q_H = \frac{t}{aT_{\mathsf{Hash}}}$, $q_G = 1$ given in §4, however, the reduction time differs by at most a small constant multiplier, and the key length by at most several bits.

From the above discussion, we conclude that in order to disprove the existence, for any complexity $T = \frac{t}{\varepsilon}$ and with any (t, ε)-CCA, of an adversary $\mathcal{A}$ who breaks the semantic security, the necessary key length can be estimated from the case $\varepsilon = 1$ for any scheme.

7 The key length which assures security

From the discussion of §6, we deduce the following table 4 of recommended key length which assures the same strength of security as the 1024/2048 bits integer factorization for each encryption scheme.

The value of key length given here is the minimum necessary assured length for each encryption scheme to have the same strength of security as the 1024/2048 bits integer factorization. Therefore one may adopt keys longer than those values given in table 4.

Also, if the reduction of the encryption scheme proven secure is ideal, the computational complexity T of the adversary who breaks the semantic security of the original encryption scheme, is equal to the computational complexity T' of solving the reduced RSA problem.

Furthermore, by using the evaluation of reduction efficiency found in §4, we can conversely evaluate the security level of each encryption scheme when the 1024/2048 bits key length is employed. Then by setting the computational complexity of the k bits RSA problems T' to that of the factorization of 1024/2048 bits integers, and by finding T from there, we can see the difficulty of breaking each encryption scheme measured by the bit length of the integer factorization equivalent to it. Table 5 shows this result.

adopted key length	1024 bits	2048 ビット
RSA–OAEP	169	338
RSA–OAEP+	225	428
Rabin–SAEP	256	481
RSA–SAEP+	160	324
Rabin–SAEP+	499	1152
RSA–REACT	774	1625
case of ideal reduction	1024	2048

Table 5: Bit length of integer factorization equivalent to the complexity of encryption scheme with 1024/2048 bits key, with respect to our reduction estimation.

8 conclusion

As can be seen from table 4, among the RSA-based encryption schemes proven secure, the key lengths necessary to supply theoretically assured security much differ depending on its efficiency of reduction. Especially, RSA–OAEP has worst efficiency, and far from its naming "optimal". On the other hand, RSA–REACT has the most efficient reduction and assures the same security with the minimum key length.

Our evaluation of security of each encryption scheme is very natural, so will not be able to be improved drastically.

We remark that when we use $e = 3$ in RSA function, the efficiency of the reduction becomes much better compared to the case of general e. For example, RSA–SAEP+ has bad efficiency of reduction for general e, but for $e = 3$ it has the same efficiency with Rabin–SAEP+, hence, security is assured with the same short key length. Also, according to [9], RSA–OAEP+ allows very efficient reduction for $e = 3$.

Acknowledgement

We thank Doctor Shigenori Uchiyama of NTT Information Sharing Platform Laboratories for suggesting the complexity of the Coppersmith algorithm. We also thank Professor David Pointcheval of ENS–CNRS who gave us precious comments on the fundamental strategy of this research. The first author thanks Professor Akira Kaneko for constant advice during the preparation of the master thesis.

Appendix

A $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA and (t, ε)-CCA

When ε is small, simple c times ($c > 1$) repetition of the adversary $\mathcal{A}$ of $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA never gives (t, ε)-CCA. This is the reason why we evaluated in detail the c times repetition of the reduction of $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA to the partial one-wayness in §6.2. In this section, we evaluate the advantage of the c-times repetition of $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA for comparison.

An adversary $\mathcal{A}$ of $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA, breaks the semantic security of the encryption scheme within time $\frac{t}{c}$, and advantage at least $\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A}) = \frac{\varepsilon}{c}$, that is, she can succeed to correctly answer the problem of determining which of m_0, m_1 is the plaintext for c with success probability $\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A})$ as follows:

$$\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}) = \frac{1}{2} + \frac{\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A})}{2} = \frac{1}{2} + \frac{\varepsilon}{2c}.$$

If we repeat this c times, we understand that the final answer is the majority of the c answers $b'^c \in \{0,1\}^c$ given by $\mathcal{A}$ of $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA. In case the frequency of $b' = 0$ and $b' = 1$ is the same ($\frac{c}{2}$ each), we determine the final answer to be 0 or 1, each with probability $\frac{1}{2}$. Let $\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}^c)$ be the success

probability of this. It can be written as

$$\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}^c) = \sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c}{i} \cdot \left(\frac{1}{2} + \frac{\varepsilon}{2c}\right)^i \cdot \left(\frac{1}{2} - \frac{\varepsilon}{2c}\right)^{c-i}.$$

Here w_i denotes

$$w_i = \begin{cases} \frac{1}{2} & (i = \frac{c}{2}) \\ 1 & (i \neq \frac{c}{2}) \end{cases}$$

Assume now that ε is small enough, and c is even. Set $\delta = \frac{\varepsilon}{c}$ for brevity, and assume that it is also small enough. Then, we can write as

$$\begin{aligned}\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}^c) &= \sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c}{i} \cdot \left(\frac{1}{2} + \frac{\delta}{2}\right)^i \cdot \left(\frac{1}{2} - \frac{\delta}{2}\right)^{c-i} = \sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c}{i} \cdot \frac{(1+\delta)^i}{2^i} \cdot \frac{(1-\delta)^{c-i}}{2^{c-i}} \\ &\sim \frac{1}{2^c} \sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c}{i} \cdot (1+i\delta)\{1-(c-i)\delta\} \sim \frac{1}{2^c} \sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c}{i} \cdot \{1+(2i-c)\delta\}.\end{aligned}$$

In view of

$$\sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c}{i} = \frac{1}{2} \sum_{i=0}^{c} \binom{c}{i} = \frac{2^c}{2},$$

we deduce from this that

$$\begin{aligned}\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}^c) &\sim \frac{1}{2^c}\Big\{ \sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c}{i} + \sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c}{i} \cdot (2i-c)\delta \Big\} \\ &= \frac{1}{2^c}\Big\{ \frac{2^c}{2} + \sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c}{i} \cdot (2i-c)\delta \Big\} = \frac{1}{2} + \frac{\delta}{2^c} \sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c}{i} \cdot (2i-c).\end{aligned}$$

Here, we have

$$\begin{aligned}\binom{c}{i} \cdot (2i-c) &= 2i \cdot \frac{c(c-1)\cdots(c-i+1)}{i!} - c \cdot \binom{c}{i} = 2 \cdot \frac{c(c-1)\cdots(c-i+1)}{(i-1)!} - c \cdot \binom{c}{i} \\ &= 2c \cdot \binom{c-1}{i-1} - c \cdot \binom{c}{i} = c \cdot \left(2 \cdot \binom{c-1}{i-1} - \binom{c}{i}\right),\end{aligned}$$

whence, we can rewrite the above as

$$\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}^c) \sim \frac{1}{2} + \frac{c\delta}{2^c} \sum_{i=\frac{c}{2}}^{c} w_i \cdot \left(2 \cdot \binom{c-1}{i-1} - \binom{c}{i}\right) = \frac{1}{2} + \frac{c\delta}{2^c}\Big\{ 2 \sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c-1}{i-1} - \sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c}{i} \Big\}.$$

Now we have

$$\begin{aligned}\sum_{i=\frac{c}{2}}^{c} w_i \cdot \binom{c-1}{i-1} &= \binom{c-1}{c-1} + \binom{c-1}{c-2} + \cdots + \binom{c-1}{(\frac{c}{2}+1)-1} + \frac{1}{2} \cdot \binom{c-1}{\frac{c}{2}-1} \\ &= \sum_{i=\frac{c}{2}}^{c-1} \binom{c-1}{i} + \frac{1}{2} \cdot \binom{c-1}{\frac{c}{2}-1},\end{aligned}$$

and also,

$$\sum_{i=\frac{c}{2}}^{c-1} \binom{c-1}{i} = \frac{1}{2} \sum_{i=0}^{c-1} \binom{c-1}{i} = \frac{2^{c-1}}{2}.$$

Thus we obtain

$$\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}^c) \sim \frac{1}{2} + \frac{c\delta}{2^c}\Big\{ 2 \cdot \Big(\frac{2^{c-1}}{2} + \frac{\binom{c-1}{\frac{c}{2}-1}}{2} \Big) - \frac{2^c}{2} \Big\} = \frac{1}{2} + \frac{c\delta}{2^c} \cdot \binom{c-1}{\frac{c}{2}-1}.$$

In view of the asymptotic value given in §B below, the success probability of A^c, the c times repetition of $\mathcal{A}$ of $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA, in the infer in semantic security, $\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}^c)$, is finally given by

$$\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}^c) \lesssim \frac{1}{2} + \frac{c\delta}{2^c} \cdot \frac{2^{c-1}}{\sqrt{c}} = \frac{1}{2} + \frac{\varepsilon}{2\sqrt{c}}.$$

Thus the advantage $\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A}^c)$ of $\mathcal{A}^c$ thereby is

$$\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A}^c) \lesssim 2 \cdot \left(\frac{1}{2} + \frac{\varepsilon}{2\sqrt{c}}\right) - 1 = \frac{\varepsilon}{\sqrt{c}}.$$

Since the whole execution time is equal to $c \cdot \frac{t}{c} = t$, the c times repetition of $\mathcal{A}$ of $(\frac{t}{c}, \frac{\varepsilon}{c})$-CCA is translated at most into $(t, \frac{\varepsilon}{\sqrt{c}})$-CCA.

B Approximate value of $\binom{n-1}{\frac{n}{2}-1}$

In this section, we deduce the asymptotic form of $\binom{n-1}{\frac{n}{2}-1}$ used in §A. In view of Stirling's formula

$$n! \sim \sqrt{2\pi}\, e^{-n} n^{n+\frac{1}{2}},$$

and of the fact $(1 - \frac{1}{n})^n \sim \frac{1}{e}$, we have

$$\begin{aligned}
\binom{n-1}{\frac{n}{2}-1} &= \frac{(n-1)!}{(\frac{n}{2}-1)!\frac{n}{2}!} = \frac{\sqrt{2\pi}\, e^{-(n-1)}(n-1)^{n-\frac{1}{2}}}{\sqrt{2\pi}\, e^{-\frac{n}{2}+1}(\frac{n}{2}-1)^{\frac{n}{2}-\frac{1}{2}} \cdot \sqrt{2\pi}\, e^{-\frac{n}{2}}(\frac{n}{2})^{\frac{n}{2}+\frac{1}{2}}} \\
&= \frac{2^n}{\sqrt{2\pi}} \cdot \frac{(n-1)^n}{(n-2)^{\frac{n}{2}} n^{\frac{n}{2}+\frac{1}{2}}} \cdot \sqrt{\frac{n-2}{n-1}} = \frac{2^n}{\sqrt{2\pi n}} \cdot \frac{(1-\frac{1}{n})^n}{(1-\frac{2}{n})^{\frac{n}{2}}} \cdot \sqrt{\frac{n-2}{n-1}} \\
&\sim \frac{2^n}{\sqrt{2\pi n}} = \frac{1}{\sqrt{\pi/2}} \frac{2^{n-1}}{\sqrt{n}} < \frac{2^{n-1}}{\sqrt{n}}.
\end{aligned}$$

References

[1] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among Notions of Security for Public-Key Encryption Schemes. *Proc. of Crypto'98*, LNCS **1462**, pp.26–45. Springer-Verlag, Berlin, 1998.

[2] M. Bellare and P. Rogaway. Random Oracles Are Practical: a Paradigm for Designing Efficient Protocols. *Proc. of the 1st CCS*, pp.62–73. ACM Press, New York, 1993.

[3] M. Bellare and P. Rogaway. Optimal Asymmetric Encryption – How to Encrypt with RSA. *Proc. of Eurocrypt'94*, LNCS **950**, pp.92–111. Springer-Verlag, Berlin, 1995.

[4] D. Bleichenbacher. A Chosen Ciphertext Attack against Protocols based on the RSA Encryption Standard PKCS #1. *Crypto'98*, LNCS **1462**, pp.1–12, Springer-Verlag, Berlin, 1998.

[5] D. Boneh. Simplified OAEP for the RSA and Rabin Functions. *Proc. of Crypto2001*, LNCS **2139**, pp.275–291, Springer-Verlag, Berlin, 2001.

[6] D. Coppersmith. Finding a Small Root of a Univariate Modular Equation. *Eurocrypt'96*, LNCS **1070**, pp.155–165, Springer-Verlag, Berlin, 1996.

[7] E. Fujisaki, T. Okamoto, D. Pointcheval and J. Stern. RSA–OAEP is Secure under the RSA Assumption. *Proc. of Crypto2001*, LNCS **2139**, pp.260–274, Springer-Verlag, Berlin, 2001.

[8] T. Okamoto and D. Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. *Proc. of CT - RSA2001*, LNCS **2020**, pp.159–175, Springer-Verlag, Berlin, 2001.

[9] V. Shoup. OAEP Reconsidered. *Proc. of Crypto2001*, LNCS **2139**, pp.239–259, Springer-Verlag, Berlin, 2001.

[10] S. Uchiyama. *Personal correspondence.* December 2001.