

# Research Reports on Mathematical and Computing Sciences

The Semantic Security and the Non-Malleability  
with the Randomness Revealed  
for Public-Key Encryption

Ryotaro Hayashi and Keisuke Tanaka

February 2007, C-241

Department of  
Mathematical and  
Computing Sciences  
Tokyo Institute of Technology

SERIES **C**: Computer Science

# The Semantic Security and the Non-Malleability with the Randomness Revealed for Public-Key Encryption

Ryotaro Hayashi and Keisuke Tanaka \*

Dept. of Mathematical and Computing Sciences  
Tokyo Institute of Technology  
W8-55, 2-12-1 Ookayama Meguro-ku, Tokyo 152-8552, Japan  
{hayashi9, keisuke}@is.titech.ac.jp

February 22, 2007

## Abstract

We consider the situation for public-key encryption that the adversary knows the randomness which was used to compute the ciphertext. In some practical scenarios, there is a possibility that the randomness is revealed. For example, the randomness used to make a ciphertext may be stored in insecure memory, or the pseudorandom generator may be corrupted. We first formalize the security notion on this situation as “the semantic security with the randomness revealed” and “the non-malleability with the randomness revealed.” In addition to the formalization, we focus on the scheme, 3-round OAEP, and prove that this scheme satisfies our security notion.

**Keywords:** public-key encryption, randomness, security, semantic security, non-malleability.

## 1 Introduction

The classical security requirement of public-key encryption schemes is that it provides the privacy of the encrypted data. Popular formalizations such as, indistinguishability and non-malleability under either the chosen plaintext attack or the adaptive chosen ciphertext attack are directed at capturing various data-privacy requirements. The widely admitted appropriate security level for public-key encryption is the indistinguishability against the adaptive chosen ciphertext attack (IND-CCA). Up until now, many public-key encryption schemes have been proposed which are secure in the sense of IND-CCA.

In EUROCRYPT 2001, Canetti and Krawczyk [5] proposed a security model of key-exchange protocols. The key-exchange protocols allow two parties who share no secret information (or share short passwords) to compute a session key via public communication. In the protocol, two parties usually use two kinds of secret information, that is, the long-term secret key and the ephemeral key which is state-specific secret information used within each session. The ephemeral key contains the randomness which is used in the session, for example. The Canetti–Krawczyk model gives the adversary the power to reveal the ephemeral key (session-state reveal query) without revealing the long-term secret key. Session-state reveal queries are motivated by practical scenarios, such as if the state-specific secret information is stored in insecure memory or if the random-number generator is corrupted. In both of these cases, as a result of the specific vulnerability, the adversary might be able to gain access to the ephemeral data. In these scenarios, for example, the (standard) signed

---

\*Supported in part by NTT Information Sharing Platform Laboratories and Grant-in-Aid for Scientific Research, Ministry of Education, Culture, Sports, Science, and Technology, 16092206.

Diffie–Hellman authenticated key-exchange protocol is not secure, that is, the adversary can compute the session key (See [15]). Recently, Krawczyk [14] proposed a key-exchange protocol called HMQRV which is secure even if the adversary can make session-state reveal queries. Lauter and Mityagin [15] also proposed a key-exchange protocol KEA+ which is secure against the attack with session-state reveal queries.

As we have seen above, the security for key-exchange protocols in the case that randomness is revealed was widely studied. In this paper, we consider a similar situation for public-key encryption, that is, the adversary knows the randomness which was used to compute the ciphertext. In some practical scenarios, there is a possibility that the randomness is revealed. Similar to the case of key-exchange protocols, the randomness used to make a ciphertext may be stored in insecure memory or the pseudorandom generator may be corrupted. Furthermore, the distribution of the randomness may not be uniform, and has some bias by using a practical pseudorandom generator. Then, we may decide the randomness used to compute the ciphertext. As a research concerning the randomness of encryption scheme, Bosley and Dodis [4] considered the question whether privacy for the symmetric-key encryption requires true randomness.

There are many public-key encryption schemes which are secure in the sense of IND-CPA or IND-CCA. Unfortunately, many public-key encryption schemes, which satisfy IND-CPA or IND-CCA, can be broken when the randomness is revealed. For example, consider the encryption function  $E(m; r) := (g^r, m \cdot y^r)$  of the ElGamal encryption scheme with the public-key  $y$ . If the adversary knows the randomness  $r$  of the ciphertext  $(c_1, c_2)$ , the adversary easily gets the plaintext  $m$  by computing  $m = c_2/y^r$ . We can apply a similar discussion to the Cramer-Shoup encryption scheme [6] and its variants generated from the general framework proposed by Cramer and Shoup [7].

We can also see that the Paillier’s encryption scheme [17] is not secure if the randomness is revealed. The encryption function of the Paillier’s encryption scheme is  $E(m; r) := (1 + mN)r^N \bmod N^2$  where  $N$  is the public-key. If the adversary knows the randomness  $r$  of the ciphertext  $c$ , the adversary can get the plaintext  $m$  by computing  $m = (T - 1)/N$  where  $T = c/r^N \bmod N^2$ .

Furthermore, some schemes whose security depend on the random oracles can be also broken in the situation that the randomness is revealed, even if the random oracles are used. For example, consider the Fujisaki-Okamoto conversion. The encryption function of this scheme is defined as  $E_{pk}(m; r) := E_{pk}^{\text{pub}}(r; H(r, m)) || E_{G(r)}^{\text{sym}}(m)$  where  $E_{pk}^{\text{pub}}$  is a public-key encryption scheme with a public-key  $pk$ ,  $E_{G(r)}^{\text{sym}}$  is a symmetric-key encryption scheme with the symmetric-key  $G(r)$ , and  $G, H$  are hash functions (modeled as the random oracles). In this conversion, the randomness  $r$  is encrypted by the public-key encryption scheme, and the plaintext  $m$  is masked by the hash value  $G(r)$  of the randomness  $r$ . We can easily see that the adversary can compute the plaintext if the randomness  $r$  is revealed. Similarly, we can easily see that REACT [16] and the scheme by Pointcheval [19] are also not secure if the randomness is revealed.

We next consider OAEP proposed by Bellare and Rogaway [2]. Let  $\varphi_{pk}$  be a trap-door permutation and  $n$  the length of the plaintext. Then, the encryption function of OAEP is  $E(m; r) := \varphi_{pk}(s || t)$  where  $s = (m || 0^k) \oplus G(r)$ ,  $t = r \oplus H(s)$ . Roughly speaking, Fujisaki, Okamoto, Pointcheval, and Stern [9] showed that OAEP with a trap-door permutation is secure in the sense of IND-CCA2 under the assumption that no poly-time algorithm, given  $c = \varphi_{pk}(s || t)$ , can compute  $s$  with non-negligible probability. However, we can see that the above assumption is not sufficient to prove that OAEP with a trap-door permutation is secure when the randomness is revealed.

In order to see this, we consider a trap-door permutation  $\hat{\varphi}_{pk}(x_1 || x_2 || x_3) := x_1 || \varphi'_{pk}(x_2) || x_3$  where  $|x_1| = n$ ,  $|x_2| = k$ , and  $\varphi'_{pk}$  is one-way. We can see that no poly-time algorithm can compute  $x_1 || x_2$  from  $\hat{\varphi}_{pk}(x_1 || x_2 || x_3)$  with non-negligible probability if  $k$  is sufficiently large and  $\varphi'_{pk}$  is one-way. That is, OAEP with  $\hat{\varphi}_{pk}$  satisfies IND-CCA2. However, in this case, if the randomness is revealed, we can decrypt the plaintext  $m$  without the secret key. We can compute  $G(r)$ , since the randomness  $r$  is revealed, and compute the most  $n$  significant bits of the preimage of  $c = \hat{\varphi}_{pk}(s || t)$  where  $c$  is a ciphertext, that is, the most  $n$  significant bits of  $s$ . Thus, we can compute the plaintext as

$m = [\text{the most } n \text{ significant bits of } G(r)] \oplus [\text{the most } n \text{ significant bits of } s]$ . We can apply a similar argument to OAEP+ [20], which satisfies IND-CCA2 if the underlying trap-door permutation is one-way. That is, OAEP+ can be broken if the randomness is revealed, even if the permutation is one-way.

In this paper, we first formalize the security notion for public-key encryption in the situation that the randomness is revealed.

The most popular formalization for the security on public-key encryption is the indistinguishability (IND). Thus, we consider the formalization of our security notion with the indistinguishability property. However, we can see that the adversary can always win the IND game in both CPA and CCA settings when the randomness is revealed. If the adversary has a pair  $(m_0, m_1)$  of messages, a ciphertext  $c$  of either  $m_0$  or  $m_1$ , and a randomness  $r$  used to compute  $c$ , the adversary can know which message was encrypted by checking whether  $c = E(m_0; r)$  or  $c = E(m_1; r)$ . Thus, the indistinguishability is not fit into the situation that the randomness is revealed.

Therefore, in this paper, we have to consider different formalizations other than the indistinguishability. One alternative would be the semantic security, which was first introduced by Goldwasser and Micali [11]. The semantic security captures the notion of the data-privacy for public-key encryption schemes in a similar way as for secret-key encryption schemes. This formalizes the intuition of the data-privacy that whatever can be efficiently computed about a message from its ciphertext can be also computed without the ciphertext.

Another alternative would be the non-malleability, which was first proposed by Dolev, Dwork, and Naor [8]. The non-malleability captures the notion of the data-privacy that, roughly speaking, given a ciphertext, we cannot obtain a different ciphertext such that the respective plaintexts are related.

In this paper, by naturally modifying the formalizations for the above two candidates, we propose “the semantic security with the randomness revealed” and “the non-malleability with the randomness revealed.” In particular, for the non-malleability with the randomness revealed, we propose two definitions based on the simulation-based non-malleability and the comparison-based non-malleability. Thus, we propose the three security definitions, the semantic security with the randomness revealed (SSR), the simulation-based non-malleability with the randomness revealed (SNMR), and the comparison-based non-malleability with the randomness revealed (CNMR). We also show that CNMR implies SNMR if the message space of the public-key encryption scheme is common to each user.

Roughly speaking, a public-key encryption scheme satisfies security if no polynomial-time adversary, given a ciphertext  $c$ , can gain the information of the plaintext of  $c$  or obtain the ciphertext such that the respective plaintexts are related (without using the secret-key) even if the randomness used to compute  $c$  is given to the algorithm. In addition, there is a major difference between the standard formalizations of the semantic security and the non-malleability and the our formalizations. In our formalizations, in order to give the ciphertexts to the adversary, we uniformly pick a message from the whole message space corresponding to the given public key. If we pick a message from the message space chosen by the adversary, then the adversary can always win the game by choosing the message space containing only two messages.

In addition to the formalization, we consider a scheme which satisfies our security notions. There might be many such existing schemes, however, in this paper, we focus on 3-round OAEP with a trap-door permutation proposed by Phan and Pointcheval [18]. 3-round OAEP satisfies IND-CCA in the random oracle model if the underlying trap-door permutation is partial one-way. Furthermore, the scheme does not have the typical redundancy which OAEP has. We show that this scheme satisfies SSR-CCA, CNMR-CCA, and SNMR-CCA if the trap-door permutation is partial one-way.

The organization of this paper is as follows. In Section 2, we review some definitions. In Section 3, we formalize the semantic security with the randomness revealed (SSR), the simulation-based non-

malleability with the randomness revealed (SNMR), and the comparison-based non-malleability with the randomness revealed (CNMR). In Section 4, we review 3-round OAEP and prove that 3-round OAEP with a partial one-way trap-door permutation provides SSR-CCA, SNMR-CCA, and CNMR-CCA in the random oracle model.

## 2 Preliminaries

First, we briefly review the definition of public-key encryption schemes.

**Definition 1.** A public-key encryption scheme  $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$  consists of three algorithms.

- The key generation algorithm  $\mathcal{K}$  is a randomized algorithm that takes as input a security parameter  $k$  and returns a pair  $(pk, sk)$  of keys, a public key and a matching secret key. For given  $pk$ , a message space  $\text{MSPC}_{\Pi}(pk)$  and a randomness space  $\text{RSPC}_{\Pi}(pk)$  are uniquely determined.
- The encryption algorithm  $\mathcal{E}$  is a randomized algorithm that takes the public key  $pk$  and a plaintext  $m \in \text{MSPC}_{\Pi}(pk)$  and returns a ciphertext  $c$ , using a random coin  $r \xleftarrow{R} \text{RSPC}_{\Pi}(pk)$ .
- The decryption algorithm  $\mathcal{D}$  is a deterministic algorithm that takes the secret key  $sk$  and a ciphertext  $c$  and returns the corresponding plaintext  $m$  or a special symbol  $\perp$  to indicate that the ciphertext is invalid.

Second, we review the definitions of families of trap-door permutations and  $\theta$ -partial one-wayness.

**Definition 2** (Families of Trap-Door Permutations). A family of trap-door permutations  $\mathcal{TP} = (K, \varphi, \psi)$  is described as follows. The key generation algorithm  $K$  takes as input a security parameter  $1^k$  and outputs a public key  $\mathbf{pk}$  and a matching secret key (trap-door)  $\mathbf{sk}$ . The function  $\varphi_{\mathbf{pk}}$  is a permutation over  $\text{Dom}_{\mathcal{TP}}(\mathbf{pk})$  where  $\text{Dom}_{\mathcal{TP}}(\mathbf{pk})$  is uniquely determined by  $\mathbf{pk}$ , while  $\psi_{\mathbf{sk}}$  is the inverse permutation of  $\varphi_{\mathbf{pk}}$ , that is,  $\varphi_{\mathbf{pk}}^{-1} = \psi_{\mathbf{sk}}$ .

**Definition 3** ( $\theta$ -Partial One-Wayness). Let  $k \in \mathbb{N}$  be a security parameter, and  $0 < \theta \leq 1$  a constant. Let  $\mathcal{TP} = (K, \varphi, \psi)$  be a family of trap-door permutations, and  $A$  an adversary. We consider the following experiment:

**Experiment  $\text{Exp}_{\mathcal{TP}, A}^{\theta\text{-pow}}(k)$**   
 $(\mathbf{pk}, \mathbf{sk}) \leftarrow K(1^k); x \xleftarrow{R} \text{Dom}_{\mathcal{TP}}(\mathbf{pk}); y \leftarrow \varphi_{\mathbf{pk}}(x)$   
 $x_1 \leftarrow A(\mathbf{pk}, y)$  where  $|x_1| = \lceil \theta \cdot |x| \rceil$   
**if**  $(\varphi_{\mathbf{pk}}(x_1 || x_2) = y$  for some  $x_2)$  **return** 1 **else return** 0

Here, “ $||$ ” denotes concatenation. We say that  $\mathcal{TP}$  is  $\theta$ -partial one-way if the function  $\text{Adv}_{\mathcal{TP}, A}^{\theta\text{-pow}}(k) = \Pr[\text{Exp}_{\mathcal{TP}, A}^{\theta\text{-pow}}(k) = 1]$  is negligible for any polynomial-time adversary  $A$ .

Note that when  $\theta = 1$  the notion of  $\theta$ -partial one-wayness means the standard notion of one-wayness.

## 3 Security Notions with the Randomness Revealed

In this section, we formalize the semantic security with the randomness revealed (SSR), the simulation-based non-malleability with the randomness revealed (SNMR), and the comparison-based non-malleability with the randomness revealed (CNMR).

There are two major differences between the standard formalizations of the semantic security and the non-malleability and the our formalizations. First, the adversary in our formalizations can obtain the randomness which was used to compute the challenge ciphertext. Second, in our formalizations,

in order to give the ciphertexts to the adversary, we uniformly pick a message from the whole message space corresponding to the given public key.

First, we formalize the semantic security with the randomness revealed (SSR). This formalization can be considered as a natural modification of the semantic security [11, 10].

**Definition 4** (SSR). *Let  $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$  be a public-key encryption scheme. Let  $A$  be an adversary and  $A'$  an algorithm which is called a simulator. Let  $f$  be a polynomial-time computable function (or a polynomial-time algorithm) whose domain and range are  $\text{MSPC}_\Pi(pk)$  and  $\{0, 1\}^*$ , respectively. For  $\text{atk} \in \{\text{cpa}, \text{cca}\}$ , we consider the following experiments:*

**Experiment  $\text{Exp}_{\Pi, A, f}^{\text{ssr-atk-1}}(k)$**   
 $(pk, sk) \leftarrow \mathcal{K}(1^k)$ ;  $m \xleftarrow{R} \text{MSPC}_\Pi(pk)$ ;  $c \leftarrow \mathcal{E}_{pk}(m; r)$ ;  $v \leftarrow A^{\mathcal{O}_{\text{atk}}}(c, r, pk)$   
**if**  $(v = f(m))$  **then return 1 else return 0**

**Experiment  $\text{Exp}_{\Pi, A', f}^{\text{ssr-atk-0}}(k)$**   
 $(pk, sk) \leftarrow \mathcal{K}(1^k)$ ;  $m \xleftarrow{R} \text{MSPC}_\Pi(pk)$ ;  $v \leftarrow A'(pk)$   
**if**  $(v = f(m))$  **then return 1 else return 0**

where  $\mathcal{O}_{\text{cpa}} = \epsilon$  and  $\mathcal{O}_{\text{cca}} = \mathcal{D}_{sk}$ . In the CCA setting, we require that the adversary  $A$  never queries the challenge  $c$  to  $\mathcal{D}_{sk}$ . We define the advantage of  $A$  against  $A'$  via

$$\mathbf{Adv}_{\Pi, A, A', f}^{\text{ssr-atk}}(k) = |\Pr[\mathbf{Exp}_{\Pi, A, f}^{\text{ssr-atk-1}}(k) = 1] - \Pr[\mathbf{Exp}_{\Pi, A', f}^{\text{ssr-atk-0}}(k) = 1]|.$$

We say that  $\Pi$  is secure in the sense of SSR-CPA (resp. SSR-CCA) if for every polynomial-time adversary  $A$  and every polynomial-time computable function  $f$ , there exists a polynomial-time simulator  $A'$  such that  $\mathbf{Adv}_{\Pi, A, A', f}^{\text{ssr-cpa}}(k)$  (resp.  $\mathbf{Adv}_{\Pi, A, A', f}^{\text{ssr-cca}}(k)$ ) is negligible.

Second, we formalize the simulation-based non-malleability with the randomness revealed (SNMR). This formalization can be considered as a natural modification of the simulation-based non-malleability [8, 3]. Note that  $A$  and  $A'$  output a vector  $\mathbf{c} = (c_1, \dots, c_t)$  where  $t = |\mathbf{c}|$ , and  $\mathbf{m} \leftarrow \mathcal{D}_{sk}(\mathbf{c})$  is a operation such that  $m_i \leftarrow \mathcal{D}_{sk}(c_i)$  for each  $i = 1, \dots, t$ , and  $\mathbf{m} \leftarrow (m_1, \dots, m_t)$ .

**Definition 5** (SNMR). *Let  $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$  be a public-key encryption scheme. Let  $A$  be an adversary and  $A'$  an algorithm which is called a simulator. Let  $R$  be a polynomial-time computable relation. For  $\text{atk} \in \{\text{cpa}, \text{cca}\}$ , we consider the following experiments:*

**Experiment  $\text{Exp}_{\Pi, A, R}^{\text{snmr-atk-1}}(k)$**   
 $(pk, sk) \leftarrow \mathcal{K}(1^k)$ ;  $m \xleftarrow{R} \text{MSPC}_\Pi(pk)$ ;  $c \leftarrow \mathcal{E}_{pk}(m; r)$   
 $\mathbf{c} \leftarrow A^{\mathcal{O}_{\text{atk}}}(c, r, pk)$ ;  $\mathbf{m} \leftarrow \mathcal{D}_{sk}(\mathbf{c})$   
**if**  $((c \notin \mathbf{c}) \wedge (\perp \notin \mathbf{m}) \wedge (R(m, \mathbf{m}) = 1))$  **then return 1 else return 0**

**Experiment  $\text{Exp}_{\Pi, A', R}^{\text{snmr-atk-0}}(k)$**   
 $(pk, sk) \leftarrow \mathcal{K}(1^k)$ ;  $m \xleftarrow{R} \text{MSPC}_\Pi(pk)$   
 $\mathbf{c} \leftarrow A'(pk)$ ;  $\mathbf{m} \leftarrow \mathcal{D}_{sk}(\mathbf{c})$   
**if**  $((\perp \notin \mathbf{m}) \wedge (R(m, \mathbf{m}) = 1))$  **then return 1 else return 0**

where  $\mathcal{O}_{\text{cpa}} = \epsilon$  and  $\mathcal{O}_{\text{cca}} = \mathcal{D}_{sk}$ . In the CCA setting, we require that the adversary  $A$  never queries the challenge  $c$  to  $\mathcal{D}_{sk}$ . We define the advantage of  $A$  against  $A'$  via

$$\mathbf{Adv}_{\Pi, A, A', R}^{\text{snmr-atk}}(k) = |\Pr[\mathbf{Exp}_{\Pi, A, R}^{\text{snmr-atk-1}}(k) = 1] - \Pr[\mathbf{Exp}_{\Pi, A', R}^{\text{snmr-atk-0}}(k) = 1]|.$$

We say that  $\Pi$  is secure in the sense of SNMR-CPA (resp. SNMR-CCA) if for every polynomial-time adversary  $A$  and every polynomial-time computable relation  $R$ , there exists a polynomial-time simulator  $A'$  such that  $\mathbf{Adv}_{\Pi, A, A', R}^{\text{snmr-cpa}}(k)$  (resp.  $\mathbf{Adv}_{\Pi, A, A', R}^{\text{snmr-cca}}(k)$ ) is negligible.

Third we formalize the comparison-based non-malleability with the randomness revealed (CNMR). This formalization can be considered as a natural modification of the comparison-based non-malleability [1, 3].

**Definition 6** (CNMR). *Let  $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$  be a public-key encryption scheme. Let  $A$  be the adversary. For  $\text{atk} \in \{\text{cpa}, \text{cca}\}$ , we consider the following experiments:*

**Experiment  $\text{Exp}_{\Pi, A}^{\text{cnmr-atk-1}}(k)$**   
 $(pk, sk) \leftarrow \mathcal{K}(1^k); m \xleftarrow{R} \text{MSPC}_{\Pi}(pk); c \leftarrow \mathcal{E}_{pk}(m; r)$   
 $(R, \mathbf{c}) \leftarrow A^{\mathcal{O}_{\text{atk}}}(c, r, pk); \mathbf{m} \leftarrow \mathcal{D}_{sk}(\mathbf{c})$   
**if  $((c \notin \mathbf{c}) \wedge (\perp \notin \mathbf{m}) \wedge (R(m, \mathbf{m}) = 1))$  then return 1 else return 0**

**Experiment  $\text{Exp}_{\Pi, A}^{\text{cnmr-atk-0}}(k)$**   
 $(pk, sk) \leftarrow \mathcal{K}(1^k); m, m' \xleftarrow{R} \text{MSPC}_{\Pi}(pk); c \leftarrow \mathcal{E}_{pk}(m; r)$   
 $(R, \mathbf{c}') \leftarrow A^{\mathcal{O}_{\text{atk}}}(c, r, pk); \mathbf{m} \leftarrow \mathcal{D}_{sk}(\mathbf{c})$   
**if  $((c \notin \mathbf{c}') \wedge (\perp \notin \mathbf{m}) \wedge (R(m', \mathbf{m}) = 1))$  then return 1 else return 0**

where  $R$  is a polynomial-time computable relation. Above,  $\mathcal{O}_{\text{cpa}} = \epsilon$  and  $\mathcal{O}_{\text{cca}} = \mathcal{D}_{sk}$ . In the CCA setting, we require that the adversary  $A$  never queries the challenge  $c$  (or  $c'$ ) to  $\mathcal{D}_{sk}$ . We define the advantage of  $A$  via

$$\mathbf{Adv}_{\Pi, A}^{\text{cnmr-atk}}(k) = |\Pr[\mathbf{Exp}_{\Pi, A}^{\text{cnmr-atk-1}}(k) = 1] - \Pr[\mathbf{Exp}_{\Pi, A}^{\text{cnmr-atk-0}}(k) = 1]|.$$

We say that  $\Pi$  is secure in the sense of CNMR-CPA (resp. CNMR-CCA) if  $\mathbf{Adv}_{\Pi, A}^{\text{cnmr-cpa}}(k)$  (resp.  $\mathbf{Adv}_{\Pi, A}^{\text{cnmr-cca}}(k)$ ) is negligible for any polynomial-time adversary  $A$ .

**Remark 1.** Bellare and Sahai [3] proposed the security notion for public-key encryption called the indistinguishability under parallel attack. They also showed that the simulation-based non-malleability, the comparison-based non-malleability, and their notion are equivalent. However, as mentioned in Section 1, the indistinguishability is not fit into the situation that the randomness is revealed. Therefore, in this paper, we propose two security notions based on the simulation-based non-malleability and the comparison-based non-malleability.

In [3], Bellare and Sahai showed that the comparison-based non-malleability implies the simulation-based non-malleability. In the following, we show that CNMR implies SNMR under the assumption that the message space is common to each user.

**Theorem 1.** *Let  $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$  be a public-key encryption scheme such that the message space is common to each public key. For any  $\text{ATK} \in \{\text{CPA}, \text{CCA}\}$ , if a public-key encryption scheme  $\Pi$  is secure in the sense of CNMR-ATK, then  $\Pi$  is also secure in the sense of SNMR-ATK.*

*Proof.* The proof is similar to the proof that CNM-ATK implies SNM-ATK by Bellare and Sahai [3].

Let  $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$  be a public-key encryption scheme. Assume that  $\Pi$  is secure in the sense of CNMR-ATK. Let  $A$  be a polynomial-time adversary  $A$  and  $R$  a polynomial-time computable relation. In order to show that the scheme is secure in the sense of SNMR-ATK, we have to construct the simulator  $A'$ . The idea is that  $A'$  will run  $A$  on a newly chosen public key of which  $A'$  knows the corresponding decryption key.

**Algorithm  $A'(pk)$**   
 $(pk', sk') \leftarrow \mathcal{K}(1^k); m' \xleftarrow{R} \text{MSPC}_{\Pi}(pk'); c' \leftarrow \mathcal{E}_{pk'}(m'; r')$   
 $\mathbf{c}' \leftarrow A^{\mathcal{O}'_{\text{atk}}}(c', r', pk')$   
**if  $(c \in \mathbf{c}')$  then return 0**  
 $\mathbf{m}' \leftarrow \mathcal{D}_{sk'}(\mathbf{c}'); \mathbf{c} \leftarrow \mathcal{E}_{pk}(\mathbf{m}')$   
**return  $\mathbf{c}$**

where  $\mathcal{O}'_{\text{cpa}} = \epsilon$  and  $\mathcal{O}'_{\text{cca}} = \mathcal{D}_{sk'}$ . Note that  $A'$  can run the decryption algorithm with  $sk'$  since  $A'$  has the secret key  $sk'$ .

Now, we want to show that  $\mathbf{Adv}_{\Pi, A, A', R}^{\text{snmr-atk}}(k)$  is negligible. We prove this by using the assumption that  $\Pi$  is secure in the sense of CNMR-ATK. We consider the following adversary  $B$  attacking  $\Pi$  in the sense of CNMR-ATK.

**Algorithm**  $B^{\mathcal{O}_{\text{atk}}}(c, r, pk)$   
 $\mathbf{c} \leftarrow A^{\mathcal{O}_{\text{atk}}}(c, r, pk)$   
**return**  $(R, \mathbf{c})$

By the definition of  $B$ , it is easy to see that  $\Pr[\mathbf{Exp}_{\Pi, A, R}^{\text{snmr-atk-1}}(k) = 1] = \Pr[\mathbf{Exp}_{\Pi, B}^{\text{cnmr-atk-1}}(k) = 1]$ . We describe the experiment  $\mathbf{Exp}_{\Pi, A', R}^{\text{snmr-atk-0}}(k)$ , replacing the simulator  $A'$  in the experiment with that described above.

**Experiment**  $\mathbf{Exp}_{\Pi, A', R}^{\text{snmr-atk-0}}(k)$   
 $(pk, sk) \leftarrow \mathcal{K}(1^k); m \xleftarrow{R} \text{MSPC}_{\Pi}(pk)$   
 $(pk', sk') \leftarrow \mathcal{K}(1^k); m' \xleftarrow{R} \text{MSPC}_{\Pi}(pk'); c' \leftarrow \mathcal{E}_{pk'}(m'; r')$   
 $\mathbf{c}' \leftarrow A^{\mathcal{O}_{\text{atk}}}(c', r', pk')$   
**if**  $(c \in \mathbf{c}')$  **then return** 0  
 $\mathbf{m}' \leftarrow \mathcal{D}_{sk'}(\mathbf{c}'); \mathbf{c} \leftarrow \mathcal{E}_{pk}(\mathbf{m}')$   
 $\mathbf{m} \leftarrow \mathcal{D}_{sk}(\mathbf{c})$   
**if**  $((\perp \notin \mathbf{m}) \wedge (R(m, \mathbf{m}) = 1))$  **then return** 1 **else return** 0

Since  $\text{MSPC}_{\Pi}(pk) = \text{MSPC}_{\Pi}(pk')$  and  $\mathbf{m} = \mathbf{m}'$ , we can rewrite the above experiment as follows.

**Experiment**  $\mathbf{Exp}_{\Pi, A', R}^{\text{snmr-atk-0}}(k)$   
 $(pk', sk') \leftarrow \mathcal{K}(1^k); m, m' \xleftarrow{R} \text{MSPC}_{\Pi}(pk'); c' \leftarrow \mathcal{E}_{pk'}(m'; r')$   
 $\mathbf{c}' \leftarrow A^{\mathcal{O}_{\text{atk}}}(c', r', pk'); \mathbf{m}' \leftarrow \mathcal{D}_{sk'}(\mathbf{c}')$   
**if**  $((c \notin \mathbf{c}') \wedge (\perp \notin \mathbf{m}') \wedge (R(m, \mathbf{m}') = 1))$  **then return** 1 **else return** 0

Therefore, we can see that  $\Pr[\mathbf{Exp}_{\Pi, A', R}^{\text{snmr-atk-0}}(k) = 1] = \Pr[\mathbf{Exp}_{\Pi, B}^{\text{cnmr-atk-0}}(k) = 1]$ . Hence,  $\mathbf{Adv}_{\Pi, A, A', R}^{\text{snmr-atk}}(k) = \mathbf{Adv}_{\Pi, B}^{\text{cnmr-atk}}(k)$ . Since the advantage of  $B$  is negligible, that of  $A$  is also negligible.  $\square$

## 4 3-Round OAEP and its Security

In this section, we review 3-round OAEP proposed by Phan and Pointcheval [18], and prove that 3-round OAEP with a partial one-way trap-door permutation provides SSR-CCA, SNMR-CCA, and CNMR-CCA in the random oracle model.

### 4.1 3-Round OAEP

In this section, we review 3-round OAEP by Phan and Pointcheval [18]. 3-round OAEP is a novel construction in order to remove redundancy of (2-round) OAEP. Unlike OAEP, all of the ciphertexts of 3-round OAEP are valid (which means the encryption function is not only a probabilistic injection, but also a surjection), and 3-round OAEP does not have the usual redundancy  $m||0^k$  which OAEP has. In [18], they proposed 3-round OAEP with *any* trap-door *bijection*. Thus, the domain and the range of the function may be different. We modify their definition to define 3-round OAEP with a trap-door *permutation*.

**Definition 7** (3-round OAEP with a trap-door permutation). *3-round OAEP*  $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$  with a family of trap-door permutations  $\mathcal{TP} = (K, \varphi, \psi)$  is as follows. The key generation algorithm  $\mathcal{K}$  takes a security parameters  $1^k$  and  $1^\ell$ , runs the key generation algorithm  $K(1^k, 1^\ell)$  of  $\mathcal{TP}$ , and

gets a pair of public and secret keys  $(\mathbf{pk}, \mathbf{sk})$  for  $\mathcal{TP}$ , where  $\text{Dom}_{\mathcal{TP}}(\mathbf{pk}) = \{0, 1\}^{k+\ell}$ . Then, the key generation algorithm  $\mathcal{K}$  returns a public key  $pk = (\mathbf{pk}, 1^k, 1^\ell)$  and a corresponding secret key  $sk = \mathbf{sk}$ . For any  $pk$ , the plaintext space  $\text{MSPC}_{\Pi}(pk)$  and the randomness space  $\text{RSPC}_{\Pi}(pk)$  are  $\{0, 1\}^\ell$  and  $\{0, 1\}^k$ , respectively. The encryption and decryption algorithms are as follows. Note that  $F : \{0, 1\}^k \rightarrow \{0, 1\}^\ell$ ,  $G : \{0, 1\}^\ell \rightarrow \{0, 1\}^k$ , and  $H : \{0, 1\}^k \rightarrow \{0, 1\}^\ell$  are hash functions.

<b>Algorithm <math>\mathcal{E}_{pk}(m; r)</math></b> $r \xleftarrow{R} \{0, 1\}^k$ $s \leftarrow m \oplus F(r)$ $t \leftarrow r \oplus G(s)$ $u \leftarrow s \oplus H(t)$ $c \leftarrow \varphi_{\mathbf{pk}}(t  u)$ <b>return <math>c</math></b>	<b>Algorithm <math>\mathcal{D}_{sk}(c)</math></b> $t  u \leftarrow \psi_{\mathbf{sk}}(c)$ where $ t  = k$ and $ u  = \ell$ $s \leftarrow u \oplus H(t)$ $r \leftarrow t \oplus G(s)$ $m \leftarrow s \oplus F(r)$ <b>return <math>m</math></b>
---	---

Phan and Pointcheval proved that 3-round OAEP with a family of trap-door permutations  $\mathcal{TP}$  is secure in the sense of IND-CCA2 if  $\mathcal{TP}$  is  $\theta$ -partial one-way where  $\theta = k/(k + \ell)$ .

## 4.2 SSR-CCA Security of 3-Round OAEP with a Trap-Door Permutation

In this section, we prove that 3-round OAEP with a partial one-way trap-door permutation provides SSR-CCA in the random oracle model.

In the proof of IND-CCA2 by Phan and Pointcheval, they showed that the probability that the adversary asks  $r^*$  is negligible. Then, the adversary cannot get any information about the message  $m$ , since the value  $F(r^*)$  cannot be guessed by the adversary. However, in our proof, the randomness  $r^*$  is revealed to the adversary. Thus, we take a different strategy in the proof of the following theorem.

**Theorem 2.** *Assume that there exists a polynomial-time computable function  $\tilde{f} : \{0, 1\}^\ell \rightarrow \{0, 1\}^*$ , and an adversary  $A$  attacking the SSR-CCA security of 3-round OAEP  $\Pi$  with  $\mathcal{TP}$ , and making at most  $q_d$  queries to the decryption oracle,  $q_f$   $F$ -oracle queries,  $q_g$   $G$ -oracle queries, and  $q_h$   $H$ -oracle queries. Then, there exist a simulator  $A'$  of  $A$  and a  $\theta$ -partial inverting adversary  $B$  for  $\mathcal{TP}$ , such that for any  $k, \ell$ , and  $\theta = \frac{k}{k+\ell}$ ,*

$$\mathbf{Adv}_{\Pi, A, A', \tilde{f}}^{\text{ssr-cca}}(k) \leq \frac{q_d(6q_g + 3q_d + 1) + q_g}{2^\ell} + \frac{3q_d(3q_f + 2q_d)}{2^k} + 2q_h \cdot \mathbf{Adv}_{\mathcal{TP}, B}^{\theta\text{-pow}}(k),$$

and the running time of  $A$  and that of  $B$  are bounded by that of  $A$  plus  $q_g \cdot q_h \cdot T_\varphi + q_d \cdot T_{lu}$  where  $T_\varphi$  denotes the time for evaluating the permutation  $\varphi_{\mathbf{pk}}$  and  $T_{lu}$  denotes the time for looking up in a list.

*Proof.* Assume that there exists a polynomial-time computable function  $\tilde{f} : \{0, 1\}^\ell \rightarrow \{0, 1\}^*$ , and an adversary  $A$  attacking the SSR-CCA security of 3-round OAEP  $\Pi$  with  $\mathcal{TP}$ .

In the following, we construct the simulator  $A'$  of  $A$  and the  $\theta$ -partial inverting adversary  $B$  for  $\mathcal{TP}$ . In order to evaluate the advantage of  $A$  and that of  $B$ , we define a sequence  $\text{Exp}_1, \text{Exp}_2$ , etc., of modified experiments starting from the actual experiment  $\text{Exp}_0$  of  $A$  in the SSR-CCA game, that is,  $\text{Exp}_0$  is the same as  $\mathbf{Exp}_{\Pi, A, \tilde{f}}^{\text{ssr-cca-1}}$ . Each of the experiments operates on the same underlying probability space: the public and secret keys of the cryptosystems, the coin tosses of the adversary  $A$ , the random oracles  $F, G$ , and  $H$ , and the message and the randomness for the challenge ciphertext.

In the following, all variables with asterisk refer to the challenge ciphertext, and all variables with no asterisk refer to the decryption queries.

$\text{Exp}_0$ .  $\text{Exp}_0$  is the same as  $\mathbf{Exp}_{\Pi, A, \tilde{f}}^{\text{ssr-cca-1}}$  in the actual SSR-CCA game. A pair of keys  $(pk, sk)$  is generated by  $\mathcal{K}(1^k, 1^\ell)$ , and the CCA-adversary  $A$  takes  $pk = (\mathbf{pk}, 1^k, 1^\ell)$ , the challenge ciphertext  $c^*$ , and the randomness  $r^*$  where

$$m \xleftarrow{R} \{0, 1\}^\ell, \quad r^* \xleftarrow{R} \{0, 1\}^k, \quad s^* = m \oplus F(r^*), \quad t^* = r^* \oplus G(s^*), \quad u^* = s^* \oplus H(t^*), \quad c^* = \varphi_{\mathbf{pk}}(t^*||u^*),$$

and  $A$  outputs  $v$ . In the above experiment, the adversary  $A$  can make access to the random oracles  $F, G, H$ , and the decryption oracle  $\mathcal{D}_{sk}$ . However,  $A$  cannot ask the challenge ciphertext  $c^*$  to the decryption oracle.

We denote by  $S_0$  the event “ $v = \tilde{f}(m)$ ” and use a similar notation  $S_i$  in each  $\text{Exp}_i$  below. By the definition, we have  $\Pr[S_0] = \Pr[\mathbf{Exp}_{\Pi, A, \tilde{f}}^{\text{ssr-cca-1}}(k) = 1]$ .

**Remark 2.** *In the proof of IND-CCA2 by Phan and Pointcheval, they modified the experiment by setting  $r^*$  independently of anything else, as well as  $F(r^*)$ . Furthermore, they bounded the probability that the adversary asks  $r^*$  by negligible probability in the following experiments.*

*However, we cannot use this strategy, since the adversary knows  $r^*$  and we cannot bound the probability that the adversary asks  $r^*$ . Thus, we take a different strategy in the following experiments.*

$\text{Exp}_1$ . We choose two random values  $s^+ \xleftarrow{R} \{0, 1\}^\ell$  and  $g^+ \xleftarrow{R} \{0, 1\}^k$  in advance (i.e. before the adversary  $A$  runs), and use  $s^+$  and  $g^+$  instead of  $s^*$  and  $G(s^*)$ , respectively. In  $\text{Exp}_1$ , we apply the following special rules.

Note that we set  $s^*$  before choosing  $m$  but  $s^*$  and  $m$  are still identically distributed as those in  $\text{Exp}_0$ .

**R1:** We compute the challenge ciphertext  $c^*$  by setting

$$s^* \leftarrow s^+, \quad m \leftarrow s^* \oplus F(r^*), \quad t^* \leftarrow r^* \oplus g^+.$$

**R2:** Whenever the random oracle  $G$  is queried at  $s^*$ , the answer is  $g^+$ .

Since we replace  $(s^*, G(s^*), m)$  by a different, but identically distributed (by the definition of the random oracle  $G$ ) set of random variables, we have  $\Pr[S_1] = \Pr[S_0]$ .

$\text{Exp}_2$ . In this experiment, we drop the rule **R2** from  $\text{Exp}_1$ . Therefore,  $g^+$  is just used for computing the challenge ciphertext, and if  $s^*$  is queried to  $G$  then we respond not  $g^+$  but  $G(s^*)$  by using the random oracle  $G$ .

$\text{Exp}_1$  and  $\text{Exp}_2$  may differ if  $s^*$  is queried to  $G$ . Let  $\text{AskG}_2$  denotes the event that, in  $\text{Exp}_2$ ,  $s^*$  is queried to  $G$  (except by the encryption oracle, for producing the challenge). We use an identical notation  $\text{AskG}_i$  for each  $\text{Exp}_i$  below. Then,  $|\Pr[S_2] - \Pr[S_1]| \leq \Pr[\text{AskG}_2]$ .

$\text{Exp}_3$ . We now set  $t^*$  independently of anything else, as well as  $H(t^*)$ . We choose two random values  $t^+ \xleftarrow{R} \{0, 1\}^k$  and  $h^+ \xleftarrow{R} \{0, 1\}^\ell$  in advance (i.e. before the adversary  $A$  runs), and use  $t^+$  instead of  $t^*$ , as well as  $h^+$  instead of  $H(t^*)$ . In  $\text{Exp}_3$ , we apply the following special rules. Note that we change the way to compute  $g^+$  (but still identically distributed as that in  $\text{Exp}_2$ ).

**R1':** We compute the challenge ciphertext  $c^*$  by setting

$$t^* \leftarrow t^+, \quad g^+ \leftarrow t^+ \oplus r^*, \quad u^* \leftarrow s^* \oplus h^+.$$

**R2':** Whenever the random oracle  $H$  is queried at  $t^+$ , the answer is  $h^+$ .

Since we replace the set of elements  $(t^*, H(t^*), g^+)$  by a different, but identically distributed (by the definition of the random oracle  $H$ ) set of random variables, we have  $\Pr[S_3] = \Pr[S_2]$  and  $\Pr[\text{AskG}_3] = \Pr[\text{AskG}_2]$ .

$\text{Exp}_4$ . In this experiment, we drop the rule **R2'** from  $\text{Exp}_3$ . Therefore,  $h^+$  is just used for computing the challenge ciphertext, and if  $t^*$  is queried to  $H$  then we respond not  $h^+$  but  $H(t^*)$  by using the random oracle  $H$ .

$\text{Exp}_3$  and  $\text{Exp}_4$  may differ if  $t^*$  is queried to  $H$ . Let  $\text{AskH}_4$  denotes the event that, in  $\text{Exp}_4$ ,  $t^*$  is queried to  $H$  (except by the encryption oracle, for producing the challenge). We use

an identical notation  $\text{AskH}_i$  for each  $\text{Exp}_i$  below. Then,  $|\Pr[S_4] - \Pr[S_3]| \leq \Pr[\text{AskH}_4]$  and  $|\Pr[\text{AskG}_4] - \Pr[\text{AskG}_3]| \leq \Pr[\text{AskH}_4]$ .

$\text{Exp}_5$ . In this experiment, we randomly choose  $u^+ \xleftarrow{R} \{0, 1\}^\ell$  and simply set  $u^* \leftarrow u^+$ , instead of  $s^* \oplus h^+$ . Since  $h^+$  is uniformly distributed over  $\{0, 1\}^\ell$  and not revealed to the adversary, the distribution of  $u^+$  and that of  $u^* = s^* \oplus h^+$  in  $\text{Exp}_4$  are identical. Therefore,  $\Pr[S_5] = \Pr[S_4]$ ,  $\Pr[\text{AskG}_5] = \Pr[\text{AskG}_4]$ , and  $\Pr[\text{AskH}_5] = \Pr[\text{AskH}_4]$ .

The challenge ciphertext  $c^*$  in this experiment is computed by

$$r^* \xleftarrow{R} \{0, 1\}^k, \quad s^+ \xleftarrow{R} \{0, 1\}^\ell, \quad s^* \leftarrow s^+, \quad m \leftarrow s^* \oplus F(r^*),$$

and

$$t^+ \xleftarrow{R} \{0, 1\}^k, \quad u^+ \xleftarrow{R} \{0, 1\}^\ell, \quad t^* \leftarrow t^+, \quad u^* \leftarrow u^+, \quad c^* \leftarrow \varphi_{\text{pk}}(t^* || u^*).$$

Here,  $s^+$  is never revealed to the adversary and the distribution of  $s^* = s^+$  is independent of the adversary's view. Therefore,  $\Pr[\text{AskG}_5] \leq (q_g + q_d)/2^\ell$ .

$\text{Exp}_6$ . In  $\text{Exp}_5$ , when manufacturing the challenge ciphertext, we randomly choose  $c^+ \xleftarrow{R} \{0, 1\}^{k+\ell}$ , and simply set  $c^* \leftarrow c^+$ , ignoring the encryption oracle altogether.

Since  $\varphi_{\text{pk}}$  is a permutation over  $\{0, 1\}^{k+\ell}$ , and  $t^* = t^+$  and  $u^* = u^+$  are uniformly distributed over  $\{0, 1\}^k$  and  $\{0, 1\}^\ell$ , respectively, the distribution of  $c^* = \varphi_{\text{pk}}(t^* || u^*)$  is the same as that of  $c^+$ . Thus, we have  $\Pr[S_6] = \Pr[S_5]$  and  $\Pr[\text{AskH}_6] = \Pr[\text{AskH}_5]$ .

$\text{Exp}_7$ . In this experiment, we simulate the random oracles  $F, G, H$ . We use three lists,  $F$ -List,  $G$ -List,  $H$ -List for simulating the random oracles  $F, G$ , and  $H$ , respectively. They are initially set to empty lists.

- When the adversary, the decryption oracle, or the encryption oracle (which makes a challenge ciphertext) makes a query  $r \in \{0, 1\}^k$  to  $F$ , if there exists a pair  $(r, f) \in F$ -List then we respond  $f$ . Otherwise, we respond a random string  $f \xleftarrow{R} \{0, 1\}^\ell$  and put  $(r, f)$  into  $F$ -List.
- When the adversary or the decryption oracle makes a query  $s \in \{0, 1\}^\ell$  to  $G$ , if there exists a pair  $(s, g) \in G$ -List then we respond  $g$ . Otherwise, we respond a random string  $g \xleftarrow{R} \{0, 1\}^k$  and put  $(s, g)$  into  $G$ -List.
- When the adversary or the decryption oracle makes a query  $t \in \{0, 1\}^k$  to  $H$ , if there exists a pair  $(t, h) \in H$ -List then we respond  $h$ . Otherwise, we respond a random string  $h \xleftarrow{R} \{0, 1\}^\ell$  and put  $(t, h)$  into  $H$ -List.

Since we can simulate these oracles perfectly, we have  $\Pr[S_7] = \Pr[S_6]$  and  $\Pr[\text{AskH}_7] = \Pr[\text{AskH}_6]$ .

$\text{Exp}_8$ . We now simulate the decryption oracle by using  $D$ -List, which is initially set to empty. For a decryption query  $c$ , if there exists a pair  $(m, c) \in D$ -List then we respond  $m$  as the plaintext. Otherwise, we simulate the decryption oracle as follows and put  $(m, c)$  into  $D$ -List:

**D-1.** We look up for  $(t, h) \in H$ -List and  $(s, g) \in G$ -List such that  $\varphi_{\text{pk}}(t, s \oplus h) = c$ . If the record is found, we define  $u \leftarrow s \oplus h$ . Otherwise, we set  $t \leftarrow \perp$  and  $u \leftarrow \perp$ .

**D-2.** If  $(t, h) \in H$ -List and  $(s, g) \in G$ -List (i.e. the first case in **D-1**), then we compute

$$r \leftarrow t \oplus g, \quad f \leftarrow F(r), \quad \text{and} \quad m \leftarrow s \oplus f,$$

and return  $m$ . Otherwise (i.e. the second case in **D-1**), we choose  $m \xleftarrow{R} \{0, 1\}^\ell$  and return  $m$ .

In addition, we modify the simulation of the random oracle  $G$  as follows:

Whenever a pair  $(s, g)$  is added to  $G$ -List, the following process is executed: for any  $(t, h) \in H$ -List and any  $(m, c) \in D$ -List such that  $c = \varphi_{\text{pk}}(t || (h \oplus s))$ , we compute  $r \leftarrow t \oplus g$ ,  $f \leftarrow m \oplus s$ , and add  $(r, f)$  to  $F$ -List.

The simulations of the decryption oracle and the random oracle  $G$  described above are the same as those in the proof of IND-CCA2 by Phan and Pointcheval. We prove the following lemma in Appendix A by a similar way as that in the proof of IND-CCA2 by Phan and Pointcheval.

**Lemma 1.**

$$\Pr[\text{AskH}_7] \leq q_d(2q_g + q_d)/2^\ell + q_d(3q_f + 2q_d)/2^k + \Pr[\text{AskH}_8],$$

$$\Pr[S_7] \leq q_d(2q_g + q_d)/2^\ell + q_d(3q_f + 2q_d)/2^k + \Pr[S_8],$$

and the running time for simulating the decryption oracle, as well as the random oracles is bounded by  $q_g \cdot q_h \cdot T_\varphi + q_d \cdot T_{lu}$  where  $T_\varphi$  denotes the time for evaluating the permutation  $\varphi_{\text{pk}}$  and  $T_{lu}$  denotes the time for looking up in a list.

We now evaluate  $\Pr[S_8]$  and  $\Pr[\text{AskH}_8]$ . We first construct the simulator  $A'$  of  $A$  as follows.

1.  $A'$  takes  $\text{pk}$  and sets  $r^* \xleftarrow{R} \{0, 1\}^k$ ,  $c^+ \xleftarrow{R} \{0, 1\}^{k+\ell}$ , and  $c^* \leftarrow c^+$ .
2.  $A'$  runs  $A$  against  $\text{Exp}_8$  with the input  $(c^*, r^*)$ .
3.  $A'$  outputs  $v$  which is the output of  $A$ .

Above,  $A'$  does not use the random oracles  $F, G, H$  which the simulator has, but simulates these oracles as in  $\text{Exp}_8$ . The simulator  $A'$  also simulates the decryption oracle  $\mathcal{D}_{sk}$  for  $A$  as in  $\text{Exp}_8$ . It is easy to see that  $\Pr[S_8] = \Pr[\mathbf{Exp}_{\Pi, A', f}^{\text{ssr-cca-0}}(k) = 1]$ .

We next construct an algorithm  $B$  attacking  $\theta$ -partial one-wayness of  $\mathcal{TP}$ , where  $\theta = k/(k + \ell)$ , by using  $A$  against  $\text{Exp}_8$ .

1.  $B$  takes  $\text{pk}$  and  $y^*$  where  $x^* \xleftarrow{R} \{0, 1\}^{k+\ell}$  and  $y^* = \varphi_{\text{pk}}(x^*)$ . Then,  $B$  sets  $c^* \leftarrow y^*$  and  $r^* \leftarrow \{0, 1\}^k$ .
2.  $B$  runs  $A$  against  $\text{Exp}_8$  with the input  $(c^*, r^*)$ .
3. When  $A$  terminates,  $B$  outputs  $t \xleftarrow{R} \{t' | (t', h') \in H\text{-List}\}$ .

Note that  $B$  simulates the random oracles and the decryption oracle for  $A$  as in  $\text{Exp}_8$  (by using  $F$ -List,  $G$ -List,  $H$ -List, and  $D$ -List). If  $\text{AskH}_8$  occurs then there exists an element  $t^* \in \{0, 1\}^k$  in  $H$ -List such that  $\varphi_{\text{pk}}(t^* || u') = y^*$  for some  $u' \in \{0, 1\}^\ell$ . Therefore, we have  $\Pr[\text{AskH}_8] \leq q_h \cdot \mathbf{Adv}_{\mathcal{TP}, B}^{\theta\text{-pow}}(k)$ .

In conclusion, we have

$$\begin{aligned} & \Pr[\mathbf{Exp}_{\Pi, A', f}^{\text{ssr-cca-1}}(k) = 1] \\ &= \Pr[S_0] = \Pr[S_1] \leq \Pr[S_2] + \Pr[\text{AskG}_2] = \Pr[S_3] + \Pr[\text{AskG}_3] \\ &\leq \Pr[S_4] + \Pr[\text{AskH}_4] + \Pr[\text{AskG}_4] + \Pr[\text{AskH}_4] \\ &= \Pr[S_5] + \Pr[\text{AskG}_5] + 2 \cdot \Pr[\text{AskH}_5] \\ &\leq \Pr[S_5] + (q_g + q_d)/2^\ell + 2 \cdot \Pr[\text{AskH}_5] \\ &= \Pr[S_6] + (q_g + q_d)/2^\ell + 2 \cdot \Pr[\text{AskH}_6] \\ &= \Pr[S_7] + (q_g + q_d)/2^\ell + 2 \cdot \Pr[\text{AskH}_7] \\ &\leq \Pr[S_8] + (q_g + q_d)/2^\ell + 2 \cdot \Pr[\text{AskH}_8] + 3 \cdot (q_d(2q_g + q_d)/2^\ell + q_d(3q_f + 2q_d)/2^k) \\ &\leq \Pr[\mathbf{Exp}_{\Pi, A', f}^{\text{ssr-cca-0}}(k) = 1] + \frac{q_d(6q_g + 3q_d + 1) + q_g}{2^\ell} + \frac{3q_d(3q_f + 2q_d)}{2^k} + 2q_h \cdot \mathbf{Adv}_{\mathcal{TP}, B}^{\theta\text{-pow}}(k), \end{aligned}$$

and

$$\mathbf{Adv}_{\Pi, A, A', \tilde{f}}^{\text{ssr-cca}}(k) \leq \frac{q_d(6q_g + 3q_d + 1) + q_g}{2^\ell} + \frac{3q_d(3q_f + 2q_d)}{2^k} + 2q_h \cdot \mathbf{Adv}_{\mathcal{TP}, B}^{\theta\text{-pow}}(k).$$

We now estimate the running time of  $A$  and that of  $B$ . From Lemma 1, the running time for simulating the decryption oracle, as well as the random oracles is bounded by  $q_g \cdot q_h \cdot T_\varphi + q_d \cdot T_{lu}$  where  $T_\varphi$  denotes the time for evaluating the permutation  $\varphi_{pk}$  and  $T_{lu}$  denotes the time for looking up in a list. Therefore, the running time of  $A$  and that of  $B$  are bounded by that of  $A$  plus  $q_g \cdot q_h \cdot T_\varphi + q_d \cdot T_{lu}$ .  $\square$

### 4.3 CNMR-CCA Security of 3-Round OAEP with a Trap-Door Permutation

In this section, we prove that 3-round OAEP with a partial one-way trap-door permutation provides CNMR-CCA in the random oracle model.

**Theorem 3.** *Assume that there exists an adversary  $A$  attacking the CNMR-CCA security of 3-round OAEP  $\Pi$  with  $\mathcal{TP}$ , and making at most  $q_d$  queries to the decryption oracle,  $q_f$   $F$ -oracle queries,  $q_g$   $G$ -oracle queries, and  $q_h$   $H$ -oracle queries. Then, there exists a  $\theta$ -partial inverting adversary  $B$  for  $\mathcal{TP}$ , such that for any  $k, \ell$ , and  $\theta = \frac{k}{k+\ell}$ ,*

$$\mathbf{Adv}_{\Pi, A}^{\text{cnmr-cca}}(k) \leq \frac{3q_d(2q_g + q_d)}{2^\ell} + \frac{2q_d(3q_f + 2q_d)}{2^k} + 2q_h \cdot \mathbf{Adv}_{\mathcal{TP}, B}^{\theta\text{-pow}}(k),$$

and the running time of  $B$  is that of  $A$  plus  $q_g \cdot q_h \cdot T_\varphi + q_d \cdot T_{lu}$  where  $T_\varphi$  denotes the time for evaluating the permutation  $\varphi_{pk}$  and  $T_{lu}$  denotes the time for looking up in a list.

*Proof (sketch).* Assume that there exists an adversary  $A$  attacking the CNMR-CCA security of 3-round OAEP  $\Pi$  with  $\mathcal{TP}$ . We modify the experiment  $\mathbf{Exp}_{\Pi, A}^{\text{cnmr-cca-1}}(k)$  in a similar way as that in the proof of Theorem 2.

We first review the experiment  $\mathbf{Exp}_{\Pi, A}^{\text{cnmr-cca-1}}(k)$ . In this experiment, a pair of keys  $(pk, sk)$  is generated by  $\mathcal{K}(1^k, 1^\ell)$ . Then, the CCA-adversary  $A$  takes  $pk = (pk, 1^k, 1^\ell)$ , the challenge ciphertext  $c^*$ , and the randomness  $r^*$  where

$$m \stackrel{R}{\leftarrow} \{0, 1\}^\ell, \quad r^* \stackrel{R}{\leftarrow} \{0, 1\}^k, \quad s^* = m \oplus F(r^*), \quad t^* = r^* \oplus G(s^*), \quad u^* = s^* \oplus H(t^*), \quad c^* = \varphi_{pk}(t^* || u^*),$$

and  $A$  outputs  $(R, \mathbf{c})$ . In the above experiment, the adversary  $A$  can make access to the random oracles  $F, G, H$ , and the decryption oracle  $\mathcal{D}_{sk}$ . However,  $A$  cannot ask the challenge ciphertext  $c^*$  to the decryption oracle.

We now construct the modified experiment  $\widetilde{\mathbf{Exp}}_{\Pi, A}(k)$  by applying the same modification from  $\mathbf{Exp}_0$  to  $\mathbf{Exp}_6$  in the proof of Theorem 2. That is, we choose  $c^+ \stackrel{R}{\leftarrow} \{0, 1\}^{k+\ell}$  and simply set  $c^* \leftarrow c^+$ . From a similar argument in the proof of Theorem 2, we have  $|\Pr[\mathbf{Exp}_{\Pi, A}^{\text{cnmr-cca-1}}(k) = 1] - \Pr[\widetilde{\mathbf{Exp}}_{\Pi, A}(k) = 1]| \leq (q_g + q_d)/2^\ell + 2 \cdot \Pr[\text{AskH}]$ , where  $\text{AskH}$  denotes the event that  $t^*$  (the most  $\theta$  significant bits of  $\psi_{sk}(c^*)$ ) is queried to  $H$  by the adversary  $A$  in  $\mathbf{Exp}_{\Pi, A}^{\text{cnmr-cca-1}}(k)$  (or  $\widetilde{\mathbf{Exp}}_{\Pi, A}(k)$ ).

Here, we can see that the experiment  $\widetilde{\mathbf{Exp}}_{\Pi, A}(k)$  is essentially the same as the experiment  $\mathbf{Exp}_{\Pi, A}^{\text{cnmr-cca-0}}(k)$ . In fact, the distribution of  $(c^*, r^*)$  is independent of  $m$  in  $\widetilde{\mathbf{Exp}}_{\Pi, A}(k)$ , and it is also independent of  $m'$  in  $\mathbf{Exp}_{\Pi, A}^{\text{cnmr-cca-0}}(k)$ . Furthermore,  $m$  in  $\widetilde{\mathbf{Exp}}_{\Pi, A}(k)$  and  $m'$  in  $\mathbf{Exp}_{\Pi, A}^{\text{cnmr-cca-0}}(k)$  have the same distribution. Therefore, we have  $\Pr[\widetilde{\mathbf{Exp}}_{\Pi, A}(k) = 1] = \Pr[\mathbf{Exp}_{\Pi, A}^{\text{cnmr-cca-0}}(k) = 1]$ .

Hence, we have

$$\begin{aligned} \Pr[\mathbf{Exp}_{\Pi, A}^{\text{cnmr-cca-1}}(k) = 1] &\leq \Pr[\widetilde{\mathbf{Exp}}_{\Pi, A}(k) = 1] + (q_g + q_d)/2^\ell + 2 \cdot \Pr[\text{AskH}] \\ &= \Pr[\mathbf{Exp}_{\Pi, A}^{\text{cnmr-cca-0}}(k) = 1] + (q_g + q_d)/2^\ell + 2 \cdot \Pr[\text{AskH}]. \end{aligned}$$

Therefore,

$$\mathbf{Adv}_{\Pi, A}^{\text{nmr-cca}}(k) \leq (q_g + q_d)/2^\ell + 2 \cdot \Pr[\text{AskH}].$$

We next evaluate the probability  $\Pr[\text{AskH}]$  by constructing an algorithm  $B$  attacking  $\theta$ -partial one-wayness of  $\mathcal{TP}$ . In order to construct such an algorithm, we have to simulate the random oracles and the decryption oracle. To do this, we construct the modified experiment by applying the same modification from  $\text{Exp}_6$  to  $\text{Exp}_8$  in the proof of Theorem 2. We can apply Lemma 1 and a similar discussion in Theorem 2, and we can construct the  $\theta$ -partial inverting algorithm  $B$  for  $\mathcal{TP}$  such that

$$\Pr[\text{AskH}] \leq q_d(2q_g + q_d)/2^\ell + q_d(3q_f + 2q_d)/2^k + q_h \cdot \mathbf{Adv}_{\mathcal{TP}, B}^{\theta\text{-pow}}(k),$$

and get the claimed result.  $\square$

The message space of 3-round OAEP is common to each user. Therefore, from Theorems 1 and 3, 3-round OAEP is secure in the sense of SNMR-CCA in the random oracle model assuming that  $\mathcal{TP}$  is  $\theta$ -partial one-way where  $\theta = \frac{k}{k+\ell}$ .

## References

- [1] BELLARE, M., DESAI, A., POINTCHEVAL, D., AND ROGAWAY, P. Relations among Notions of Security for Public-Key Encryption Schemes. In Krawczyk [13], pp. 26–45.
- [2] BELLARE, M., AND ROGAWAY, P. Optimal Asymmetric Encryption – How to Encrypt with RSA. In *Advances in Cryptology – EUROCRYPT '94* (Perugia, Italy, May 1994), A. De Santis, Ed., vol. 950 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 92–111.
- [3] BELLARE, M., AND SAHAI, A. Non-malleable Encryption: Equivalence between Two Notions, and an Indistinguishability-Based Characterization. In *Advances in Cryptology – CRYPTO '99* (Santa Barbara, California, USA, August 1999), M. Wiener, Ed., vol. 1666 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 519–536.
- [4] BOSLEY, C., AND DODIS, Y. Does Privacy Require True Randomness? To appear in *Theory of Cryptography Conference – TCC 2007* (Amsterdam, The Netherlands, February 2007). <http://eprint.iacr.org/2006/283>.
- [5] CANETTI, R., AND KRAWCZYK, H. Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In *Advances in Cryptology – EUROCRYPT 2001* (Innsbruck, Austria, May 2001), B. Pfitzmann, Ed., vol. 2045 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 453–474.
- [6] CRAMER, R., AND SHOUP, V. A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In Krawczyk [13], pp. 13–25.
- [7] CRAMER, R., AND SHOUP, V. Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. In *Advances in Cryptology – EUROCRYPT 2002* (Amsterdam, The Netherlands, April 2002), L. Knudsen, Ed., vol. 2332 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 45–64.
- [8] DOLEV, D., DWORK, C., AND NAOR, M. Non-Malleable Cryptography. *SIAM Journal on Computing* 30, 2 (2000), 391–437.
- [9] FUJISAKI, E., OKAMOTO, T., POINTCHEVAL, D., AND STERN, J. RSA-OAEP is Secure under the RSA Assumption. In Kilian [12], pp. 260–274.

- [10] GOLDREICH, O. *Foundations of Cryptography: Volume II Basic Applications*. Cambridge University Press, 2004.
- [11] GOLDWASSER, S., AND MICALI, S. Probabilistic Encryption. *Journal of Computer and System Sciences* 28 (April 1984), 270–299.
- [12] KILIAN, J., Ed. *Advances in Cryptology – CRYPTO 2001* (Santa Barbara, California, USA, August 2001), vol. 2139 of *Lecture Notes in Computer Science*, Springer-Verlag.
- [13] KRAWCZYK, H., Ed. *Advances in Cryptology – CRYPTO ’98* (Santa Barbara, California, USA, August 1998), vol. 1462 of *Lecture Notes in Computer Science*, Springer-Verlag.
- [14] KRAWCZYK, H. HMQV: A High-Performance Secure Diffie-Hellman Protocol. In *Advances in Cryptology – CRYPTO 2005* (Santa Barbara, California, USA, August 2005), V. Shoup, Ed., vol. 3621 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 546–566.
- [15] LAUTER, K., AND MITYAGIN, A. Security Analysis of KEA Authenticated Key Exchange Protocol. In *PKC 2006 – 9th International Workshop on Theory and Practice in Public Key Cryptography* (New York, USA, April 2006), M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, Eds., vol. 3958 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 378–394.
- [16] OKAMOTO, T., AND POINTCHEVAL, D. REACT: Rapid Enhanced-Security Asymmetric Cryptosystem Transform. In *Topics in Cryptology – CT-RSA 2003* (San Francisco, CA, USA, April 2001), D. Naccache, Ed., vol. 2020 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 159–175.
- [17] PAILLIER, P. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *Advances in Cryptology – EUROCRYPT ’99* (Prague, Czech Republic, May 1999), J. Stern, Ed., vol. 1592 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 223–238.
- [18] PHAN, D. H., AND POINTCHEVAL, D. Chosen-Ciphertext Security without Redundancy. In *Advances in Cryptology – ASIACRYPT 2003* (Taipei, Taiwan, November 2003), C. S. Lai, Ed., vol. 2894 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 1–18.
- [19] POINTCHEVAL, D. Chosen-Ciphertext Security for Any One-Way Cryptosystem. In *PKC 2000 – 3rd International Workshop on Theory and Practice in Public Key Cryptography* (Melbourne, Victoria, Australia, January 2000), H. Imai and Y. Zheng, Eds., vol. 1751 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 129–146.
- [20] SHOUP, V. OAEP Reconsidered. In Kilian [12], pp. 239–259.

## A Proof of Lemma 1

The following proof is similar to that of IND-CCA2 by Phan and Pointcheval [18]. We define a sequence  $\text{Exp}_{7.1}$ ,  $\text{Exp}_{7.2}$ , etc., of modified attack experiments starting from the experiment  $\text{Exp}_7$ .

In the following, we only consider the difference between  $\Pr[\text{AskH}_7]$  and  $\Pr[\text{AskH}_8]$ . Note that the difference between  $\Pr[S_7]$  and  $\Pr[S_8]$  can be bounded in a similar way.

$\text{Exp}_{7.1}$ . In this experiment, we simulate the decryption oracle by using  $D$ -List which is initially set to empty.

- When the adversary makes a query  $c$  to the decryption oracle, if there exists a pair  $(m, c) \in D\text{-List}$  then we respond  $m$  as the plaintext. Otherwise, we compute the plaintext as follows and put  $(m, c)$  into  $D$ -List.

**D-1.** We first compute  $t||u \leftarrow \psi_{\text{sk}}(c)$ .

**D-2.** We next compute

$$h \leftarrow H(t), s \leftarrow u \oplus h, g \leftarrow G(s), r \leftarrow t \oplus g, f \leftarrow F(r), m \leftarrow s \oplus f,$$

and return  $m$ .

Since we can simulate the decryption oracle perfectly, we have  $\Pr[\text{AskH}_{7.1}] = \Pr[\text{AskH}_7]$ .

In the following, we modify the simulation **D-2** of the decryption oracle. We consider three cases with respect to the decryption query  $c$  by the adversary. Note that we can distinguish between the following three cases after computing  $t||u \leftarrow \psi_{\text{sk}}(c)$  in the simulation **D-1** of the decryption oracle.

**D-2-noT.**  $(t, h) \notin H\text{-List}$ , that is, the value  $t$  has not been queried to the oracle  $H$ .

**D-2-TnoS.**  $(t, h) \in H\text{-List}$  and  $(s, g) \notin G\text{-List}$  where  $s = u \oplus h$ , that is, the value  $t$  has been already queried to the oracle  $H$ , but the value  $s = u \oplus h$  has not been queried to the oracle  $G$ .

**D-2-TS.**  $(t, h) \in H\text{-List}$  and  $(s, g) \in G\text{-List}$ , that is, the values  $t$  and  $s = u \oplus h$  have been already queried to the oracles  $H$  and  $G$ , respectively.

In the following, we modify the above decryption process of each case.

**Exp<sub>7.2</sub>.** In this experiment, we modify the simulation of the decryption oracle in the case **D-2-noT**. That is, we respond the decryption query in the case **D-2-noT** as follows:

$$\begin{aligned} &\text{Set } h \stackrel{R}{\leftarrow} \{0, 1\}^\ell, s \leftarrow u \oplus h, g \stackrel{R}{\leftarrow} \{0, 1\}^k, r \leftarrow t \oplus g, f \leftarrow F(r), m \leftarrow s \oplus f. \\ &\text{Add } (s, g) \text{ to } G\text{-List and } (t, h) \text{ to } H\text{-List, and return } m. \end{aligned}$$

This makes difference only if  $s$  is already in  $G\text{-List}$ . Since  $t$  has not been queried to  $H$  in the case **D-2-noT**,  $h = H(t)$  is uniformly distributed. Therefore, the probability that  $s$  has already been queried to  $G$  is  $(q_g + q_d)/2^\ell$ . Summing up for all decryption queries, we get  $|\Pr[\text{AskH}_{7.2}] - \Pr[\text{AskH}_{7.1}]| \leq q_d(q_g + q_d)/2^\ell$ .

**Exp<sub>7.3</sub>.** In this experiment, we modify again the simulation of the decryption oracle in the case **D-2-noT** by not querying the oracle  $F$ :

$$\begin{aligned} &\text{Set } h \stackrel{R}{\leftarrow} \{0, 1\}^\ell, s \leftarrow u \oplus h, g \stackrel{R}{\leftarrow} \{0, 1\}^k, r \leftarrow t \oplus g, f \stackrel{R}{\leftarrow} \{0, 1\}^\ell, m \leftarrow s \oplus f. \\ &\text{Add } (r, f) \text{ to } F\text{-List, } (s, g) \text{ to } G\text{-List, and } (t, h) \text{ to } H\text{-List, and return } m. \end{aligned}$$

This makes difference only if  $r$  is already in  $F\text{-List}$ . Since  $g$  is uniformly distributed, the probability that  $r$  has already been queried to  $F$  is  $(q_f + q_d)/2^k$ . Summing up for all decryption queries, we get  $|\Pr[\text{AskH}_{7.3}] - \Pr[\text{AskH}_{7.2}]| \leq q_d(q_f + q_d)/2^k$ .

**Exp<sub>7.4</sub>.** In this experiment, we modify again the simulation of the decryption oracle in the case **D-2-noT** by answering as follows:

$$\begin{aligned} &\text{Set } m \stackrel{R}{\leftarrow} \{0, 1\}^\ell, h \stackrel{R}{\leftarrow} \{0, 1\}^\ell, s \leftarrow u \oplus h, g \stackrel{R}{\leftarrow} \{0, 1\}^k, r \leftarrow t \oplus g, f \leftarrow m \oplus s. \\ &\text{Add } (r, f) \text{ to } F\text{-List, } (s, g) \text{ to } G\text{-List, and } (t, h) \text{ to } H\text{-List, and return } m. \end{aligned}$$

Here, we first choose  $m$  and define  $f$  from  $m$ , while we first choose  $f$  and define  $m$  from  $f$  in **Exp<sub>7.3</sub>**. Two experiments **Exp<sub>7.4</sub>** and **Exp<sub>7.3</sub>** are perfectly indistinguishable and we have  $\Pr[\text{AskH}_{7.4}] = \Pr[\text{AskH}_{7.3}]$ .

Exp<sub>7.5</sub>. We now modify the simulation of the decryption oracle in the case **D-2-TnoS** by not calling either  $F$  or  $G$  as follows. In the case **D-2-TnoS**, the adversary asks  $c = \varphi_{\text{pk}}(t||u)$  such that  $h = H(t)$  is known, but  $s = u \oplus h$  has never been queried to  $G$ .

Set  $g \xleftarrow{R} \{0, 1\}^k$ ,  $r \leftarrow t \oplus g$ ,  $f \xleftarrow{R} \{0, 1\}^\ell$ ,  $m \leftarrow s \oplus f$ .  
Add  $(r, f)$  to  $F$ -List and  $(s, g)$  to  $G$ -List, and return  $m$ .

This makes difference only if  $r$  is already in  $F$ -List. Since  $g$  is uniformly distributed ( $s$  is not in  $G$ -List), the probability that  $r$  has already been queried to  $F$  is less than  $(q_f + q_d)/2^k$ . Summing up for all decryption queries, we get  $|\Pr[\text{AskH}_{7.5}] - \Pr[\text{AskH}_{7.4}]| \leq q_d(q_f + q_d)/2^k$ .

Exp<sub>7.6</sub>. We modify again the simulation of the decryption oracle in the case **D-2-TnoS** by answering as follows:

Set  $m \xleftarrow{R} \{0, 1\}^\ell$ ,  $g \xleftarrow{R} \{0, 1\}^k$ ,  $r \leftarrow t \oplus g$ ,  $f \leftarrow m \oplus s$ .  
Add  $(r, f)$  to  $F$ -List and  $(s, g)$  to  $G$ -List, and return  $m$ .

Here, we first choose  $m$  and define  $f$  from  $m$ , while we first choose  $f$  and define  $m$  from  $f$  in Exp<sub>7.5</sub>. Two experiments Exp<sub>7.6</sub> and Exp<sub>7.5</sub> are perfectly indistinguishable and we have  $\Pr[\text{AskH}_{7.6}] = \Pr[\text{AskH}_{7.5}]$ .

Exp<sub>7.7</sub>. In this experiment, we do not store either  $(s, g)$  in  $G$ -List or  $(r, f)$  in  $F$ -List, in the cases **D-2-noT** and **D-2-TnoS**. Instead of storing these elements, we add some process to the simulation of the random oracle  $G$ .

- In the case **D-2-noT**, the decryption oracle sets  $h \xleftarrow{R} \{0, 1\}^\ell$ ,  $m \xleftarrow{R} \{0, 1\}^\ell$ , adds  $(t, h)$  to  $H$ -List, and returns  $m$ .
- In the case **D-2-TnoS**, the decryption oracle returns  $m \xleftarrow{R} \{0, 1\}^\ell$ .
- In addition, whenever a pair  $(s, g)$  is added to  $G$ -List, the following process is executed: for any  $(t, h) \in H$ -List and any  $(m, c) \in D$ -List such that  $c = \varphi_{\text{pk}}(t||h \oplus s)$ , we compute  $r \leftarrow t \oplus g$ ,  $f \leftarrow m \oplus s$ , and add  $(r, f)$  to  $F$ -List.

Exp<sub>7.7</sub> and Exp<sub>7.6</sub> are perfectly indistinguishable unless  $r$  is asked to  $F$  before  $s$  is asked to  $G$ . In fact, if  $r$  is asked after  $s$ , at the moment that  $s$  is asked, by the above simulation, we will find  $(t, h)$  and therefore  $(r, f)$  is computed and added to  $F$ -List as in the Exp<sub>7.6</sub>.

Until  $s$  is asked to  $G$ ,  $g$  is a uniform variable, so is  $r$ . Therefore, the probability that  $r$  has been asked to  $F$  is  $q_f/2^k$ . Summing up for all decryption queries, we get  $|\Pr[\text{AskH}_{7.7}] - \Pr[\text{AskH}_{7.6}]| \leq q_d \cdot q_f/2^k$ .

Exp<sub>7.8</sub>. In this experiment, we modify again the simulation of the decryption oracle in the case **D-2-noT**. We do not store  $(t, h)$  in  $H$ -List and just answer  $m \leftarrow \{0, 1\}^\ell$  in the case **D-2-noT**. In the previous experiment, for the query  $h$  to  $H$ , we answer randomly  $h$ , so the adversary in the two experiments Exp<sub>7.8</sub> and Exp<sub>7.7</sub> cannot distinguish the answers of a query to  $H$ . Nevertheless,  $H$ -List has been changed and therefore, the answer for a query to  $F$  can be changed. We can see that the two experiments Exp<sub>7.8</sub> and Exp<sub>7.7</sub> are perfectly indistinguishable unless  $s$  is asked to  $G$  before  $t$  is asked to  $H$ . In fact, if  $s$  is asked to  $G$  after  $t$  is asked to  $H$ , at the moment that  $s$  is asked, by the above simulation, we will find  $(t, h)$  and therefore  $(r, f)$  is computed and added in  $F$ -List as in the Exp<sub>7.7</sub>.

Until  $t$  is asked to  $H$ ,  $h$  is a uniform variable, so is  $s = u \oplus h$ . Therefore, the probability that  $s$  has been asked to  $G$  is  $q_g/2^\ell$ . Summing up for all decryption queries, we get  $|\Pr[\text{AskH}_{7.8}] - \Pr[\text{AskH}_{7.7}]| \leq q_d \cdot q_g/2^\ell$ .

Exp<sub>7.9</sub>. We now complete the simulation of the decryption oracle by modifying the case **D-2-TS** and the step **D-1**. We do not ask any query to  $\psi_{\text{sk}}$  in **D-1**. Intuitively, if both  $t$  and  $s$  have been

asked, we can easily find them, and can return  $m$ . Otherwise, we give a random answer as in the previous experiment.

- In **D-1**, for the decryption query  $c$ , we look up for  $(t, h) \in H\text{-List}$  and  $(s, g) \in G\text{-List}$  such that  $\varphi_{\text{pk}}(t, s \oplus h) = c$ . If the record is found, we define  $u \leftarrow s \oplus h$ . Otherwise, we set  $t \leftarrow \perp$  and  $u \leftarrow \perp$ .
- In the case **D-2-TS**, we compute  $r \leftarrow t \oplus g$ ,  $f \leftarrow F(r)$ , and  $m \leftarrow s \oplus f$ , and return  $m$ .

The two experiments  $\text{Exp}_{7.9}$  and  $\text{Exp}_{7.8}$  are perfectly indistinguishable. In fact, in the first case nothing is modified, and in the second case by making  $t \leftarrow \perp$  and  $u \leftarrow \perp$ , the answer of the decryption oracle for the query  $c$  will be a random  $m$  as in the previous experiment. Thus,  $\Pr[\text{AskH}_{7.9}] = \Pr[\text{AskH}_{7.8}]$ .

Since  $\text{Exp}_{7.9}$  is identical to  $\text{Exp}_8$ , we have

$$\begin{aligned}
& \Pr[\text{AskH}_7] \\
&= \Pr[\text{AskH}_{7.1}] \\
&\leq q_d(q_g + q_d)/2^\ell + \Pr[\text{AskH}_{7.2}] \\
&\leq q_d(q_g + q_d)/2^\ell + q_d(q_f + q_d)/2^k + \Pr[\text{AskH}_{7.3}] \\
&= q_d(q_g + q_d)/2^\ell + q_d(q_f + q_d)/2^k + \Pr[\text{AskH}_{7.4}] \\
&\leq q_d(q_g + q_d)/2^\ell + 2q_d(q_f + q_d)/2^k + \Pr[\text{AskH}_{7.5}] \\
&= q_d(q_g + q_d)/2^\ell + 2q_d(q_f + q_d)/2^k + \Pr[\text{AskH}_{7.6}] \\
&\leq q_d(q_g + q_d)/2^\ell + 2q_d(q_f + q_d)/2^k + q_dq_f/2^k + \Pr[\text{AskH}_{7.7}] \\
&\leq q_d(q_g + q_d)/2^\ell + 2q_d(q_f + q_d)/2^k + q_dq_f/2^k + q_dq_g/2^\ell + \Pr[\text{AskH}_{7.8}] \\
&= q_d(q_g + q_d)/2^\ell + 2q_d(q_f + q_d)/2^k + q_dq_f/2^k + q_dq_g/2^\ell + \Pr[\text{AskH}_{7.9}] \\
&= q_d(2q_g + q_d)/2^\ell + q_d(3q_f + 2q_d)/2^k + \Pr[\text{AskH}_8].
\end{aligned}$$

Note that we can bound  $\Pr[S_7]$  in a similar way, and get

$$\Pr[S_7] \leq q_d(2q_g + q_d)/2^\ell + q_d(3q_f + 2q_d)/2^k + \Pr[S_8].$$

We now estimate the time for simulating the decryption oracle, as well as the random oracles. The running time for simulating these oracles is bounded by  $q_g \cdot q_h \cdot T_\varphi + q_d \cdot T_{lu}$  where  $T_\varphi$  denotes the time for evaluating the permutation  $\varphi_{\text{pk}}$  and  $T_{lu}$  denotes the time for looking up in a list. We can indeed perform the simulation granted an additional list  $GH\text{-List}$  which contains all the tuples  $(t, h, s, g, c)$  where  $(t, h) \in H\text{-List}$ ,  $(s, g) \in G\text{-List}$ , and  $c = \varphi_{\text{pk}}(t, s \oplus h)$ .