Building Better Signcryption Schemes with Tag-KEMs

Tor E. Bjørstad[1] and Alexander W. Dent[2]

[1] The Selmer Center, Department of Informatics,
University of Bergen, Norway
[2] Royal Holloway, University of London,
Egham, Surrey, U.K.
tor.bjorstad@ii.uib.no a.dent@rhul.ac.uk

Abstract. Signcryption schemes aim to provide all of the advantages of simultaneously signing and encrypting a message. Recently, Dent [8, 9] and Bjørstad [4] investigated the possibility of constructing provably secure signcryption schemes using hybrid KEM-DEM techniques [7]. We build on this work by showing that more efficient insider secure hybrid signcryption schemes can be built using tag-KEMs [1]. To prove the effectiveness of this construction, we will provide several examples of secure signcryption tag-KEMs, including a brand new construction based on the Chevallier-Mames signature scheme [5] which has the tightest known security reductions for both confidentiality and unforgeability.

1 Introduction

The signcryption primitive was introduced by Zheng in 1997 [13] to study asymmetric schemes that offer most or all the benefits provided by public-key encryption and signature schemes. Signcryption schemes must provide message authenticity, confidentiality and integrity, and may also offer a way to provide non-repudiation. As such, a signcryption scheme provides a secure, authenticated channel for message transmission. Although Zheng only considered schemes that are more computationally efficient than a direct composition of encryption and signature schemes, the definition of signcryption is normally expanded to include any asymmetric scheme that provides this functionality, regardless of efficiency. Direct composition of public-key encryption and signatures has been studied by An *et. al.* [2].

In order to obtain efficient encryption schemes in practice, hybrid techniques are commonly used. The practice of combining symmetric and asymmetric schemes to encrypt and transmit long messages efficiently has been common knowledge for many years. However, formal analysis was first performed by Cramer and Shoup in the late 1990s [7]. The usual construction paradigm, known as the KEM-DEM construction, consists of two parts: a key encapsulation mechanism (KEM) and a data encapsulation mechanism (DEM). The KEM uses asymmetric techniques to encrypt a symmetric key, while the DEM uses a symmetric cipher to encrypt the message payload using the key from the KEM.

The main benefit of the KEM-DEM construction paradigm is that the security of KEM and DEM may be analyzed separately.

The use of hybrid techniques to build signcryption schemes has been studied by Dent [8–10] and Bjørstad [4]. This has provided a useful perspective for analysis of those classes of signcryption schemes that use hybrid techniques. However, previous efforts have yielded complex verification-decryption (unsigncryption) algorithms, stemming from the need to verify a link between message, key and encapsulation. This article will examine a way to simplify the hybrid construction through use of tag-KEMs [1]. We show that adapting the tag-KEM + DEM construction to signcryption yields simpler scheme descriptions and better generic security reductions than previous efforts.

To demonstrate the usefulness of this new paradigm, we construct several signcryption schemes based on signcryption tag-KEMs. The first is a simple modification of Zheng's original signcryption scheme [13]. This scheme has become baseline standard for judging the efficiency and security of any new signcryption scheme or construction method. The second is a new signcryption scheme based on the Chevallier-Mames signature scheme [5]. As far as the authors are aware, this new signcryption scheme has the tightest known security bounds.

2 Preliminaries

2.1 Signcryption

The signcryption primitive was introduced in 1997 by Zheng [13].

Definition 1 (Signcryption). *A signcryption scheme* $\mathcal{SC} = (Com,\ Key_S,\ Key_R,\ SC,\ USC)$ *is defined as tuple of five algorithms.*

- *A probabilistic common parameter generation algorithm, Com. It takes as input a security parameter* 1^k*, and returns all the global information* I *needed by users of the scheme, such as choice of groups or hash functions.*
- *A probabilistic sender key generation algorithm,* Key_S*. It takes as input the global information* I*, and outputs a private/public keypair* (sk_S, pk_S) *that is used to send signcrypted messages.*
- *A probabilistic receiver key generation algorithm,* Key_R*. It takes as input the global information* I*, and outputs a private/public keypair* (sk_R, pk_R) *that is used to receive signcrypted messages.*
- *A probabilistic signcryption algorithm* SC*. It takes as input the private key of the sender* sk_S*, the public key of the receiver* pk_R*, and a message* m*. It outputs a signcryptext* σ*.*
- *A deterministic unsigncryption algorithm* USC*. It takes as input the public key of the sender* pk_S*, the private key of the receiver* sk_R*, and a signcryptext* σ*. It outputs either a message* m *or the unique error symbol* $\perp$*.*

For a signcryption scheme to be sound, it is required that $m = USC\big(pk_S,\ sk_R,\ SC(sk_S, pk_R, m)\big)$ *for (almost) all fixed keypairs* (sk_S, pk_S) *and* (sk_R, pk_R)*.*

For a signcryption scheme to be useful, it is necessary that it also satisfies well-defined notions of security corresponding to the design goals of confidentiality and authenticity/integrity. Formally, the probability of an adversary breaking the security of signcryption should be *negligible* as a function of the security parameter 1^k.

Definition 2 (Negligible Function). *A function $f : \mathbb{N} \to \mathbb{R}$ is negligible if, for every polynomial p, there exists a $n_0 \in \mathbb{N}$ such that $|f(n)| \leq 1/|p(n)|$ for all $n \geq n_0$.*

Security models are commonly phrased in terms of games played between a hypothetical *challenger* and an *adversary*, who are both modelled as probabilistic Turing machines. The canonical notion of confidentiality for signcryption is that of indistinguishability of signcryptions (IND-CCA2). This is adapted directly from the corresponding security notion for encryption schemes: an adversary should not, even when given adaptive access to signcryption and unsigncryption oracles, be able to distinguish between the signcryption of two messages of his own choice. This security notion may be expressed by a game played between the challenger and a two-stage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. For a given security parameter 1^k, the game proceeds as follows:

1. The challenger generates a set of global parameters $I = Com(1^k)$, a sender keypair $(sk_S, pk_S) = Key_S(I)$ and a receiver keypair $(sk_R, pk_R) = Key_R(I)$.
2. The adversary runs $\mathcal{A}_1$ on the input (I, pk_S, pk_R). During its execution, $\mathcal{A}_1$ is given access to signcryption and unsigncryption oracles. The signcryption oracle takes a message m as input, and returns $SC(sk_S, pk_R, m)$. The unsigncryption oracle takes a signcryptext σ as input, and returns $USC(pk_S, sk_R, \sigma)$. $\mathcal{A}_1$ terminates by outputting two messages (m_0, m_1) of equal length, and some state information *state*.
3. The challenger computes a challenge signcryption by generating a random bit $b \in \{0, 1\}$ and computing $\sigma = SC(sk_S, pk_R, m_b)$.
4. The adversary runs $\mathcal{A}_2$ on the input $(state, \sigma)$. During its execution, $\mathcal{A}_2$ has access to signcryption and unsigncryption oracles as above, with the restriction that the challenge signcryptext σ may not be asked to the unsigncryption oracle. $\mathcal{A}_2$ terminates by outputting a guess b' for the value of b.

The adversary wins the game whenever $b = b'$. The advantage of $\mathcal{A}$ is defined as $\big|Pr[b = b'] - 1/2\big|$.

With regards to the authenticity and integrity of signcryption, the notion of existential forgery (UF-CMA) is adapted from analysis of signature schemes. It is however necessary to distinguish between different types of such forgery. In an *outsider-secure* signcryption scheme, the adversary is given access to signcryption and unsigncryption oracles, and the public keys of the sender and receiver. For the stronger notion of *insider security*, the unsigncryption oracle is replaced by giving the adversary direct access to the receiver's *private* key. This article will focus on insider-secure signcryption only. Efficient and secure hybrid signcryption scheme against outsider adversaries have been constructed by Dent [10]

It is also necessary to specify what it means for the adversary to win the security game. We use the notion of *strong* existential unforgeability (sUF-CMA). Here an adversary wins if it outputs a valid message/signcryption pair (m, σ) and the signcryption σ was not returned by the signcryption oracle when queried on the message m. Given a security parameter 1^k, a game for the sUF-CMA insider security of a signcryption scheme proceeds as follows:

1. The challenger generates a set of global parameters $I = Com(1^k)$, a sender keypair $(sk_S, pk_S) = Key_S(I)$ and a receiver keypair $(sk_R, pk_R) = Key_R(I)$.
2. The adversary $\mathcal{A}$ is run on the input (I, pk_S, sk_R, pk_R). During its execution, $\mathcal{A}$ is given access to a signcryption oracle, which takes a message m as input and returns $SC(sk_S, pk_R, m)$. $\mathcal{A}$ terminates by outputting a message m and a signcryptext σ.

The adversary wins the game if $m = USC(pk_S, sk_R, \sigma)$ and the signcryption oracle never returned σ when queried on the message m. The advantage of $\mathcal{A}$ is defined as $Pr[\mathcal{A} \text{ wins}]$.

2.2 Tag-KEMs

In the traditional KEM-DEM framework for hybrid encryption, the KEM uses public key methods to encrypt and transmit the symmetric key used by the DEM. Formally, a KEM consists of an asymmetric key generation algorithm that outputs a private/public keypair, an encapsulation algorithm that encrypts a random symmetric key using public-key techniques, and a decapsulation algorithm that uses the corresponding private key to decrypt said symmetric key from its encapsulation. This paradigm for building hybrid encryption schemes was extended in early 2005, when Abe *et. al.* [1] showed that one might build more efficient hybrid schemes by replacing the KEM with what they call a *tag-KEM.*

Definition 3 (Tag-KEM). *A tag-KEM* $TKEM = (Gen, Sym, Encap, Decap)$ *is defined as a tuple of four algorithms:*

- *A probabilistic key generation algorithm, Gen. It takes as input a security parameter* 1^k*, and outputs a private key sk and a public key pk. The public key contains all specific choices used by the scheme, such as choice of groups.*
- *A probabilistic symmetric key generation algorithm, Sym. It takes as input a public key pk, and outputs a symmetric key* K *and some internal state information* ω*.*
- *A probabilistic encapsulation algorithm, Encap. It takes as input the state information* ω *together with an arbitrary string* τ*, which is called a tag, and outputs an encapsulation* E*.*
- *A deterministic decapsulation algorithm, Decap. It takes a private key sk, an encapsulation* E *and a tag* τ *as input, and outputs a symmetric key* K*.*

For a tag-KEM to be sound, the decapsulation algorithm Decap must output the correct key K when run with a correctly formed encapsulation E of K, and the corresponding private key and tag.

Tag-KEMs as such may be viewed as a generalisation of regular KEMs: if the tag τ is a fixed string, the *Sym* and *Encap* algorithms together make up the encapsulation algorithm of the traditional model.

Definition 4 (DEM). *A data encapsulation mechanism $DEM = (Enc, Dec)$ is defined as a pair of algorithms:*

- *A symmetric encryption algorithm Enc, that takes a symmetric key $K \in \mathcal{K}$ and a message m as input, and returns a ciphertext $C = Enc_K(m)$. The set $\mathcal{K}$ is called the keyspace of the DEM.*
- *A symmetric decryption algorithm Dec, that takes a symmetric key $K \in \mathcal{K}$ and a ciphertext c as input, and returns a message $m = Dec_K(C)$.*

For soundness, the encryption and decryption algorithms should be each other's inverses under a fixed key K. Notationally, $m = Dec_K\big(Enc_K(m)\big)$.

For the purposes of this paper, it is only required that DEMs are secure with respect to indistinguishability against *passive* attackers (IND-PA). Formally, this security notion is captured by the following game, played between a challenger and a two-stage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$:

1. The challenger generates a random symmetric $K \in \mathcal{K}$.
2. The adversary runs $\mathcal{A}_1$ with the security parameter 1^k as input. $\mathcal{A}_1$ terminates by outputting two equal length messages m_0 and m_1, as well as some state information *state*.
3. The challenger generates a random bit $b \in \{0, 1\}$ and computes the challenge ciphertext $C = Enc_K(m_b)$.
4. The adversary runs $\mathcal{A}_2$ on the input $(state, C)$. $\mathcal{A}_2$ terminates by returning a guess b' for the value of b.

The adversary wins the game whenever $b = b'$. The advantage of $\mathcal{A}$ is defined as $\big|Pr[b = b'] - 1/2\big|$.

A tag-KEM may be combined with a DEM to form a hybrid encryption scheme in a similar way as a regular KEM. However, in [1] this is done in a novel manner, by using the ciphertext output by the DEM as the tag. The explicit construction is shown in Figure 1.

The main result of Abe *et. al.* [1] is that the construction of Figure 1 is IND-CCA2 secure, provided that the DEM is secure against passive attackers (IND-PA), and it is not possible for an adversary, given a pair (E, K), to determine whether K is the key encapsulated by E, or a random key of the correct length. This contrasts with the traditional KEM-DEM construction, in which the DEM is required to be secure against an active attack for the resulting hybrid encryption scheme to be IND-CCA2.

$\mathit{Encr}(pk, m)$: $(K, \omega) \overset{R}{\leftarrow} \mathit{Sym}(pk)$. $C \leftarrow \mathit{Enc}_K(m)$. $E \overset{R}{\leftarrow} \mathit{Encap}(\omega, C)$. $\sigma \leftarrow (E, C)$. Return σ.	$\mathit{Decr}(sk, \sigma)$: $(E, C) \leftarrow \sigma$. $K \leftarrow \mathit{Decap}(sk, E, C)$. $m \leftarrow \mathit{Dec}_K(C)$. Return m. $\mathit{Key}(1^k)$: $(sk, pk) \overset{R}{\leftarrow} \mathit{Key}(1^k)$. Return (sk, pk).

Fig. 1: Construction of asymmetric encryption scheme from a tag-KEM and DEM.

3 Signcryption Tag-KEMs

3.1 Basic Definition

We define Signcryption Tag-KEMs (SCTK) by direct analogy to the previous definition of tag-KEMs for encryption.

Definition 5 (Signcryption Tag-KEM). *A signcryption tag-KEM $SCTK = (Com, Key_S, Key_R, Sym, Encap, Decap)$ is defined as a tuple of six algorithms.*

- *A probabilistic common parameter generation algorithm, Com. It takes as input a security parameter 1^k, and returns all the global information I needed by users of the scheme, such as choice of groups or hash functions.*
- *A probabilistic sender key generation algorithm, Key_S. It takes as input the global information I, and outputs a private/public keypair (sk_S, pk_S) that is used to send signcrypted messages.*
- *A probabilistic receiver key generation algorithm, Key_R. It takes as input the global information I, and outputs a private/public keypair (sk_R, pk_R) that is used to receive signcrypted messages.*
- *A probabilistic symmetric key generation algorithm, Sym. It takes as input the private key of the sender sk_S and the public key of the receiver pk_R, and outputs a symmetric key K together with internal state information ω.*
- *A probabilistic key encapsulation algorithm, Encap. It takes as input the state information ω and an arbitrary tag τ, and returns an encapsulation E.*
- *A deterministic decapsulation/verification algorithm, Decap. It takes as input the sender's public key pk_S, the receiver's private key sk_R, an encapsulation E and a tag τ. The algorithm returns either a symmetric key K or the unique error symbol $\perp$.*

For the SCTK to be sound, the decapsulation/verification algorithm must return the correct key K whenever the encapsulation E is correctly formed and the corresponding keys and tag are supplied.

The basic idea behind a signcryption tag-KEM is that the key encapsulation algorithm provides what amounts to a signature on the tag τ. Signcryption tag-KEMs may thus be combined with regular DEMs to form a hybrid signcryption scheme as shown in Figure 2, using the SCTK to provide a signature on the symmetric ciphertext c and encapsulate the symmetric key K.

$Com(1^k)$: $I \xleftarrow{R} Com(1^k)$. Return I. $Keys_S(I)$: $(sk_S, pk_S) \xleftarrow{R} Keys_S(I)$. Return (sk_S, pk_S). $Key_R(I)$: $(sk_R, pk_R) \xleftarrow{R} Key_R(I)$. Return (sk_R, pk_R).	$SC(sk_S, pk_R, m)$: $(K, \omega) \xleftarrow{R} Sym(sk_S, pk_R)$. $C \leftarrow Enc_K(m)$. $E \xleftarrow{R} Encap(\omega, C)$. $\sigma \leftarrow (E, C)$. Return σ. $USC(pk_S, sk_R, \sigma)$: $(E, C) \leftarrow \sigma$. If $\perp \leftarrow Decap(pk_S, sk_R, E, C)$: Return $\perp$ and terminate. Else $K \leftarrow Decap(pk_S, sk_R, E, C)$. $m \leftarrow Dec_K(C)$. Return m.

Fig. 2: Construction of hybrid signcryption scheme from SCTK and DEM.

Previous discussion of hybrid signcryption schemes have discussed efficient hybrid signcryption as a variant of the “Encrypt-and-Sign” [2] paradigm. A straightforward approach is to encrypt the message to be sent with a symmetric cipher, while combining the features of key encapsulation and digital signatures into one efficient operation [8, 9, 4]. Using signcryption tag-KEMs in the construction yields something more akin to a “Encrypt-then-Sign” based scheme, since the signature is made on the ciphertext “tag”.

Another feature of the signcryption tag-KEM construction is that it automatically supports the sending of associated data with a message. In particular, one may submit a tag $\tau = (C, l)$ to the encapsulation algorithm, consisting of the ciphertext C as well as a label l containing any associated data that is to be bound to C by the encapsulation. As the encapsulation acts as a signature on the input tag, the authenticity and integrity of both ciphertext and associated data is provided. The only requirement for doing this is that the tag τ must be formatted in such a way that $(C, l) \leftarrow \tau$ may be parsed in a deterministic and unambiguous manner. A standard application of this feature is the common practice of “binding” the sender’s and receiver’s public key to any signcryption sent between them. Many signcryption schemes explicitly do this, in order to provide some degree of multi-user security. Of course, a similar effect can be achieved by computing the signcryption of a combination of the message and a hash of the associated data. This provides similar results but requires either slightly greater bandwidth or a slightly reduced message space.

3.2 Security Models

For a signcryption tag-KEM to be considered secure, it must fulfill well-defined security notions with respect to confidentiality and authenticity/integrity. The tag-KEM confidentiality model used in [1] may easily adapted to the signcryption setting, and the notion of strong existential unforgeability is adapted to provide authenticity/integrity.

In the IND-CCA2 game for a signcryption tag-KEM, the adversary attempts to distinguish whether a given symmetric key is the one embedded in an encapsulation. The adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ runs in three stages, with each stage having access to oracles that fascilitate both adaptive encapsulation and decapsulation queries. For a given security parameter 1^k, this may be expressed by the following game:

1. The challenger generates a set of global parameters $I = \mathit{Com}(1^k)$, a sender keypair $(sk_S, pk_S) = \mathit{Key}_S(I)$ and a receiver keypair $(sk_R, pk_R) = \mathit{Key}_R(I)$.
2. The adversary runs $\mathcal{A}_1$ on the input (I, pk_S, pk_R). Durings its execution, $\mathcal{A}_1$ is given access to three oracles, corresponding to each of the algorithms *Sym*, *Encap* and *Decap*:
   - The symmetric key generation oracle does not take any input, and computes $(K, \omega) = \mathit{Sym}(sk_S, pk_R)$. 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.
   - The key encapsulation oracle takes an arbitrary tag τ as input, 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 $\mathit{Encap}(\omega, \tau)$.
   - The decapsulation/verification oracle takes an encapsulation E and a tag τ as input, and returns $\mathit{Decap}(pk_S, sk_R, E, \tau)$.

   $\mathcal{A}_1$ terminates by returning state information state_1.
3. The challenger computes $(K_0, \omega) = \mathit{Sym}(sk_S, pk_R)$, and generates a random symmetric key $K_1 \in \mathcal{K}$ as well as a random bit $b \in \{0, 1\}$.
4. The adversary runs $\mathcal{A}_2$ on the input (state_1, K_b). During its execution, $\mathcal{A}_2$ may access the same oracles as previously. $\mathcal{A}_2$ terminates by returning state information state_2 and a tag τ.
5. The challenger computes a challenge encapsulation $E = \mathit{Encap}(\omega, \tau)$.
6. The adversary runs $\mathcal{A}_3$ on the input (state_2, E). During its execution, $\mathcal{A}_3$ may access the same oracles as previously, with the restriction that (E, τ) may not be asked to the decapsulation oracle. $\mathcal{A}_3$ terminates by returning a guess b' for the value of b.

The adversary wins the game whenever $b = b'$. The advantage of $\mathcal{A}$ is defined as $\left|Pr[b = b'] - 1/2\right|$. A signcryption tag-KEM is said to be *IND-CCA2 secure* if, for any adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in the IND-CCA2 game is negligible with respect to the security parameter 1^k.

It is important to notice the interaction between the symmetric key generation and encapsulation oracles. This is done to allow the adversary to perform completely adaptive encapsulations, without having access to the internal information stored in ω. The IND-CCA2 game ensures that a SCTK fulfills several necessary properties with regards to malleability and information hiding, and replaces the notions of IND-CCA2 and INP-CCA2 used by Dent [8, 9] for regular signcryption KEMs.

With respect to authenticity and integrity, an adversary should not be able to find encapsulation/tag-pairs (E, τ) such that $Decap(pk_S, sk_R, E, \tau) \neq \perp$, except by the way of oracles. Since the encapsulation algorithm should provide a signature on the tag τ, this is closely tied to forging the underlying signature scheme. An attack game corresponding to the sUF-CMA security of a SCTK may thus be specified as follows:

1. The challenger generates a set of global parameters $I = Com(1^k)$, a sender keypair $(sk_S, pk_S) = Key_S(I)$ and a receiver keypair $(sk_R, pk_R) = Key_R(I)$.
2. The adversary $\mathcal{A}$ is run on the input (I, pk_S, sk_R, pk_R). During its execution, $\mathcal{A}$ may access the symmetric key generation and encapsulation oracles as were defined in the previous game. $\mathcal{A}$ terminates by returning an encapsulation E and a tag τ.

The adversary wins the game if $\perp \neq Decap(pk_S, sk_R, E, \tau)$ and the encapsulation oracle never returned E when queried on the tag τ. The advantage of $\mathcal{A}$ is defined as $Pr[\mathcal{A} \text{ wins}]$. A signcryption tag-KEM is said to be *sUF-CMA secure* if, for any adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in the sUF-CMA game is negligible with respect to the security parameter 1^k.

Definition 6 (Secure Signcryption Tag-KEM). *A signcryption tag-KEM SCTK is said to be secure if it is IND-CCA2 and sUF-CMA secure.*

3.3 Generic Security of Hybrid Signcryption

If the SCTK+DEM construction is to be of any use, the resulting signcryption scheme must be provably secure.

Theorem 1. *Let SC be a hybrid signcryption scheme constructed from a signcryption tag-KEM and a DEM. If the signcryption tag-KEM is IND-CCA2 secure and the DEM is IND-PA secure, then SC is IND-CCA2 secure.*

Proof. Let Game 0 be the regular IND-CCA2 game for signcryption, as specified in Section 2.1. In the following game, the hybrid signcryption procedure is altered to use a random key when generating the challenge signcryptext, rather than the real key output by *Sym*. We refer to the resulting game as Game 1:

1. The challenger generates a set of global parameters $I = Com(1^k)$, a sender keypair $(sk_S, pk_S) = Key_S(I)$ and a receiver keypair $(sk_R, pk_R) = Key_R(I)$.

2. The adversary runs $\mathcal{A}_1$ on the input (I, pk_S, pk_R). During its execution, $\mathcal{A}_1$ has access to signcryption and unsigncryption oracles. The signcryption oracle takes a message m as input, and returns $SC(sk_S, pk_R, m)$. The unsigncryption oracle takes a signcryptext σ as input, and returns $USC(pk_S, sk_R, \sigma)$. $\mathcal{A}_1$ terminates by outputting two messages (m_0, m_1) and some state information *state*.
3. The challenger computes $(K, \omega) = Sym(sk_S, pk_R)$, and generates a random key $K' \in \mathcal{K}$, as well as a random bit $b \in \{0, 1\}$. He then computes $C = Enc_{K'}(m_b)$ and $E = Encap(\omega, C)$, and sets $\sigma = (E, C)$.
4. The adversary runs $\mathcal{A}_2$ on the input $(state, \sigma)$. During its execution, $\mathcal{A}_2$ may access signcryption and unsigncryption oracles as above, with the restriction that σ may not be asked to the unsigncryption oracle. $\mathcal{A}_2$ terminates by outputting a guess b' for the bit b.

Let X_0 and X_1 be the events that $b = b'$ in Game 0 and Game 1, respectively. It is well known that any substantial difference in the advantage of the adversary $\mathcal{A}$ in Game 0 and Game 1 can be used to produce a distinguishing algorithm for the signcryption tag-KEM.

Figure 3 gives a complete specification of such a distinguishing algorithm $\mathcal{D}$. It plays the IND-CCA2 game against SCTK, using $\mathcal{A}$ as a subroutine. Oracle queries made by $\mathcal{A}$ are simulated by $\mathcal{D}$. It uses the subroutines $\mathcal{O}_{SC}$ to simulate signcryption oracle queries, and $\mathcal{O}_{USC}$ to simulate unsigncryption queries. The symmetric key generation, encapsulation and decapsulation/verification oracles accessible by $\mathcal{D}$ are referred to as $\mathcal{O}_S$, $\mathcal{O}_E$ and $\mathcal{O}_D$, respectively. We denote the execution of an algorithm $\mathcal{A}$ that takes input values $\alpha, \ldots$ and has access to oracles $\mathcal{O}, \ldots$ as $\mathcal{A}(\alpha, \ldots; \mathcal{O}, \ldots)$. A well-known derivation gives $|Pr[X_0] - Pr[X_1]| \leq 2\epsilon_{SCTK}$, where ϵ_{SCTK} is the advantage that $\mathcal{D}$ has in attacking the IND-CCA2 security of the SCTK.

We proceed to show that the advantage of $\mathcal{A}$ in Game 1 is bounded by that of a passive attacker against the DEM. Figure 4 specifies an adversary $\mathcal{B}$ against the IND-PA security of the DEM, that uses $\mathcal{A}$ as a subroutine. In the game described in Figure 4, $\mathcal{B}$ simulates the environment of $\mathcal{A}$ in Game 1 perfectly. Furthermore, $\mathcal{B}$ wins every time $\mathcal{A}$ would have won Game 1. Hence, they have the same advantage. It follows that

$$\epsilon_{SC} \leq 2\epsilon_{SCTK} + \epsilon_{DEM}, \tag{1}$$

where ϵ_{SC}, ϵ_{SCTK} and ϵ_{DEM} are the advantages of adversaries against IND-CCA2 security of the hybrid signcryption scheme, the IND-CCA2 security of the signcryption tag-KEM and the IND-PA security of the DEM, respectively. □

Remark 1. This reduction is significantly tighter than those found for regular hybrid signcryption in [8, 4]. In the original approach to hybrid signcryption, the confidentiality proof relies on four terms: the indistinguishability of the symmetric keys the KEM produces, the unforgeability of the KEM, the ability of the KEM to disguise the messages and the passive security of the DEM. This

$\mathcal{D}_1(I, pk_S, pk_R; \mathcal{O}_S, \mathcal{O}_E, \mathcal{O}_D)$:
$(m_0, m_1, s) \stackrel{R}{\leftarrow} \mathcal{A}_1(I, pk_S, pk_R; \mathcal{O}_{SC}, \mathcal{O}_{USC})$.
$\mathit{state}_1 \leftarrow (m_0, m_1, s)$.
Return state_1.

$\mathcal{D}_2(\mathit{state}_1, K; \mathcal{O}_S, \mathcal{O}_E, \mathcal{O}_D)$:
$b \stackrel{R}{\leftarrow} \{0, 1\}$.
$C \leftarrow \mathit{Enc}_K(m_b)$.
$\mathit{state}_2 \leftarrow (\mathit{state}_1, b, C)$.
Return (state_2, C).

$\mathcal{D}_3(\mathit{state}_2, E; \mathcal{O}_S, \mathcal{O}_E, \mathcal{O}_D)$:
$(m_0, m_1, s, b, C) \leftarrow \mathit{state}_2$.
$\sigma \leftarrow (E, C)$.
$b' \stackrel{R}{\leftarrow} \mathcal{A}_2(s, \sigma; \mathcal{O}_{SC}, \mathcal{O}_{USC})$.
If $b = b'$: Return 1.
Else: Return 0.

$\mathcal{O}_{SC}(m)$:
$K \stackrel{R}{\leftarrow} \mathcal{O}_S$.
$C \leftarrow \mathit{Enc}_K(m)$.
$E \stackrel{R}{\leftarrow} \mathcal{O}_E(C)$.
$\sigma \leftarrow (E, C)$.
Return σ.

$\mathcal{O}_{USC}(\sigma)$:
$(E, C) \leftarrow \sigma$.
If $\perp \leftarrow \mathcal{O}_D(E, C)$:
Return $\perp$ and terminate.
Else $K \leftarrow \mathcal{O}_D(E, C)$.
$m \leftarrow \mathit{Dec}_K(C)$.
Return m.

Fig. 3: Distinguisher algorithm $\mathcal{D}$.

$\mathcal{B}_1$:
$I \stackrel{R}{\leftarrow} \mathit{Com}(1^k)$.
$(sk_S, pk_S) \stackrel{R}{\leftarrow} \mathit{Keys}_S(I)$.
$(sk_R, pk_R) \stackrel{R}{\leftarrow} \mathit{Key}_R(I)$.
$(m_0, m_1, s) \stackrel{R}{\leftarrow} \mathcal{A}_1(I, pk_S, pk_R; \mathcal{O}_{SC}, \mathcal{O}_{USC})$.
$\mathit{state} \leftarrow (I, sk_S, pk_S, sk_R, pk_R, m_0, m_1, s)$.
Return $(m_0, m_1, \mathit{state}')$.

$\mathcal{B}_2(\mathit{state}, C)$:
$(I, sk_S, pk_S, sk_R, pk_R, m_0, m_1, s) \leftarrow \mathit{state}$.
$(K, \omega) \stackrel{R}{\leftarrow} \mathit{Sym}(sk_S, pk_R)$.
$E \stackrel{R}{\leftarrow} \mathit{Encap}(\omega, C)$.
$\sigma \leftarrow (C, E)$.
$b \stackrel{R}{\leftarrow} \mathcal{A}_2(s, \sigma; \mathcal{O}_{SC}, \mathcal{O}_{USC})$.
Return b.

$\mathcal{O}_{SC}(m)$:
$(K, \omega) \stackrel{R}{\leftarrow} \mathit{Sym}(sk_S, pk_R)$.
$C \leftarrow \mathit{Enc}_K(m)$.
$E \stackrel{R}{\leftarrow} \mathit{Encap}(\omega, C)$.
$\sigma \leftarrow (E, C)$.
Return σ.

$\mathcal{O}_{USC}(\sigma)$:
$(E, C) \leftarrow \sigma$.
If $\perp \leftarrow \mathit{Decap}(pk_S, sk_R, E, C)$:
Return $\perp$ and terminate.
Else $K \leftarrow \mathit{Decap}(pk_S, sk_R, E, C)$.
$m \leftarrow \mathit{Dec}_K(C)$.
Return m.

Fig. 4: IND-PA adversary against the DEM.

is particularly inefficient as many proofs of unforgeability contain weak security reductions. We see this improved security result, and the comparative simplicity of proving the security of a signcryption tag-KEM, as the main advantages of the SCTK paradigm.

Theorem 2. *Let SC be a hybrid signcryption scheme constructed from a signcryption tag-KEM and a DEM. If the signcryption tag-KEM is sUF-CMA secure, then SC is also sUF-CMA secure.*

Proof. Since every valid forgery of SC implies a valid encapsulation, it is reasonably straightforward to show that forgery of SC implies forgery of the underlying SCTK. Figure 5 specifies an adversary $\mathcal{B}$, which uses a black-box adversary $\mathcal{A}$ against the UF-CMA security of SC to win the corresponding sUF-CMA game against SCTK. In the above scenario, $\mathcal{A}$ wins the forgery game against SC whenever the returned σ unsigncrypts to m and m has not been queried to the signcryption oracle $\mathcal{O}_{SC}$. If this is the case, then $\mathcal{B}$ wins the sUF-CMA game against SCTK.

$\mathcal{B}(I, pk_S, sk_R, pk_R; \mathcal{O}_S, \mathcal{O}_E)$:	$\mathcal{O}_{SC}(m)$:
$(m, \sigma) \overset{R}{\leftarrow} \mathcal{A}(I, pk_S, sk_R, pk_R; \mathcal{O}_{SC})$.	$K \overset{R}{\leftarrow} \mathcal{O}_S$.
$(E, C) \leftarrow \sigma$.	$C \leftarrow Enc_K(m)$.
Return (E, C).	$E \overset{R}{\leftarrow} \mathcal{O}_E(C)$.
	$\sigma \leftarrow (E, C)$.
	Return σ.

Fig. 5: Construction of a sUF-CMA adversary against SCTK.

To see this, note that $\mathcal{B}$ wins whenever it returns a pair (E, C) that does not decapsulate to $\perp$ and such that E was never a response from $\mathcal{O}_E$ to a query C. Since σ is a valid ciphertext, the former condition is always fulfilled. Furthermore, one may note that the ciphertext σ is associated deterministically to m through the decapsulation algorithm. Hence, σ has been returned by $\mathcal{O}_{SC}$ if and only if m was ever queried. This implies that (E, C) was a query/response pair from $\mathcal{O}_{SC}$ if and only if (m, σ) was a query/response pair from $\mathcal{O}_E$. Hence, $\mathcal{B}$ wins every time $\mathcal{A}$ does.

It follows that

$$\epsilon_{SC} \leq \epsilon_{SCTK}, \tag{2}$$

where ϵ_{SC} is the advantage of the UF-CMA adversary against SC, and ϵ_{SCTK} is the advantage of the resulting sUF-CMA adversary against SCTK. □

4 Sample schemes

4.1 Zheng Signcryption Revisited

Zheng's original signcryption scheme [13] has become somewhat of a canonical reference when hybrid signcryption is discussed [8, 4]. It is therefore natural to see whether it can be adapted to fit the generic tag-KEM framework as well. Since Zheng's original scheme essentially uses a KEM to sign the plaintext message, this requires only minor alterations. Figure 6 gives a complete specification of a signcryption tag-KEM that, when combined with a DEM as per Figure 2, yields something very similar to Zheng's original scheme. The only difference between the schemes is that the tag τ used by *Encap* is the ciphertext $C \leftarrow Enc_K(m)$, rather than m itself. It is well established that both Zheng's signcryption scheme and its associated signcryption KEM are secure [3, 8, 4], and it is therefore no surprise that the signcryption tag-KEM specified in Figure 6 is secure as well.

$Com(1^k)$:
Pick a k-bit prime p.
Pick a large prime q that divides $p-1$.
Pick $g \in \mathbb{Z}_q^*$ of order q.
Pick cryptographic hash functions:
$\mathcal{G} : \{0,1\}^* \to \mathcal{K}$.
$\mathcal{H} : \{0,1\}^* \to \mathbb{Z}/q\mathbb{Z}$.
$I \leftarrow (p, q, g, \mathcal{G}, \mathcal{H})$.
Return I.

$Key_S(I)$:
$sk_S \xleftarrow{R} \mathbb{Z}/q\mathbb{Z}$.
$pk_S \leftarrow g^{sk_S} \mod p$.
Return (sk_S, pk_S).

$Key_R(I)$:
$sk_R \xleftarrow{R} \mathbb{Z}/q\mathbb{Z}$.
$pk_R \leftarrow g^{sk_R} \mod p$.
Return (sk_R, pk_R).

$Sym(sk_S, pk_R)$:
$n \xleftarrow{R} \mathbb{Z}/q\mathbb{Z}$.
$\kappa \leftarrow {pk_R}^n \mod p$.
$bind \leftarrow pk_S || pk_R$.
$K \leftarrow \mathcal{G}(\kappa)$.
$\omega \leftarrow (sk_S, n, \kappa, bind)$.
Return (K, ω).

$Encap(\omega, \tau)$:
$(sk_S, n, \kappa, bind) \leftarrow \omega$.
$r \leftarrow \mathcal{H}(\tau || bind || \kappa)$.
$s \leftarrow n/(sk_S + r) \mod q$.
$E \leftarrow (r, s)$.
Return E.

$Decap(pk_S, sk_R, E, \tau)$:
$(r, s) \leftarrow E$.
$\kappa \leftarrow (pk_S \cdot g^r)^{s \cdot sk_R} \mod p$.
$r' \leftarrow \mathcal{H}(\tau || bind || \kappa)$.
If $r \neq r'$:
Return $\perp$ and terminate.
Else $K \leftarrow \mathcal{G}(\kappa)$.
Return K.

Fig. 6: The Zheng signcryption tag-KEM.

Theorem 3. *Zheng-SCTK, as specified in Figure 6, is a secure signcryption tag-KEM.*

A full version of the proof is given in the full version of the paper. The security bounds for Zheng's signcryption scheme in this framework are comparable to those of the original scheme-specific reduction [3]. This was not the case in generic models for hybrid signcryption [8, 4] based on regular KEMs. In the full version of the paper, we show that an attacker who attempts to break the confidentiality of the full signcryption scheme using at most q_E queries to the signcryption oracle, q_D queries to the unsigncryption oracle, q_G queries to the random oracle representing the hash function $\mathcal{G}$ and q_H queries to the random oracle representing the hash function $\mathcal{H}$ has an advantage bounded[3] by

$$2Adv_{GDH} + Adv_{DEM}$$

where Adv_{GDH} is a related attacker's probability of solving a Gap Diffie-Hellman problem and Adv_{DEM} is the advantage that a related attacker has in breaking the passive security of the DEM. If we compare this to the results of Bjørstad [4], then we find that an attacker who attempts to break the confidentiality of Zheng's scheme in Dent's hybrid model [9] has an advantage which is bounded above by

$$4Adv_{GDH} + Adv_{DEM} + 2q_H\sqrt{Adv_{DL}}$$

where Adv_{DL} is a related attacker's probability of solving a discrete logarithm problem. This demonstrates the usefulness of the new construction, as it gives significantly tighter security bounds.

Other existing signcryption schemes may also be representable as signcryption tag-KEMs. For example, it appears likely that the hybrid signcryption scheme of Malone-Lee [11] could also be adapted to the signcryption tag-KEM paradigm, along with its corresponding proof of security.

4.2 The CM Signcryption Tag-KEM

As discussed in [13, 4], the Zheng signcryption scheme is constructed by modifying an existing signature scheme. By making the randomiser κ computed during signature verification dependent on the receiver's key sk_S, an efficient signcryption scheme can be constructed at a very low additional cost. This trick may be applied to other signature schemes as well. In this section, we propose a new signcryption tag-KEM, built from a recent signature scheme due to Chevallier-Mames [5]. The resulting construction has tight security reductions with respect to the Computational Diffie-Hellman and Gap Diffie-Hellman problems. This is of practical interest, since previous hybrid signcryption schemes have had relatively loose security reductions with respect to unforgeability. Figure 7 gives a complete specification of the CM signcryption tag-KEM.

Theorem 4. *The CM signcryption tag-KEM specified in Figure 7 is a secure signcryption tag-KEM.*

A full proof is given in full version of the paper. The proof uses techniques that are directly analogous to those used in the security proofs for Zheng's scheme [3, 4]. However, this scheme has a better security reduction for authenticity/integrity, since the security of the underlying signature scheme does not rely on a "forking lemma" argument [12]. To the authors' knowledge, this gives this scheme the best known security reductions.

As a side note, we remark that, in order to prove the integrity/authenticity of the CM signcryption tag-KEM, it was necessary to prove that the Chevallier-Mames signature scheme was strongly unforgeable. A proof of this fact was developed independently by Chevallier-Mames [6].

[3] For simplicity, we disregard the constant terms in the following expressions

$Com(1^k)$: Pick a large prime q. Let G be a cyclic group of order q, such that the representation of the elements of G is included in $\{0,1\}^k$. Pick a generator g of G. Pick cryptographic hash functions: $\mathcal{G} : \{0,1\}^* \times G^6 \to \mathbb{Z}_q$. $\mathcal{H} : G \to G$. $KDF : G \to \mathcal{K}$. $I \leftarrow (q, G, g, \mathcal{G}, \mathcal{H}, KDF)$. Return I. $Key_S(I)$: $sk_S \stackrel{R}{\leftarrow} \mathbb{Z}_q$. $pk_S \leftarrow g^{sk_S}$. Return (sk_S, pk_S). $Key_R(I)$: $sk_R \stackrel{R}{\leftarrow} \mathbb{Z}_q$. $pk_R \leftarrow g^{sk_R}$. Return (sk_R, pk_R).	$Sym(sk_S, pk_R)$: $n \stackrel{R}{\leftarrow} \mathbb{Z}_q$. $u \leftarrow {pk_R}^n$. $K \leftarrow KDF(u)$. $\omega \leftarrow (sk_S, pk_R, n, u)$. Return (K, ω). $Encap(\omega, \tau)$: $(sk_S, pk_R, n, u) \leftarrow \omega$. $h \leftarrow \mathcal{H}(u)$. $z \leftarrow h^{sk_S}$. $v \leftarrow h^n$. $c \leftarrow \mathcal{G}(\tau \| pk_R, pk_S, g, z, h, u, v)$. $s \leftarrow n + c \cdot sk_S, \mod q$. $E \leftarrow (z, c, s)$. $Decap(pk_S, sk_R, E, \tau)$: $u \leftarrow (g^s \cdot {pk_S}^{-c})^{sk_R}$. $h \leftarrow \mathcal{H}(u)$. $v \leftarrow h^s \cdot z^{-c}$. If $c \neq \mathcal{G}(\tau \| pk_R, pk_S, g, z, h, u, v)$: Return $\perp$. Else $K \leftarrow KDF(u)$. Return K.

Fig. 7: The CM signcryption tag-KEM

5 Building Better Key Agreement Mechanisms with Signcryption Tag-KEMs

The idea that signcryption KEMs can be used as key agreement mechanisms was first investigated by Dent [10]. Dent notes that whilst an encryption KEM provides a basic mechanism for agreeing a symmetric key between two parties, it does not provide any form of authentication or freshness guarantee. Moreover, he notes that signcryption KEMs (with outsider security) can be used to agree a symmetric key with authentication. A simple protocol key agreement protocol is then proposed, wherein freshness is guaranteed by the computing the MAC of a timestamp or nonce using the newly agreed symmetric key. However, as the paper remarks, this protocol is susceptible to a known key attack and should not be used in practice.

In this section we propose that signcryption tag-KEMs can be used as practical key agreement mechanisms, with the SCTK providing both the authentication and freshness components of the protocol in a simple way. Consider the following protocol which allows Alice and Bob to agree a key for a session with an ID *SID* between them:

1. Alice generates a random nonce r_A of an agreed length, and sends r_A to Bob.

2. Bob computes $(K, \omega) = Sym(sk_{Bob}, pk_{Alice})$ and $E = Encap(\omega, \tau)$ using the (unique) tag $\tau = r_A||SID$. Bob accepts K as the shared secret key, and sends C to Alice.
3. Alice computes $K = Decap(pk_{Bob}, sk_{Alice}, E, \tau)$ using the tag $\tau = r_A||SID$, and accepts K as the shared key providing $K \neq \perp$.

We argue that this protocol has the following attributes:

– **Implicit key authentication to both parties**. If both parties obtain the other's correct public key, then no attacker can distinguish between a session's correct public key and a randomly generated key without breaking the confidentiality criterion for the SCTK.
– **Resistance to known key attacks**. It is easy to see that an attacker that gains a key from any earlier protocol execution (or, indeed, in a later protocol execution) between Alice and Bob gains no advantage in breaking the scheme. This is because this "session corruption" is equivalent to making a signcryption oracle query with a random tag. Since the SCTK remains secure in this situation, so does the key agreement protocol.
– **Key confirmation from Bob to Alice**. Since no party (including Alice) can forge a signcryptext that purports to come from Bob, if Alice recovers a key K from C, then that key K *must* have been produced by Bob in the correct way. Therefore, Alice can have confidence that Bob knows the correct key. However, an extra round of interaction will be required if Alice wishes to give Bob key confirmation.

We argue that this derivation is useful because it finally gives a secure way to use KEMs for key establishment. Of course, a secure signcryption scheme can always be used as a key transport mechanism; however, it was not previously known if signcryption-style techniques could be used for key agreement. The aforementioned protocol settles this question. Whether an individual signcryption tag-KEM should be regarded as a key transport or key agreement mechanism depends upon its individual characteristics.

6 Conclusions

We have shown that there is a natural extension of the concept of a tag-KEM to the signcryption setting and proven that secure signcryption tag-KEMs can be combined with passively secure DEMs to provide signcryption schemes with full insider security. This vastly simplifies and improves upon the KEM-DEM model insider secure signcryption schemes proposed by Dent [9]. To show that this construction is viable, we have given several examples of signcryption tag-KEMs, including a brand new construction based on the Chevallier-Mames signature scheme with very tight security bounds.

Acknowledgements

Tor Bjørstad wishes to thank the ECRYPT project and the Norwegian Research Council for their generous financial support. Alexander Dent wishes to think the

ECRYPT project and the EPSRC's Junior Research Fellowship programme for their generous financial support. Both authors wish to thank the PKC 2006 anonymous reviewers for their comments.

References

1. A. Abe, R. Gennaro, K. Kurosawa, and V. Shoup. Tag-KEM/DEM: A new framework for hybrid encryption and a new analysis of Kurosawa-Desmedt KEM. In *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 128–146. Springer–Verlag, 2005.
2. J. H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 83–107. Springer–Verlag, 2002.
3. J. Baek, R. Steinfeld, and Y. Zheng. Formal proofs for the security of signcryption. In *Proceedings of PKC 2002*, volume 2274 of *Lecture Notes in Computer Science*, pages 80–98. Springer–Verlag, 2002.
4. T. E. Bjørstad. Provable security of signcryption. Master's thesis, Norwegian University of Technology and Science, 2005. http://www.ii.uib.no/~tor/pdf/msc_thesis.pdf.
5. B. Chevallier-Mames. An efficient CDH-based signature scheme with a tight security reduction. In *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 511–526. Springer–Verlag, 2005.
6. B. Chevallier-Mames. Personal correspondence, 2005.
7. R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2004.
8. A. W. Dent. Hybrid cryptography. Cryptology ePrint Archive, Report 2004/210, 2004. http://eprint.iacr.org/2004/210/.
9. A. W. Dent. Hybrid signcryption schemes with insider security. In *Proceedings of ACISP 2005*, volume 3574 of *Lecture Notes in Computer Science*, pages 253–266. Springer–Verlag, 2005.
10. A. W. Dent. Hybrid signcryption schemes with outsider security. In *Proceedings of ISC 2005*, volume 3650 of *Lecture Notes in Computer Science*, pages 203–217. Springer–Verlag, 2005.
11. J. Malone-Lee. Signcryption with non-interactive non-repudiation. Technical Report CSTR-02-004, Department of Computer Science, University of Bristol, 2004. http://www.cs.bris.ac.uk/Publications/Papers/1000628.pdf.
12. D. Pointcheval and J. Stern. Security proofs for signature schemes. In *Advances in Cryptology - EUROCRYPT '96*, volume 1070, pages 387–398. Springer–Verlag, 1996.
13. Y. Zheng. Digital signcryption or how to achieve cost (signature & encryption) $<<$ cost (signature) + cost (encryption). In *Advances in Cryptology – CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 165–179. Springer–Verlag, 1997. Unpublished full version (47 pages), dated 1999, available through the author's home page http://www.sis.uncc.edu/~yzheng/papers/signcrypt.pdf.