Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science 171 (2007) 83–91
Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

On the Security of Tag-KEM for Signcryption

Maki Yoshida [1]

Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka 560-0871, Japan

Toru Fujiwara [2]

Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka 560-0871, Japan

Abstract

Signcryption tag-KEM (Key Encapsulation Mechanism with a tag) is an authenticated tag-KEM for generic construction of hybrid signcryption. Signcryption tag-KEM allows the sender to encapsulate a symmetric key along with a tag so that the receiver can authenticate the sender, the key, and the tag. We present a definition for the security of signcryption tag-KEM which is suitable for a recent signcryption setting. We also present a proof of security for the previous generic construction of hybrid signcryption according to the given definition.

Keywords: key encapsulation, privacy, authentication, signcryption, signcryption tag-KEM

1 Introduction

Encryption and signature schemes are fundamental cryptographic tools for providing privacy and authenticity, respectively, in the public-key setting. Both privacy and authenticity are simultaneously needed in many applications on ad-hoc network where anyone can freely join or leave the network. This issue seems to be easily solved by composing a signature and encryption. However, it was noticed by [2] that such multi-user setting opens a possibility for some subtle attacks, not presented in the settings of stand-alone signature and encryption. Thus, a simple composition does not necessarily yield desired properties.

A signcryption was introduced by Zheng [8] as a primitive which simultaneously provides both of privacy and authenticity. There are many works of signcryption [2,7,6,3]. An et al. [2] addressed proper modelling of signcryption. Then, Dodis

[1] Email: maki-yos@ist.osaka-u.ac.jp

[2] Email: fujiwara@ist.osaka-u.ac.jp

doi:10.1016/j.entcs.2006.11.011

et al. [7,6] modified the definition for security more reasonably in the multi-user setting. Recently, Dent and Bjørstad [3] presented a tag-KEM/DEM framework for generic construction of hybrid signcryption. The original tag-KEM/DEM [1] was introduced for generic construction of hybrid *encryption*. The framework combines tag-KEM (Key Encapsulation Mechanism with a tag) and DEM (Data Encryption Mechanism). A tag-KEM uses asymmetric technique to encrypt a symmetric key along with a tag, while the DEM uses a symmetric cipher to encrypt the message payload using the key from the KEM.

Dent and Bjørstad [3] defined an authenticated tag-KEM for hybrid signcryption, called signcryption tag-KEM, as a primitive which simultaneously satisfies chosen ciphertext security for privacy and strong existential unforgeability for authenticity. Moreover, they showed that adapting the tag-KEM/DEM construction of hybrid encryption [1] to signcryption yields simpler scheme descriptions and better generic security reductions than the previous works.

However, the security of signcryption tag-KEM is defined for the previous definition for security of signcryption [2] which restricts the adversary so that the adversary is allowed to access de-signcryption oracle (resp. signcryption oracle) for the attacked user but not signcryption oracle (resp. de-signcryption oracle) for the attacked user if the adversary attacks privacy (resp. authenticity). On the other hand, the modified definition [7,6] allows the adversary to access both oracles, irrespective of whether the adversary is attacking privacy or authenticity (such attack is called *simultaneous attack* [6]). This means that the modified definition for security of signcryption becomes stronger than the previous one and the security of signcryption tag-KEM in [3] is not enough strong for yielding the modified security of signcryption.

Our Contribution. We define security of signcryption tag-KEM for the modified definition of signcryption [7,6] which allows simultaneous attacks. Specifically, the adversary is allowed to access all oracles corresponding to signcryption oracle and de-signcryption oracle for the attacked user. When addressing security of signcryption, there are two formalizations. One assumes that the adversary is an outsider who only knows the public information. Such security is called *Outsider security*. The other, stronger notion, assumes that the adversary is a legal user of the system. Such security is called *Insider security*. In this paper, we consider the stronger notion, i.e., Insider security. Then, we prove that the new definition for security of signcryption tag-KEM also yields the modified definition for security of signcryption with the same security reductions as in [3] by using the tag-KEM/DEM construction of hybrid signcryption in [3].

2 Preliminaries

We review the definitions of signcryption in [7] and DEM in [1].

2.1 Signcryption

Syntax. A signcryption is defined as a three-tuple of polynomial-time algorithms:

- $SC.Gen(1^k)$, a key generation algorithm, takes as input a security parameter 1^k, and outputs a keypair (sk, pk) where sk is the user's secret key and pk is the user's public key. Without loss of generality, it is assumed that pk is determined from sk. The public key pk defines all relative spaces, i.e., space for messages denoted by $\mathcal{M}$. Note, that in the signcryption setting all participating parties need to invoke $SC.Gen$. For a user P, denote its keys by sk_P and pk_P.
- $SEnc(sk_S, pk_R, m)$, a signcryption algorithm, takes as input the sender's secret key sk_S, the receiver's public key pk_R, and a message m. It returns a signcryption SC.
- $VDec(sk_R, pk_S, SC)$, a de-signcryption algorithm, takes as input the receiver's secret key sk_R, the sender's public key pk_S, and a signcryption SC. It outputs a message m or the unique error symbol $\perp$ in case SC is "invalid."

Completeness. For any sender S, any receiver R, and any message $m \in \mathcal{M}$, if $(sk_S, pk_S) \leftarrow SC.Gen(1^k)$, $(sk_R, pk_R) \leftarrow SC.Gen(1^k)$, and $C \leftarrow SEnc(sk_S, pk_R, m)$, then $VDec(sk_R, pk_S, C) = m$.

Insider security. Insider-secure signcryption protects a given user U even if his partner might be malicious. For privacy, if honest user S sent a signcryption to U and later exposed his key to the adversary, the adversary still cannot decrypt the signcryption. For authenticity, without U's secret key, the adversary cannot forge signcryption from U to another user R, even with R's secret key.

When addressing the security, we deal with two issues: Security goal and attack model. In [7], for privacy and authenticity, a common type of security goal and attack model is considered, called indistinguishability (IND)/strong existential unforgeability (abbreviated as sUF, sEUF, or sEF) and chosen ciphertext attack (CCA2)/chosen message attack (CMA), respectively. We denote the resulting security notion by IND-CCA2/sUF-CMA. The notation follows [2,7,3].

The security IND-CCA2 requires that any probabilistic polynomial time adversary should be unable to find any pair (m_0, m_1) for which he can distinguish $SEnc(sk_S, pk_U, m_0)$ from $SEnc(sk_S, pk_U, m_1)$, with adaptive access to the following two oracles for the attacked user U corresponding to each of $SEnc$ and $VDec$.

- $\mathcal{O}_{SE,U}$, the signcryption oracle, takes as input any user's (receiver's) public key pk and any message m. It returns $SEnc(sk_U, pk, m)$,
- $\mathcal{O}_{VD,U}$, the de-signcryption oracle, takes as input any user's (sender's) public key pk and a signcryption SC. It returns $VDec(sk_U, pk, SC)$.

Allowing access to oracle $\mathcal{O}_{SE,U}$ is a main difference from the previous definition [2].

To create "valid" signcryptions that the adversary must distinguish between, he outputs the secret key sk_S of the user S sending messages to U. This means that even when compromising S, the adversary is still unable to understand messages S sent to U. Let $\mathcal{A}_{SC,cca}$ be a probabilistic polynomial time oracle machine that plays the following game.

(i) $(sk_U, pk_U) \leftarrow SC.Gen(1^k)$.

(ii) $(m_0, m_1, sk_S, v) \leftarrow \mathcal{A}_{SC,cca}^{\mathcal{O}_{SE,U},\mathcal{O}_{VD,U}}(pk_U)$.

(iii) $b \leftarrow \{0,1\}$.

(iv) $SC \leftarrow SEnc(sk_S, pk_U, m_b)$.

(v) $b' \leftarrow \mathcal{A}_{SC,cca}^{\mathcal{O}_{SE,U},\mathcal{O}_{VD,U}}(v, SC)$.

In Step (iv), $\mathcal{A}_{SC,cca}$ is restricted not to ask (pk_S, SC) to de-signcryption oracle $\mathcal{O}_{VD,U}$, but can still use, for example, $(pk_{S'}, SC)$ for $pk_{S'} \neq pk_S$. Variable v is state information of the adversary. $\mathcal{A}_{SC,cca}$ is considered successful only if $b = b'$. We define $Adv_{\mathcal{A}_{SC,cca}} = \left|Pr[b = b'] - \frac{1}{2}\right|$ and $\epsilon_{SC,cca} = \max_{\mathcal{A}_{SC,cca}}(Adv_{\mathcal{A}_{SC,cca}})$ where maximum is taken over all machines. We say that a signcryption is IND-CCA2-secure if $\epsilon_{SC,cca}$ is negligible in k.

The security sUF-CMA requires that any probabilistic polynomial time adversary should not only be able to generate a "valid" signcryption SC of some message m from U to any user R, with adaptive access to the above two oracles. Allowing access to oracle $\mathcal{O}_{VD,U}$ is a main difference from the previous definition [2]. In order to define "valid," the adversary is allowed to come up with the presumed secret key sk_R as part of his forgery. Let $\mathcal{A}_{SC,cma}$ be a probabilistic polynomial time oracle machine that plays the following game.

(i) $(sk_U, pk_U) \leftarrow SC.Gen(1^k)$.

(ii) $(SC, sk_R) \leftarrow \mathcal{A}_{SC,cma}^{\mathcal{O}_{SE,U},\mathcal{O}_{VD,U}}(pk_U)$.

In Step (ii), $\mathcal{A}_{SC,cma}$ is restricted not to obtain SC in response to any $SEnc(sk_U, pk_R, m)$ query, i.e., any $\mathcal{O}_{SE,U}$ query on the public key pk_R and some message m. $\mathcal{A}_{SC,cma}$ is considered successful only if $VDec(sk_R, pk_U, SC) \neq \perp$. We define $Adv_{\mathcal{A}_{SC,cma}}$ as the probability it succeeds, and $\epsilon_{SC,cma} = \max_{\mathcal{A}_{SC,cma}}(Adv_{\mathcal{A}_{SC,cma}})$ where maximum is taken over all machines. We say that a signcryption is sUF-CMA-secure if $\epsilon_{SC,cma}$ is negligible in k.

2.2 DEM

Syntax. A DEM is defined as a tuple of two polynomial-time algorithms (Enc, Dec) associated to (symmetric) key-space $\mathcal{K}_D$ defined by the security parameter k. We consider $\mathcal{K}_D$ is $\{0,1\}^k$ and message space is $\{0,1\}^*$. Here we omit the detail description of these algorithms except the minimum syntax description, $Enc(dk, m) = C$ and $Dec(dk, C) = m$.

Completeness. For any symmetric key $dk \in \mathcal{K}_D$ and any message $m \in \{0,1\}^*$, $Dec(dk, Enc(dk, m)) = m$.

Security. For the purposes of this paper, we only require passive security for DEM as [3]. Let $\mathcal{A}_D$ be a probabilistic polynomial time machine that plays the following game.

(i) $(m_0, m_1, v) \leftarrow \mathcal{A}_D(1^k)$

(ii) $K \leftarrow \mathcal{K}_D$, $\xi \leftarrow \{0,1\}$, $C \leftarrow Enc(K, m_\xi)$.

(iii) $\xi' \leftarrow \mathcal{A}_D(v, C)$.

We define $Adv_{\mathcal{A}_D} = \left|Pr[\xi = \xi'] - \frac{1}{2}\right|$ and $\epsilon_D = \max_{\mathcal{A}_D}(Adv_{\mathcal{A}_D})$ where maximum is taken over all machines. We say that DEM is one-time secure if ϵ_D is negligible in k.

3 New Definition for Insider Security

We now define the notion of Insider security for signcryption tag-KEM. In the previous model of signcryption tag-KEM in [3], the sender's key and the receiver's key are generated by different algorithms while in a general model of signcryption the keys are generated by the same algorithm. We follow the general model of signcryption.

3.1 Syntax of Signcryption Tag-KEMs

A signcryption tag-KEM is defined as a four-tuple of polynomial-time algorithms:

- $SCTK.Gen(1^k)$, a key generation algorithm, takes as input a security parameter 1^k, and outputs a pair of keys (sk, pk) where sk is the user's secret key and pk the user's public key. Without loss of generality, we assume that pk is determined from sk. The public key pk defines all relative spaces, i.e., spaces for tags and encapsulated keys denoted by $\mathcal{T}$ and $\mathcal{K}_K$, respectively.
- $Sym(sk_S, pk_R)$, a symmetric key generation algorithm, takes as input the secret key of the sender sk_S and the public key of the receiver pk_R. It outputs a symmetric key $K \in \mathcal{K}_D$ together with internal state information ω where $\mathcal{K}_D$ is the key space of DEM.
- $SEncap(\omega, \tau)$, a key encapsulation algorithm, takes as input the state information ω and an arbitrary tag τ. It returns an encapsulation E.
- $VDecap(sk_R, pk_S, E, \tau)$, a decapsulation algorithm, takes as input the receiver's secret key sk_R, the sender's public key pk_S, an encapsulation E, and a tag τ. It outputs a symmetric key K or the unique error symbol $\perp$ in case E is "invalid."

Completeness. For any sender S, any receiver R, and any tag $\tau \in \mathcal{T}$, if $(sk_S, pk_S) \leftarrow SCTK.Gen(1^k)$, $(sk_R, pk_R) \leftarrow SCTK.Gen(1^k)$, and $(K, \omega) \leftarrow Sym(sk_S, pk_R)$, then $VDecap(sk_R, pk_S, SEncap(\omega, \tau), \tau) = K$.

3.2 Insider Security of Signcryption tag-KEM

The definition for Insider security of signcryption and signcryption tag-KEM in [3] neither mention nor allow simultaneous attacks. Therefore, we define the notion of Insider security so that the adversary has access to all oracles associated with not only $\mathcal{O}_{SE,U}$ but also $\mathcal{O}_{VD,U}$.

We consider IND/sUF and CCA2/CMA as security goal and attack model, respectively. We denote the resulting security notion by IND-CCA2/sUF-CMA.

The security IND-CCA2 requires that any probabilistic polynomial time adversary should be unable to distinguish whether a given K is the one embedded in an encapsulation (E, τ) or not, with adaptive access to three oracles for the attacked user U corresponding to each of Sym, $SEncap$, and $VDecap$.

- $\mathcal{O}_{S,U}$, the symmetric key generation oracle, takes as input any user's (receiver's) public key pk, runs $Sym(sk_U, pk)$, and obtains (K, ω). It then stores the value ω (hidden from the view of the adversary, and overwriting any previously stored values), and returns the symmetric key K.
- $\mathcal{O}_{E,U}$, the key encapsulation oracle, takes as input an arbitrary tag τ, and checks whether there exists a stored value ω. If there is not, it returns $\perp$ and terminates. Otherwise it erases the value from storage, and returns $SEncap(\omega, \tau)$.
- $\mathcal{O}_{D,U}$, the key decapsulation oracle, takes as input any user's (sender's) public key pk, an encapsulation E, and a tag τ. It returns $VDecap(sk_U, pk, E, \tau)$.

Note that oracles $\mathcal{O}_{S,U}$ and $\mathcal{O}_{E,U}$ correspond to $\mathcal{O}_{SE,U}$, and oracle $\mathcal{O}_{D,U}$ corresponds to $\mathcal{O}_{VD,U}$. Allowing access to oracles $\mathcal{O}_{S,U}$ and $\mathcal{O}_{E,U}$ is a main difference from the previous definition [3].

To create "valid" encapsulations that the adversary must distinguish between, he outputs the secret key sk_S of the user S embedding a key for U same as the adversary against signcryption in [7]. This means that even when compromising S, the adversary is still unable to understand keys S embedded for U. Let $\mathcal{A}_{SCTK,cca}$ be a probabilistic polynomial time oracle machine that plays the following game.

(i) $(sk_U, pk_U) \leftarrow SCTK.Gen(1^k)$.

(ii) $(sk_S, v_1) \leftarrow \mathcal{A}_{SCTK,cca}^{\mathcal{O}_{S,U},\mathcal{O}_{E,U},\mathcal{O}_{D,U}}(pk_U)$.

(iii) $(K_1, \omega) \leftarrow Sym(sk_S, pk_U)$, $K_0 \leftarrow \mathcal{K}_D$, $\delta \leftarrow \{0,1\}$.

(iv) $(\tau, v_2) \leftarrow \mathcal{A}_{SCTK,cca}^{\mathcal{O}_{S,U},\mathcal{O}_{E,U},\mathcal{O}_{D,U}}(v_1, K_\delta)$.

(v) $E \leftarrow Encap(\omega, \tau)$.

(vi) $\delta' \leftarrow \mathcal{A}_{SCTK,cca}^{\mathcal{O}_{S,U},\mathcal{O}_{E,U},\mathcal{O}_{D,U}}(v_2, E)$.

In Step (vi), $\mathcal{A}_{SCTK,cca}$ is restricted not to ask (pk_S, E, τ) to the decapsulation oracle $\mathcal{O}_{D,U}$, but can still use, for example, $(pk_{S'}, E, \tau')$ for $pk_{S'} \neq pk_S$ or $\tau' \neq \tau$. Variables v_1, v_2 are state information of the adversary. $\mathcal{A}_{SCTK,cca}$ is considered successful only if $\delta = \delta'$. We define IND-CCA2-security similarly with the definition for $Adv_{\mathcal{A}_{SCTK,cca}} = \left|Pr[\delta = \delta'] - \frac{1}{2}\right|$ and $\epsilon_{SCTK,cca} = \max_{\mathcal{A}_{SCTK,cca}}(Adv_{\mathcal{A}_{SCTK,cca}})$.

The security sUF-CMA requires that any probabilistic polynomial time adversary should not be able to generate a "valid" encapsulation E from U to any user R, with adaptive access to the three oracles. Allowing access to oracle $\mathcal{O}_{D,U}$ is a main difference from the previous definition [3]. In order to define "valid," we also allow the adversary to come up with the presumed secret key sk_R as part of his forgery same as the adversary against signcryption in [7]. Let $\mathcal{A}_{SCTK,cma}$ be a probabilistic polynomial time oracle machine that plays the following game.

(i) $(sk_U, pk_U) \leftarrow SCTK.Gen(1^k)$.

(ii) $(E, \tau, sk_R) \leftarrow \mathcal{A}_{SCTK,cma}^{\mathcal{O}_{S,U},\mathcal{O}_{E,U},\mathcal{O}_{D,U}}(pk_U)$.

In Step (ii), $\mathcal{A}_{SCTK,cma}$ is restricted not to obtain E in response to any $SEncap(\omega, \tau)$

query with $(K, \omega) \leftarrow Sym(sk_U, pk_R)$, i.e., any $\mathcal{O}_{E,U}$ query on the tag τ and a value ω stored by an $\mathcal{O}_{S,U}$ query on the public key pk_R. $\mathcal{A}_{SCTK,cma}$ is considered successful only if $VDecap(sk_R, pk_U, E, \tau) \neq \perp$. We define sUF-CMA-security similarly with the definition for $Adv_{\mathcal{A}_{SCTK,cma}}$, the probability $\mathcal{A}_{SCTK,cma}$ succeeds, $\epsilon_{SCTK,cma} = \max_{\mathcal{A}_{SCTK,cma}}(Adv_{\mathcal{A}_{SCTK,cma}})$.

4 Generic Construction of Hybrid Signcryption and Its Security Proof

We review the generic construction of hybrid signcryption in [3], and prove that the new definition for security of signcryption tag-KEM yields the modified definition for security of signcryption in [7].

4.1 Generic Construction of Hybrid Signcryption

The construction is based on the same idea of the generic construction of hybrid asymmetric encryption proposed in [1].

- $SC.Gen(1^k)$
  $(sk, pk) \leftarrow SCTK.Gen(1^k)$. Output (sk, pk).
- $SEnc(sk_S, pk_R, m)$
  (i) $(K, \omega) \leftarrow Sym(sk_S, pk_R)$.
  (ii) $C \leftarrow Enc(K, m)$.
  (iii) $E \leftarrow SEncap(\omega, C)$.
  (iv) $SC \leftarrow (E, C)$. Output SC.
- $VDec(sk_R, pk_S, SC)$
  (i) $SC \leftarrow (E, C)$.
  (ii) If $\perp \leftarrow VDecap(sk_R, pk_S, E, C)$, output $\perp$ and terminate.
  (iii) Otherwise $K \leftarrow VDecap(sk_R, pk_S, E, C)$.
  (iv) $m \leftarrow Dec(K, C)$. Output m.

4.2 Proof of the Security

We now turn to proving security of the hybrid signcryption scheme.

Theorem 4.1 *If signcryption tag-KEM is IND-CCA2 secure and DEM is one-time secure, then the hybrid signcryption scheme in Section 4.1 is IND-CCA2 secure where* $\epsilon_{SC,cca} \leq 2\epsilon_{SCTK,cca} + \epsilon_D$. *Moreover, if signcryption tag-KEM is sUF-CMA secure, then the hybrid signcryption scheme in Section 4.1 is sUF-CMA secure where* $\epsilon_{SC,cma} \leq \epsilon_{SCTK,cma}$.

The proof is almost the same as the one for the previous definition for security. We present here the proof for the IND-CCA2 security. The IND-CCA2 security of hybrid signcryption is proven in the same way as the IND-CCA2 security of hybrid *encryption* in [1].

Proof. Let Game 0 be the regular IND-CCA2 game for signcryption.

(i) $(sk_U, pk_U) \leftarrow Gen(1^k)$.

(ii) $(m_0, m_1, sk_S, v) \leftarrow A_{SC,cca}^{\mathcal{O}_{SE,U},\mathcal{O}_{VD,U}}(pk_U)$.

(iii) $b \leftarrow \{0,1\}$.

(iv) $(\omega, K) \leftarrow Sym(sk_S, pk_U)$, $C \leftarrow Enc(K, m_b)$, $E \leftarrow SEncap(\omega, C)$, $SC \leftarrow (E, C)$.

(v) $b' \leftarrow \mathcal{A}_{SC,cca}^{\mathcal{O}_{SE,U},\mathcal{O}_{VD,U}}(v, SC)$.

In the following game, the game is altered to use a random key when generating the challenge signcryption, rather than the real key output by Sym. We refer to the resulting game as Game 1.

(i) $(sk_U, pk_U) \leftarrow Gen(1^k)$.

(ii) $(m_0, m_1, sk_S, v) \leftarrow A_{SC,cca}^{\mathcal{O}_{SE,U},\mathcal{O}_{VD,U}}(pk_U)$.

(iii) $b \leftarrow \{0,1\}$.

(iv) $(\omega, K) \leftarrow Sym(sk_S, pk_U)$, $K' \leftarrow \mathcal{K}_D$, $C \leftarrow Enc(K', m_b)$, $E \leftarrow SEncap(\omega, C)$, $SC \leftarrow (E, C)$.

(v) $b' \leftarrow \mathcal{A}_{SC,cca}^{\mathcal{O}_{SE,U},\mathcal{O}_{VD,U}}(v, SC)$.

Let X_0 and X_1 be the events that $b = b'$ in Game 0 and Game 1, respectively. We can prove the theorem by showing that $|Pr[X_1] - Pr[X_0]| \leq 2\epsilon_{SCTK,cca}$ and $|Pr[X_1] - \frac{1}{2}| \leq \epsilon_D$. The proof of both inequality are slightly different from the previous proof because of access to $\mathcal{O}_{D,U}$. In the following, we prove the former inequality by constructing $\mathcal{A}_{SCTK,cca}$ that attacks the underlying signcryption tag-KEM by using $\mathcal{A}_{SC,cca}$. First, $\mathcal{A}_{SCTK,cca}$ is given public key pk_U and passes it to $\mathcal{A}_{SC,cca}$. Given m_0 and m_1 from $\mathcal{A}_{SC,cca}$, $\mathcal{A}_{SCTK,cca}$ requests a challenge K_δ of the game. $\mathcal{A}_{SCTK,cca}$ then selects $b \leftarrow \{0,1\}$ and computes $C = Enc(K_\delta, m_b)$. By sending oracle $\mathcal{O}_{E,U}$ C as a tag, $\mathcal{A}_{SCTK,cca}$ receives E and sends signcryption (E, C) to $\mathcal{A}_{SC,cca}$. For every query from $\mathcal{A}_{SC,cca}$ to oracle $\mathcal{O}_{SE,U}$, $\mathcal{A}_{SCTK,cca}$ sends oracle $\mathcal{O}_{S,U}$ the public key, and uses the returned key, Enc, and oracle $\mathcal{O}_{E,U}$ in the same way. Every query from $\mathcal{A}_{SC,cca}$ to oracle $\mathcal{O}_{VD,U}$ is forwarded to oracle $\mathcal{O}_{D,U}$. If $\perp$ is returned, it is forwarded to $\mathcal{A}_{SC,cca}$. Otherwise, $\mathcal{A}_{SCTK,cca}$ decrypts E by using the key given from $\mathcal{O}_{D,U}$, and passes the resulting message to $\mathcal{A}_{SC,cca}$. When $\mathcal{A}_{SC,cca}$ outputs $b' = b$, $\mathcal{A}_{SCTK,cca}$ outputs $\delta' = 1$ meaning that K_δ is the real key. Otherwise, $\mathcal{A}_{SCTK,cca}$ outputs $\delta' = 0$ meaning that K_δ is random. We can prove that the view of $\mathcal{A}_{SC,cca}$ is identical to that in Game 0 when $\delta = 1$, and that in Game 1 when $\delta = 0$. Accordingly, $Pr[b' = b|\delta = 1] = Pr[X_0]$ and $Pr[b' = b|\delta = 0] = Pr[X_1]$. Therefore, it holds

$$\begin{aligned} Pr[\delta' = \delta] - \frac{1}{2} &= \frac{1}{2}(Pr[\delta' = 1|\delta = 1] - Pr[\delta' = 1|\delta = 0]) \\ &= \frac{1}{2}(Pr[b' = b|\delta = 1] - Pr[b' = b|\delta = 0]) \\ &= \frac{1}{2}(Pr[X_0] - Pr[X_1]). \end{aligned}$$

Since $|Pr[\delta' = \delta] - \frac{1}{2}| \leq \epsilon_{SCTK,cca}$, we have $|Pr[X_1] - Pr[X_0]| \leq 2\epsilon_{SCTK,cca}$. □

5 Conclusion

We have presented the security notion for signcryption tag-KEM and proven security of the previous construction of hybrid signcryption according to the given definition. One of the future works is to present a construction of optimal signcryption tag-KEM.

Acknowledgment

The authors would like to thank Dr. Alexander W. Dent for his helpful comments and suggestions.

References

[1] Abe, M., R. Gennaro, K. Kurosawa, and V. Shoup, *Tag-KEM/DEM: A New Framework for Hybrid Encryption and a New Analysis of Kurosawa-Desmedt KEM*, EUROCRYPT 2005, LNCS **3494** (2005), 128–146.

[2] An, J.H., Y. Dodis, and T. Rabin, *On the Security of Joint Signature and Encryption*, EUROCRYPT 2002, LNCS **2332** (2002), 83–107.

[3] Bjørstad, T.E., and A.W. Dent, *Building Better Signcryption Schemes with Tag-KEMs*, URL: http://eprint.iacr.org/2005/405.pdf.

[4] Baek, J., R. Steinfeld and Y. Zheng, *Formal Proofs for the Security of Signcryption*, PKC 2002, LNCS **2274** (2002), 80–98.

[5] Chevallier-Mames, B., *An Efficient CDH-based Signature Schemes with a Tight Security Reduction*, CRYPTO 2005, LNCS **3621** (2005), 511-526.

[6] Dodis, Y., *Signcryption (Short Survey)*, Encyclopedia of Cryptography and Security, Springer Verlag, 2005.

[7] Dodis, Y., M.J. Freedman, S. Jarecki, and S. Walfish, *Optimal Signcryption from Any Trapdoor Permutation*, URL: http://eprint.iacr.org/2004/020.pdf.

[8] Zheng, Y., *Digital Signcryption or How to Achieve Cost(Signature & Encryption) << Cost(Signature) + Cost(Encryption)*, Crypto'97, LNCS **1294** (1997), 165–179.