Aggregate Signcryption

Alexander W. Dent

Qualcomm Research

Abstract. Signcryption schemes provide an efficient messaging system for data that needs to be sent with data confidentiality, data integrity and data origin authentication. However, the bandwidth overhead for the use of signcryption in a network in which a large number of messages need to be sent may be high. Motivated by aggregate signature schemes, we propose the concept of an aggregate signcryption scheme. An aggregate signcryption scheme allows distinct signcryption ciphertexts intended for the same recipient to be merged into a single signcryption ciphertext of smaller size *without losing any of their security guarantees*. This has the potential to provide significant bandwidth savings. We propose security models for this scheme, analyse the trivial generic constructions, propose an efficient new scheme, and analyse the bandwidth requirements of these schemes for a practical distributed database application.

1 Introduction

One of the most fundamental challenges for secure communication is end-to-end security for message transmission. This problem has been solved in many contexts using different technologies. In this paper we concentrate on the use of public-key cryptography to give data confidentiality, data integrity and origin authentication for message transmission. The original version of the problem was solved with the invention of public-key encryption and digital signature schemes [15]. Efficient combinations of public-key encryption and digital signatures are known as signcryption schemes [30] and can provide computationally efficient secure message transmissions.

This paper explores the concept of aggregate signcryption – a topic first introduced in the identity-based setting [28] and later explored in the certificateless setting [24]. This discussion will focus on the traditional public-key setting. Aggregate signcryption schemes take the conceptual ideas of aggregate signature schemes and apply them to signcryption schemes. An aggregate signcryption scheme allows individual signcryption ciphertexts intended for the same recipient to be aggregated into a single (shorter) combined ciphertext without losing any of the security guarantees that would be present if the original signcryption ciphertexts were transmitted individually.

In other words, the aggregate ciphertext still provides a mechanism for message transmission with data confidentiality, data integrity and origin authentication, but with significantly reduced overhead. We stress that the aggregation algorithm is entirely public – it can be performed by any entity given a number of signcryption ciphertexts and the corresponding public keys.

This allows for signcryption ciphertexts intended for a single receiver to be combined within the network. If a network node receives two signcryption ciphertext which need to be routed to the same receiver, then the node can run the aggregation algorithm and forward the aggregate signcryption ciphertext at a reduced bandwidth cost.

In this paper, we explore the concept of aggregate signcryption in the public-key setting (including a model for a new type of attack which we term a denial of decryption attack). We analyse the obvious mechanisms for constructing an aggregate signcryption scheme through the combination of a public-key encryption scheme and an aggregate signature scheme (Section 4). We show that two of the four classic combination mechanisms, Encrypt-then-Sign ($\mathcal{E}t\mathcal{S}$) and Commit-then-Encrypt-and-Sign ($\mathcal{C}t\mathcal{E}\&\mathcal{S}$), lead to secure aggregate signcryption schemes. Next we develop a specific example of an aggregate signcryption scheme which has high bandwidth efficiency (Section 5) through a novel non-trivial optimisation of an encrypt-then-sign construction using the Boneh-Franklin identity-based encryption scheme [10] and the BGLS aggregate signature scheme [11]. The proof relies on the random oracle methodology.

Lastly, in Section 6, we analyse an example of the aggregate signcryption scheme in practice. We consider implementing the direct construction of Section 5 in the Gaian database system [6]. The Gaian database is a dynamic, distributed, federated database system in which data is held in nodes in an evolving database "graph". The database is federated in the sense that a node which wishes to process a query on data that is not held locally will flood the database graph with queries for that data and collate the responses. The database is designed for use in scenarios which include military applications and so there is a strong requirement for secure message transmission in mobile networks with limited bandwidth. This makes aggregate signcryption an excellent candidate for securing intra-node communication in the Gaian database (particularly for the query response messages which every node has to produce and which are all sent to a single recipient). Our results show that the new scheme provides significant bandwidth savings compared to existing solutions (more than 13% when compared to the best existing comparable solution).

2 Preliminaries

2.1 Notation

Let $\leftarrow$ denote assignment and $\xleftarrow{\$}$ denote random assignment. Thus, if S a finite set, then $y \xleftarrow{\$} S$ denotes the assignment to y of a uniform random element of S. We assume that any vector $\boldsymbol{x}$ is an ordered set of elements $(x^{(1)}, \dots, x^{(n)})$ for some n. If $\mathcal{A}$ is a deterministic algorithm than $y \leftarrow \mathcal{A}(x)$ denotes the output of $\mathcal{A}$ given input x. If $\mathcal{A}$ is a probabilistic algorithm then $y \xleftarrow{\$} \mathcal{A}(x)$ denotes the output of $\mathcal{A}$ given input x and fresh random coins. Some algorithms may output an "invalid" symbol $\perp$. If we want to assess whether the output $\mathcal{A}(x)$ has some property, then we will assume that $\perp$ never has that property unless explicitly stated.

2.2 Aggregate Signatures

Aggregate signcryption schemes can be considered to be motivated by aggregate signature schemes; hence, it is useful to consider the purpose and security of aggregate signature schemes. An aggregate signature scheme [11] is a tuple of PPT algorithms ($\texttt{Gen}, \texttt{Sign}, \texttt{Aggregate}, \texttt{Ver}$).

- The key-generation algorithm generates a public/private key pair $(pk, sk) \xleftarrow{\$} \texttt{Gen}(1^k)$.
- The signing algorithm takes as input a message m and the private key sk, and outputs a signature $\sigma \xleftarrow{\$} \texttt{Sign}(sk, m)$.
- The aggregation algorithm $\texttt{Aggregate}$ takes as input a vector of public keys $\boldsymbol{pk}$, a vector of messages $\boldsymbol{m}$ and a vector of signatures $\boldsymbol{\sigma}$, and outputs an aggregate signature $\sigma \leftarrow \texttt{Aggregate}(\boldsymbol{pk}, \boldsymbol{m}, \boldsymbol{\sigma})$.
- The verification algorithm $\texttt{Ver}$ takes as input a vector of public keys $\boldsymbol{pk}$, a vector of messages $\boldsymbol{m}$ and an aggregate signature σ, and outputs either the valid symbol $\top$ or an invalid symbol $\perp$.

We require that if $\sigma^{(i)} \xleftarrow{\$} \texttt{Sign}(sk^{(i)}, m^{(i)})$ for $1 \leq i \leq n$ and $\sigma \xleftarrow{\$} \texttt{Aggregate}(\boldsymbol{pk}, \boldsymbol{\sigma})$, then $\texttt{Ver}(\boldsymbol{pk}, \boldsymbol{m}, \sigma) = \top$. We also assume that if $\sigma \xleftarrow{\$} \texttt{Sign}(sk, m)$ then $\texttt{Aggregate}(pk, \sigma) = \sigma$ and so the verification algorithm works correctly on individual signatures as well as aggregate signatures.

The security notion for an aggregate signature scheme was given by Boneh *et al.* [11] and involves the game shown in Figure 1. An attacker $\mathcal{A}$ has advantage $\Pr[\text{Expt}^{\texttt{UF}}_{\mathcal{A}}(k) = 1]$. An aggregate signature scheme is UF-CMA secure if every PPT attacker $\mathcal{A}$ has negligible advantage.

Aggregate signature schemes are designed to improve the bandwidth/verification efficiency for digital signatures that have to be verified by the same entity. For example, in the case where an entity wishes to verify a signature using a public key that is supplied with a digital certificate. The verifying entity has to verify the original signature and the signature on the certificate. If both signatures are computed using the same aggregate signature scheme, then the two signatures can be compressed into one aggregate signature which can be verified at the same time. However, aggregate signature schemes only provide data integrity and data origin authentication, we aim to provide a similar functionality but providing data confidentiality too.

```
$\textsc{Expt}^{\texttt{UF}}_{\mathcal{A}}(k)$:
  $(pk, sk) \xleftarrow{\$} \texttt{Gen}(1^k)$
  $(\boldsymbol{pk}, \boldsymbol{m}, \sigma) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_S}(pk)$
  Output 1 if
    (a) $\texttt{Ver}(\boldsymbol{pk}, \boldsymbol{m}, \sigma) = \top$
    (b) $\exists\ i$ such that $pk^{(i)} = pk$ and
      $m^{(i)}$ was not queried to $\mathcal{O}_S(m^{(i)})$
  Else output 0

$\mathcal{O}_S(m)$:
  Return $\sigma \xleftarrow{\$} \texttt{Sign}(sk, m)$
```

```
$\textsc{Expt}^{\texttt{IND-b}}_{\mathcal{A}}(k)$:
  $(pk, sk) \xleftarrow{\$} \texttt{Gen}(1^k)$
  $(m_0, m_1, \omega) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_D}(pk)$
  $C^* \xleftarrow{\$} \texttt{Enc}(pk, m_b)$
  If $|m_0| \neq |m_1|$ then $C^* \leftarrow \perp$
  $b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_D}(C^*, \omega)$
  Output $b'$

$\mathcal{O}_D(C)$:
  Return $m \leftarrow \texttt{Dec}(sk, C)$
```

Fig. 1. The UF-CMA security model for aggregate signatures (LHS) and the IND-CCA2 security model for public-key encryption (RHS). In the IND-CCA2 model, $\mathcal{A}_2$ is not allowed to submit a $\mathcal{O}_D(C^*)$ query.

2.3 Public-Key Encryption

Since signcryption is often compared to public-key encryption, it is also useful to introduce that notion here too. A public-key encryption scheme is a triple of PPT algorithms $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$.

- The key-generation algorithm generates a public/private key pair $(pk, sk) \xleftarrow{\$} \texttt{Gen}(1^k)$. The public key defines a (polynomial-time recognisable) message space $\mathcal{M}$ and a (polynomial-time recognisable) ciphertext space $\mathcal{C}$.
- The encryption algorithm takes a public key pk and a message $m \in \mathcal{M}$ as input, and outputs a ciphertext $C \xleftarrow{\$} \texttt{Enc}(pk, m)$.
- The decryption algorithm takes a private key sk and a ciphertext $C \in \mathcal{C}$, and outputs a either a message $m \leftarrow \texttt{Dec}(sk, C)$ or an invalid symbol $\perp$.

We require that if $C \xleftarrow{\$} \texttt{Enc}(pk, m)$ then $\texttt{Dec}(sk, C) = m$.

The security model for public-key encryption was given in a series of papers [18, 25, 26]. It is defined by the game shown in Figure 1. An attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ has advantage

$$Adv^{\texttt{IND}}_{\mathcal{A}}(k) = |\Pr[\textsc{Expt}^{\texttt{IND-1}}_{\mathcal{A}}(k) = 1] - \Pr[\textsc{Expt}^{\texttt{IND-0}}_{\mathcal{A}}(k) = 1]|\,.$$

An encryption scheme is IND-CCA2 secure if every PPT attacker $\mathcal{A}$ has negligible advantage (where $\mathcal{A}_2$ may not make a $\mathcal{O}_D(C^*)$ query). We may also define a weaker notion of security known as IND-RCCA security [13]. It is identical to the IND-CCA2 except if $\mathcal{A}_2$ submits an $\mathcal{O}_D(C)$ query with $\texttt{Dec}(sk, C) \in \{m_0, m_1\}$ then the decryption oracle returns `test`. This notion of security has been shown to be sufficient in situations where an attacker's ability to replay messages is not considered a breach of security.

2.4 Symmetric Encryption

A symmetric encryption is a pair of deterministic polynomial-time algorithms $(\textsc{Enc}, \textsc{Dec})$. The encryption algorithm takes as input a key K of length k and a message $m \in \{0,1\}^*$, and outputs a ciphertext $C \leftarrow \textsc{Enc}_K(m)$. The decryption algorithm takes as input a key K of length k and a ciphertext $C \in \{0,1\}^*$, and outputs either a message $m \leftarrow \textsc{Dec}_K(C)$ or the error symbol $\perp$. We require that for all $K \in \{0,1\}^k$ and $m \in \{0,1\}^*$, we have that $\texttt{Dec}_K(\texttt{Enc}_K(m)) = m$.

The security model for symmetric encryption was developed in a series of papers [18, 4]. It is defined by the game:

```
$\textsc{Expt}_{\mathcal{A}}^{\texttt{sym-b}}(k)$:
  $K \xleftarrow{\$} \{0,1\}^k$
  $(m_0, m_1, \omega) \xleftarrow{\$} \mathcal{A}_1(1^k)$
  $C^* \xleftarrow{\$} \texttt{Enc}_K(m_b)$
  If $|m_0| \neq |m_1|$ then $C^* \leftarrow \perp$
  $b' \xleftarrow{\$} \mathcal{A}_2(C^*, \omega)$
  Output $b'$
```

An attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ has advantage

$$Adv_{\mathcal{A}}^{\texttt{sym}}(k) = |\Pr[\textsc{Expt}_{\mathcal{A}}^{\texttt{sym-1}}(k) = 1] - \Pr[\textsc{Expt}_{\mathcal{A}}^{\texttt{sym-0}}(k) = 0]|$$

and the symmetric encryption scheme is said to be IND-CPA secure if every PPT attacker $\mathcal{A}$ has negligible advantage.

2.5 Commitment Schemes

A commitment scheme is a triple of PPT algorithms ($\texttt{Setup}, \texttt{Commit}, \texttt{Open}$):

- The setup algorithm outputs public parameters $param \xleftarrow{\$} \texttt{Setup}(1^k)$.
- The commit algorithm takes as input the public parameters $param$ and a message m, and outputs a commitment/decommitment pair $(c, d) \xleftarrow{\$} \texttt{Commit}(param, m)$.
- The open algorithm takes as input the public parameters $param$, the commitment c, and the decommitment d; it outputs a message $m \leftarrow \texttt{Open}(param, c, d)$.

We require that if $(c, d) \xleftarrow{\$} \texttt{Commit}(param, m)$ then $m \leftarrow \texttt{Open}(param, c, d)$.

We require two security properties from a commitment scheme [2]: hiding and relaxed binding. For the hiding game, we consider an attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in the game in the LHS of Figure .The attacker's advantage is defined to be

$$Adv_{\mathcal{A}}^{\texttt{hide}}(k) = |\Pr[\textsc{Expt}_{\mathcal{A}}^{\texttt{hid-1}}(k) = 1] - \Pr[\textsc{Expt}_{\mathcal{A}}^{\texttt{hid-0}}(k) = 1]|$$

and a scheme is said to be hiding if every PPT attacker has negligible advantage.

```
$\textsc{Expt}_{\mathcal{A}}^{\texttt{hid-b}}(k)$:
  $param \xleftarrow{\$} \texttt{Setup}(1^k)$
  $(m_0, m_1, \omega) \xleftarrow{\$} \mathcal{A}_1(param)$
  $(c, d) \xleftarrow{\$} \texttt{Commit}(param, m_b)$
  If $|m_0| \neq |m_1|$ then $c \leftarrow \perp$
  $b' \xleftarrow{\$} \mathcal{A}_2(c, \omega)$
  Output $b'$
```

```
$\textsc{Expt}_{\mathcal{A}}^{\texttt{bind}}(k)$:
  $param \xleftarrow{\$} \texttt{Setup}(1^k)$
  $(m, \omega) \xleftarrow{\$} \mathcal{A}_1(param)$
  $(c, d) \xleftarrow{\$} \texttt{Commit}(param, m)$
  $d' \xleftarrow{\$} \mathcal{A}_2(c, d, \omega)$
  $m' \leftarrow \texttt{Open}(param, c, d')$
  Output 1 if $m' \neq m$
  Else output 0
```

Fig. 2. The security models for hiding (LHS) and binding (RHS) for commitment schemes.

For the relaxed binding game, we consider an attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ playing the game on the RHS of Figure 2. The attacker's advantage is defined to be $\Pr[\textsc{Expt}_{\mathcal{A}}^{\texttt{bind}} = 1]$ and a scheme is said to be relaxed binding if every PPT attacker has negligible advantage.

For technical reasons, we require that there exists at most one commitment c corresponding to a decommitment d and that all messages m of the same length produce decommitments d of the same length. This latter condition is used in proving the security of the $\mathcal{C}t\mathcal{E}\&\mathcal{S}$ construction (see Section A) but appears to be missing from the original analysis of this construction [2]. Both of these conditions are trivially satisfied by all reasonable commitment schemes.

3 Aggregate Signcryption

An aggregate signcryption scheme is designed to have the same security properties as a signcryption scheme (i.e. confidentiality and integrity protection) but it allows for multiple signcryption ciphertexts to be combined into a single signcryption ciphertext which communicates the same information and has the same security properties, but consumes less bandwidth. Our security notions combine those of signcryption [2, 3] with those of aggregate signatures [7, 11]. For technical reasons, we will assume that the set of valid public keys is recognisable[1] and every valid public key has exactly one corresponding private key.

Our security models are very similar to those of [28] except with the obvious changes made for the models to be applied in the public-key setting. One interesting pair of differences is that the confidentiality model of Sharmilla *et al.* insists that the challenge ciphertext is an aggregate and only forbids the attacker from submitting the challenge ciphertext to the decryption oracle; thus implicitly prevent the scheme from allowing aggregating two aggregate ciphertexts. Technically speaking, their model does not appear to guarantee the security of non-aggregate ciphertexts.

3.1 Syntax

In an aggregate signcryption scheme, we consider a situation where many senders wish to send a number of different messages to a single recipient. An aggregate signcryption scheme is a tuple of PPT algorithms ($\texttt{Setup}, \texttt{Gen}_S, \texttt{Gen}_R, \texttt{Signcrypt}, \texttt{Aggregate}, \texttt{Unsigncrypt}$):

– The setup algorithm outputs a set of public parameters $\mathit{param} \xleftarrow{\$} \texttt{Setup}(1^k)$. We will assume that this is an implicit input to all other algorithms.
– The sender key generation algorithm outputs a sender key-pair $(pk_S, sk_S) \xleftarrow{\$} \texttt{Gen}_S(\mathit{param})$.
– The receiver key generation algorithm outputs a receiver key-pair $(pk_R, sk_R) \xleftarrow{\$} \texttt{Gen}_R(\mathit{param})$.
– The signcryption algorithm takes a sender private key, a receiver public key, and a message m, and outputs a ciphertext $C \xleftarrow{\$} \texttt{Signcrypt}(sk_S, pk_R, m)$.
– The aggregation algorithm takes a vector of sender public keys $\boldsymbol{pk_S}$, a receiver public key pk_R and a vector of ciphertexts $\boldsymbol{C}$, and outputs a single ciphertext $C \xleftarrow{\$} \texttt{Aggregate}(\boldsymbol{pk_S}, pk_R, \boldsymbol{C})$ or the invalid symbol $\perp$.
– The unsigncryption algorithm takes a vector of sender public keys $\boldsymbol{pk_S}$, a receiver private key sk_R and a ciphertext C, and either a vector of messages $\boldsymbol{m} \leftarrow \texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R, C)$ or the invalid symbol $\perp$.

We require that if $C^{(i)} \xleftarrow{\$} \texttt{Signcrypt}(sk_S^{(i)}, pk_R, m^{(i)})$ for $1 \le i \le n$ and $C \xleftarrow{\$} \texttt{Aggregate}(\boldsymbol{pk_S}, pk_R, \boldsymbol{C})$, then $\boldsymbol{m} \leftarrow \texttt{Unsigncrypt}(\boldsymbol{pk_S}, pk_R, C)$. We also assume that if $C \xleftarrow{\$} \texttt{Signcrypt}(sk_S, pk_R, m)$ then $\texttt{Aggregate}(pk_S, pk_R, C) = C$. We stress that the aggregation algorithm only uses public information to aggregate the signcryption ciphertexts.

3.2 Confidentiality

We will adapt the security definitions of confidentiality for a signcryption scheme [2, 3] to the aggregate signcryption setting. We wish to forbid the attacker from submitting the ciphertext C^* to the decryption oracle *or any ciphertext can be obtained through aggregation with* C^*. To do this, we incorporate the ideas of [7, 16], and define the span of a ciphertext C sent from S to R to be

$$\langle pk_S^*, pk_R^*, C^* \rangle = \{(\boldsymbol{pk_S}, pk_R^*, \texttt{Aggregate}(\boldsymbol{pk_S}, pk_R^*, \boldsymbol{C})) : \exists\, i \text{ with } pk_S^{(i)} = pk_S^* \text{ and } C^{(i)} = C^*\}\,.$$

Let $SPK(k) = \{(pk_S, sk_S)\}$ be the set of all key pairs that could be output by $\texttt{Gen}_S(1^k)$. We define several security models for (insider) confidentiality. These are both defined in Figure 3.

[1] I.e. there exists a polynomial-time algorithm which can, given a bitstring $\theta \in \{0,1\}^*$, determine whether θ is a public key that could be output by the sender or receiver key generation algorithm

```
$\textsc{Expt}_{\mathcal{A}}^{\texttt{IND-b}}(k)$:
    $(pk_R^*, sk_R^*) \xleftarrow{\$} \texttt{Gen}_R(1^k)$
    $(m_0, m_1, pk_S^*, sk_S^*, \omega) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_U}(pk_R^*)$
    $C^* \xleftarrow{\$} \texttt{Signcrypt}(sk_S^*, pk_R^*, m_b)$
    If $(pk_S^*, sk_S^*) \notin SPK(k)$ then $C^* \leftarrow \perp$
    If $|m_0| \neq |m_1|$ then $C^* \leftarrow \perp$
    $b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_U}(C^*, \omega)$
    Output $b'$

$\mathcal{O}_U(\boldsymbol{pk_S}, C)$:
    Return $\texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, C)$
```

```
$\textsc{Expt}_{\mathcal{A}}^{\texttt{IND-b}}(k)$:
    $(pk_R^*, sk_R^*) \xleftarrow{\$} \texttt{Gen}_R(1^k)$
    $(m_0, m_1, pk_S^*, \omega) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_U}(pk_R^*)$
    Find $sk_S^*$ such that $(pk_S^*, sk_S^*) \in SPK(k)$
    (If no such $sk_S^*$ exists then $C^* \leftarrow \perp$)
    $C^* \xleftarrow{\$} \texttt{Signcrypt}(sk_S^*, pk_R^*, m_b)$
    If $|m_0| \neq |m_1|$ then $C^* \leftarrow \perp$
    $b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_U}(C^*, \omega)$
    Output $b'$

$\mathcal{O}_U(\boldsymbol{pk_S}, C)$:
    Return $\texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, C)$
```

Fig. 3. The IND-CCA2 security model for confidentiality (LHS) and the SKI-IND-CCA2 security model for confidentiality (RHS). In both cases, $\mathcal{A}_2$ is forbidden from submitting a $\mathcal{O}_U$ query $(\boldsymbol{pk_S}, C)$ where $(\boldsymbol{pk_S}, pk_R^*, C) \in \langle pk_S^*, pk_R^*, C^* \rangle$.

The difference between the IND-CCA2 security game and the secret-key ignorant SKI-IND-CCA2 game is whether the attacker is required to output a complete sender key-pair, rather than just the sender public key. The IND-CCA2 game models situations in which the process of registering a public key requires the sender to prove knowledge of the private key; the SKI-IND-CCA2 game models situations in which a sender can register any public key [14]. An attacker's advantage is defined to be

$$Adv_{\mathcal{A}}^{\texttt{IND}}(k) = |\Pr[\textsc{Expt}_{\mathcal{A}}^{\texttt{IND-1}}(k) = 1] - \Pr[\textsc{Expt}_{\mathcal{A}}^{\texttt{IND-0}}(k) = 1]|$$

and a signcryption scheme is said to be (SKI-)IND-CCA2 secure if every PPT attacker has negligible advantage. Outsider security models can be developed in a similar way.

We may define weaker notions of confidentiality for signcryption schemes similar to the IND-RCCA security notion for public-key encryption [13]. These (SKI-)IND-RCCA notions are defined as above except that the $\mathcal{O}_U$ oracle returns `test` whenever $\mathcal{A}_2$ submits a query $(\boldsymbol{pk_S}, C)$ for which there exists i with $pk_S^{(i)} = pk_S^*$ and $m^{(i)} \in \{m_0, m_1\}$ for $\boldsymbol{m} \leftarrow \texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, C)$. This models situations in which the ability to replay a message is not considered a threat to security.

3.3 Integrity

Integrity for aggregate signcryption can be extended from the notions of integrity protection for aggregate signatures. Let $RPK(k) = \{(pk_R, sk_R)\}$ be the set of all key pairs that could be output by $\texttt{Gen}_R(1^k)$. We define two security models for integrity in Figure 4. In both cases, the advantage of an attacker $\mathcal{A}$ is given by $\Pr[\textsc{Expt}_{\mathcal{A}}^{\texttt{UF}}(k) = 1]$ and a scheme is said to be (SKI-)UF-CMA secure if every PPT attacker $\mathcal{A}$ has negligible advantage.

We note that there are some dangers about using a scheme proven secure in the UF-CMA security model: it may still have the property that an attacker can produce a ciphertext which is a forgery from some sender S to some receiver R (but cannot compute R's private decryption key). The existence of such a ciphertext appears to imply that S sent some message to R, although the exact nature of the message is unknown, which may be damaging to R's reputation.

It is possible to describe a "strong" unforgeability model for aggregate signcryption by insisting that the attacker forge an output $(\boldsymbol{pk_S}, pk_R^*, C^*)$ where $(\boldsymbol{pk_S}, pk_R^*, C^*) \notin \langle (pk_S^*, pk_R^*, C) \rangle$ for every response C returned by an $\mathcal{O}_U(pk_R^*, m)$ query. It is unclear as to whether this definition has a practical application and so we will not pursue it further in this work.

3.4 Denial of Decryption

Aggregation is designed to combine signcryption ciphertexts intended for a single receiver. The basic syntax does not discuss what might happen if a user tries to aggregate signcryption ci-

```
$\textsc{Expt}^{\texttt{UF}}_{\mathcal{A}}(k)$:
  $(pk_S^*, sk_S^*) \xleftarrow{\$} \texttt{Gen}_S(1^k)$
  $(\boldsymbol{pk_S}, pk_R^*, sk_R^*, C^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_S}(pk_S^*)$
  If $(pk_R^*, sk_R^*) \notin RPK(k)$ then output 0
  $\boldsymbol{m} \leftarrow \texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, C^*)$
  If $\boldsymbol{m} = \perp$ then output 0
  If $\exists\, i$ such that $pk_S^{(i)} = pk_S^*$ and $(pk_R^*, m^{(i)})$
    was not queried to $\mathcal{O}_S$ then output 1
  Else output 0

$\mathcal{O}_S(pk_R, m)$:
  Return $\texttt{Signcrypt}(sk_S^*, pk_R, m)$
```

```
$\textsc{Expt}^{\texttt{UF}}_{\mathcal{A}}(k)$:
  $(pk_S^*, sk_S^*) \xleftarrow{\$} \texttt{Gen}_S(1^k)$
  $(\boldsymbol{pk_S}, pk_R^*, C^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_S}(pk_S^*)$
  Find $sk_R^*$ such that $(pk_R^*, sk_R^*) \in RPK(k)$
  (If no such $sk_R^*$ exists then output 0)
  $\boldsymbol{m} \leftarrow \texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, C^*)$
  If $\boldsymbol{m} = \perp$ then output 0
  If $\exists\, i$ such that $pk_S^{(i)} = pk_S^*$ and $(pk_R^*, m^{(i)})$
    was not queried to $\mathcal{O}_S$ then output 1
  Else output 0

$\mathcal{O}_S(pk_R, m)$:
  Return $\texttt{Signcrypt}(sk_S^*, pk_R, m)$
```

Fig. 4. The UF-CMA security model for aggregate signcryption (LHS) and the SKI-UF-CMA security model for aggregate signcryption (RHS).

phertexts which are intended for different receivers or which are invalid. An attacker may inject a false ciphertext into a batch of legitimate ciphertexts and cause the aggregation algorithm to output an invalid aggregate ciphertext. This will prevent the receiver from receiving legitimate messages. We term this a *denial of decryption* (DoD) attack after the similar effect that can occur in certificateless cryptosystems [23]. To prevent DoD attacks, we may insist that the aggregation algorithm refuse to aggregate invalid signcryption ciphertexts. Note that this attack can easily be prevented in aggregate signature schemes by insisting that the aggregation algorithm check that validity of all the individual signatures before computing the aggregate signature; however, most signcryption schemes do not allow a third-party to check the validity of a signcryption ciphertext[2].

In defining DoD security, we note that the aim of the attack is to prevent a legitimate receiver from unsigncrypting an aggregate of signcryption ciphertexts. Hence, it is reasonable to assume that the receiver is an uncorrupted entity; however, it is possible that the attacker is a legitimate sender who wishes to interfere with the communication between other senders and the (honest) receiver by injecting malicious ciphertexts into the aggregation process. Thus, we obtain the security model in Figure 5. An attacker $\mathcal{A}$ has advantage $\Pr[\textsc{Expt}^{\texttt{DoD}}_{\mathcal{A}}(k) = 1]$ and an aggregate signcryption scheme is DoD-secure if every PPT attacker has negligible advantage.

```
$\textsc{Expt}^{\texttt{DoD}}_{\mathcal{A}}(k)$:
  $(pk_R^*, sk_R^*) \xleftarrow{\$} \texttt{Gen}_R(1^k)$
  $(\boldsymbol{pk_S}, \boldsymbol{C}) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_U}(pk_R^*)$
  For $i = 1, \ldots, n$:
    $m^{(i)} \leftarrow \texttt{Unsigncrypt}(pk_S^{(i)}, sk_R^*, C^{(i)})$
  $\hat{C} \xleftarrow{\$} \texttt{Aggregate}(\boldsymbol{pk_S}, pk_R, \boldsymbol{C})$
  If $\hat{C} = \perp$ then output 0
  $\hat{\boldsymbol{m}} \leftarrow \texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, \hat{C})$
  If $\boldsymbol{m} = \hat{\boldsymbol{m}}$ then output 0
  Else output 1

$\mathcal{O}_U(\boldsymbol{pk_S}, C)$:
  Return $\texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, C)$
```

Fig. 5. The denial of decryption security model for aggregate signcryption.

Essentially, DoD-security states that if the aggregation algorithm does not output $\perp$ then each signcryption ciphertext is valid and that the unsigncryption of the aggregate ciphertext is the same as the unsigncryption of the individual ciphertexts. One "simple" way to convert an aggregate

[2] We note that this problem also occurs with history-free aggregate signatures/MACs [16].

signcryption scheme without DoD-security into a scheme with DoD-security is to append a NIZK proof that the signcryption operation has been performed correctly using the public key pk_R^* to the ciphertext. This can always be achieved as NIZK proofs exist for all NP languages [17]; however, in practice, these NIZK schemes are unusable and so it remains a challenge to construct practical aggregate signcryption schemes with DoD security.

4 Generic Constructions

Of the four methods to generically combine public-key encryption and digital signatures, only two are candidates for constructing secure aggregate signcryption schemes: encrypt-then-sign ($\mathcal{E}t\mathcal{S}$) and commit-then-encrypt-and-sign ($\mathcal{C}t\mathcal{E}\&\mathcal{S}$)[3]. These constructions were first analysed for non-aggregate signcryption schemes in the work of An, Dodis and Rabin [2]. In this section, we present the $\mathcal{E}t\mathcal{S}$ scheme. The $\mathcal{C}t\mathcal{E}\&\mathcal{S}$ construction is discussed in Appendix A.

If $(\texttt{SigGen}, \texttt{Sign}, \texttt{SigAgg}, \texttt{Ver})$ is an aggregate signature scheme and $(\texttt{EncGen}, \texttt{Enc}, \texttt{Dec})$ is a public key encryption scheme, then we may construct an $\mathcal{E}t\mathcal{S}$ aggregate signcryption scheme is given in Figure 6. This scheme does not make use of a `Setup` algorithm. Throughout this section, we will assume that an entity X has a unique identity ID_X and that there exists a mechanism to link the public key of X to their identity ID_X. This mechanism can be provided by a certificate authority or may be implicit (e.g. if $ID_X = pk_X$).

```
$\texttt{Gen}_S(1^k)$:
  $(pk_S, sk_S) \xleftarrow{\$} \texttt{SigGen}(1^k)$
  Output $(pk_S, sk_S)$

$\texttt{Gen}_R(1^k)$:
  $(pk_R, sk_R) \xleftarrow{\$} \texttt{EncGen}(1^k)$
  Output $(pk_R, sk_R)$

$\texttt{Signcrypt}(sk_S, pk_R, m)$:
  $m' \leftarrow (ID_S, m)$
  $\chi \xleftarrow{\$} \texttt{Enc}(pk_R, m')$
  $m'' \leftarrow (ID_R, \chi)$
  $\sigma \xleftarrow{\$} \texttt{Sign}(sk_S, m'')$
  Output $C \leftarrow (\chi, \sigma)$

$\texttt{Aggregate}(pk_R, \boldsymbol{C})$:
  Parse $\boldsymbol{C}$ as $((\chi^{(1)}, \sigma^{(1)}), \ldots, (\chi^{(n)}, \sigma^{(n)}))$
  $\sigma \xleftarrow{\$} \texttt{SigAgg}(\boldsymbol{pk_S}, \boldsymbol{\sigma})$
  Output $(\boldsymbol{\chi}, \sigma)$

$\texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R, C)$:
  Parse $C$ as $(\boldsymbol{\chi}, \sigma)$
  $\boldsymbol{m''} \leftarrow ((ID_R, \chi^{(1)}), \ldots, (ID_R, \chi^{(n)}))$
  If $\texttt{Ver}(\boldsymbol{pk_S}, \boldsymbol{m''}, \sigma) = \perp$ then output $\perp$
  For $1 \leq i \leq n$:
    $m'^{(i)} \leftarrow \texttt{Dec}(sk_R, \chi^{(i)})$
    Parse $m'^{(i)}$ as $(ID_S^{(i)}, m^{(i)})$
    If $pk_S^{(i)}$ is not the public key of $ID_S^{(i)}$ then output $\perp$
  Output $\boldsymbol{m}$
```

Fig. 6. The Encrypt-then-Sign Construction

Theorem 1. *The following results hold:*

1. *If $\mathcal{A}$ is a PPT attacker against the insider SKI-UF-CMA unforgeability of the signcryption scheme, then there exists a PPT attacker $\mathcal{B}$ against the UF-CMA unforgeability of the signature scheme such that $Adv_{\mathcal{A}}^{\mathsf{UF}}(k) \leq Adv_{\mathcal{B}}^{\mathsf{UF}}(k)$.*
2. *If $\mathcal{A}$ is a PPT attacker against the IND-RCCA security of the signcryption scheme and the set $SKP(k)$ is polynomial-time recognisable, then there exists a PPT attacker $\mathcal{B}$ against the IND-RCCA security of the encryption scheme such that $Adv_{\mathcal{A}}^{\mathsf{RCCA}}(k) \leq Adv_{\mathcal{B}}^{\mathsf{RCCA}}(k)$.*

Surprisingly, despite the publicly verifiable signature, the construction does not provide any defence against denial of decryption attacks. Even if the aggregation algorithm checks the validity of the signature in each of the ciphertexts $C^{(i)}$, it will not necessarily be able to detect whether

[3] Of the remaining two constructions, encrypt-and-sign ($\mathcal{E}\&\mathcal{S}$) is insecure and sign-then-encrypt ($\mathcal{S}t\mathcal{E}$) is not suitable for aggregation.

each public-key ciphertext $\chi^{(i)}$ is valid or whether the decryption contains the correct identity $ID_S^{(i)}$. The construction does have the advantage that if the aggregation algorithm does check the individual signatures, then the unsigncryption algorithm will only fail to unsigncrypt the messages associated with invalid encryption ciphertexts. The receiver can still recover legitimate messages. Thus, the attacker cannot deny the receiver information, but can artificially inflate the size of aggregated ciphertext.

5 Direct Construction

One method for constructing a secure encryption scheme is to apply the CHK construction to an identity-based encryption scheme [9, 20]. To fully encrypt a message, a signature key-pair (pk, sk) is created for each ciphertext, the message is encrypted using the identity-based encryption scheme with the signature public key pk as the identity. A signcryption scheme can be produced using an $\mathcal{E}t\mathcal{S}$ construction if the resulting encryption ciphertext is signed. It is clear that the bandwidth of the $\mathcal{E}t\mathcal{S}$ construction can be reduced if an aggregate signature scheme is used in both the CHK construction and as the signature scheme. Furthermore, if the signature scheme allows for the aggregation of aggregate signatures, then the result is an aggregate signcryption scheme.

We aim to improve efficiency by applying this methodology to the Boneh-Franklin identity-based encryption scheme [10] and the BGLS aggregate signature scheme [11]. We are able to improve efficiency by re-using an element generated as part of the identity-based encryption scheme as the ephemeral signature public key in the CHK construction. The resulting scheme makes use of a group $\mathbb{G}$ generated by an element g and a non-degenerate, bilinear map $e : \mathbb{G}^2 \to \mathbb{G}_T$. The scheme is given in Figure 7. The proof relies on the random oracle methodology.

```
Gen_S(1^k):
  $x_S \xleftarrow{\$} \mathbb{Z}_q$
  $y_S \leftarrow g^{x_S}$
  $pk_S \leftarrow y_S$; $sk_S \leftarrow x_S$
  Output $(pk_S, sk_S)$

Gen_R(1^k):
  $x_R \xleftarrow{\$} \mathbb{Z}_q$
  $y_R \leftarrow g^{x_R}$
  $pk_R \leftarrow y_R$; $sk_R \leftarrow x_R$
  Output $(pk_R, sk_R)$

Aggregate($\boldsymbol{pk_S}, pk_R, \boldsymbol{C}$):
  Parse $C^{(i)}$ as $(T^{(i)}, c^{(i)}, \sigma^{(i)})$
  If $T^{(i)} = T^{(j)}$ for $i \neq j$
    Output $\perp$
  For $i = 1, \dots, n$:
    $P_1 \leftarrow e(T^{(i)}, H_1(T^{(i)}, c^{(i)}, ID_S^{(i)}))$
    $P_2 \leftarrow e(y_S^{(i)}, H_2(T^{(i)}, c^{(i)}, ID_R))$
    If $e(g, \sigma^{(i)}) \neq P_1 P_2$ then output $\perp$
  $\sigma \leftarrow \prod_{i=1}^{n} \sigma^{(i)}$
  Output $(\boldsymbol{T}, \boldsymbol{c}, \sigma)$

Signcrypt($sk_S, pk_R, m$):
  $t \xleftarrow{\$} \mathbb{Z}_q$; $T \leftarrow g^t$
  $K \leftarrow H_K(e(y_R, H_t(T, ID_S))^t)$
  $c \leftarrow \textsc{Enc}_K(m)$
  $\sigma \leftarrow H_1(T, c, ID_S)^t H_2(T, c, ID_R)^{x_S}$
  $C \leftarrow (T, c, \sigma)$
  Output $C$

Unsigncrypt($\boldsymbol{pk_S}, sk_R, C$)
  Parse $C$ as $(\boldsymbol{T}, \boldsymbol{c}, \sigma)$
  If $T^{(i)} = T^{(j)}$ for $i \neq j$ then output $\perp$
  $P_1 \leftarrow \prod_{i=1}^{n} e(T^{(i)}, H_1(T^{(i)}, c^{(i)}, ID_S^{(i)}))$
  $P_2 \leftarrow \prod_{i=1}^{n} e(y_S^{(i)}, H_2(T^{(i)}, c^{(i)}, ID_R))$
  If $e(g, \sigma) \neq P_1 P_2$ then output $\perp$
  For $i = 1, \dots, n$:
    $K \leftarrow H_K(e(T^{(i)}, H_t(T^{(i)}, ID_S^{(i)}))^{x_R})$
    $m^{(i)} \leftarrow \textsc{Dec}_K(c^{(i)})$
    If $m^{(i)} = \perp$ then output $\perp$
  Output $\boldsymbol{m}$
```

Fig. 7. The Direct Construction

The scheme is secure if it is instantiated on a group $\mathbb{G}$ of order q (which depends on the security parameter k) on which the CDH and BDH problems are hard. In other words, the following advantages are negligible for every PPT attacker $\mathcal{A}$:

$$Adv_{\mathcal{A}}^{\texttt{CDH}}(k) = \Pr\left[\mathcal{A}(g^a, g^b) = g^{ab} \ : \ a, b \xleftarrow{\$} \mathbb{Z}_q\right]$$

$$Adv_{\mathcal{A}}^{\texttt{BDH}}(k) = \Pr\left[\mathcal{A}(g^a, g^b, g^c) = e(g,g)^{abc} \ : \ a, b, c \xleftarrow{\$} \mathbb{Z}_q\right]$$

This construction is DoD secure (under reasonable assumptions about the action of the IND-CPA symmetric encryption scheme). The only way that a ciphertext would fail to decrypt is if σ does not satisfy the appropriate algebraic relation; however, this is publicly checkable and will be checked by the aggregation algorithm. Therefore, the aggregation algorithm will refuse to form aggregated ciphertexts which do not decrypt successfully. The scheme has the added advantage that the aggregation algorithm can easily detect ciphertexts which will not aggregate correctly – these are the invalid ciphertext (which can be publicly checked) and the ciphertexts which share a common value of T. In the latter case, the entity performing aggregation can solve the problem simply by forming multiple aggregate ciphertexts (ensuring that all aggregate ciphertexts are constructed from ciphertext with different T values).

Theorem 2. *Suppose the symmetric encryption scheme* $(\texttt{Enc}, \texttt{Dec})$ *has the property that for all* $K \in \{0,1\}^k$ *and* $C \in \{0,1\}^*$*, we have that* $\texttt{Dec}_K(C) \neq \perp$*. Then the aggregate signcryption scheme has perfect DoD security. In other words, for any (unbounded) attacker* $\mathcal{A}$ *against the DoD security of the aggregate signcryption scheme, we have that* $\Pr[\textsc{Expt}_{\mathcal{A}}^{DoD} = 1] = 0$.

Proof: Let $(\boldsymbol{pk_S}, \boldsymbol{C})$ be such that $\hat{C} \leftarrow \texttt{Aggregate}(\boldsymbol{pk_S}, pk_R^*, \boldsymbol{C})$ satisfies $\hat{C} \neq \perp$. Since $\hat{C} \neq \perp$ we must have that every $C^{(i)}$ is computed using a different value of $T^{(i)}$ and that

$$e(g, \sigma^{(i)}) = e(T^{(i)}, H_1(T^{(i)}, c^{(i)}, ID_S^{(i)})) \cdot e(y_S^{(i)}, H_2(T^{(i)}, c^{(i)}, ID_R)) \,.$$

Thus, if we parse $\hat{C}$ as $(\boldsymbol{T}, \boldsymbol{c}, \hat{\sigma})$, we have that

$$e(g, \hat{\sigma}) = \prod_{i=1}^{n} e(T^{(i)}, H_1(T^{(i)}, c^{(i)}, ID_S^{(i)})) \cdot e(y_S^{(i)}, H_2(T^{(i)}, c^{(i)}, ID_R)) \,.$$

Consider the message vectors $\texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, \hat{C})$ and $\texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, \boldsymbol{C})$ (where the latter message vector is computed component-wise). In neither case will the unsigncryption algorithm output $\perp$. However, an inspection of the unsigncryption algorithm will show that all later operations depends only on $\boldsymbol{T}$, $\boldsymbol{c}$, $\boldsymbol{pk_S}$ and sk_R^*. These values are identical in $(\boldsymbol{pk_S}, \boldsymbol{C})$ and $(\boldsymbol{pk_S}, \hat{C})$; hence, $\texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, \hat{C}) = \texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, \boldsymbol{C})$ and so there exist no $(\boldsymbol{pk_S}, \boldsymbol{C})$ which could cause the attacker to win the DoD game. □

The confidentiality/integrity security proof is more familiar.

Theorem 3. *Let* $\mathcal{A}$ *be an attacker which makes at most* q_X *queries to the hash oracle* H_X *(for* $X \in \{1, 2, t, K\}$*),* q_S *queries to a signcryption oracle, and* q_U *queries to the unsigncryption oracle. Suppose* $\mathbb{G}$ *has prime order* q*. In the SKI-IND-CCA2 confidentiality model, there exists an algorithm* $\mathcal{B}$ *to solve the CDH problem, an algorithm* $\mathcal{B}'$ *to solve the BDH problem, and an attacker* $\mathcal{B}''$ *against the IND-CPA security of the symmetric encryption scheme such that*

$$Adv_{\mathcal{A}}^{\mathit{IND}}(k) \leq 2Adv_{\mathcal{B}}^{\mathit{CDH}}(k) + 2(q_K + q_U)Adv_{\mathcal{B}'}^{\mathit{BDH}}(k) + Adv_{\mathcal{B}''}^{\mathit{sym}}(k) + \frac{2(q_1 + q_2 + q_t + q_U)}{q} \,.$$

In the SKI-UF-CMA unforgeability model, there exists an algorithm $\mathcal{B}$ *to solve the CDH problem such that*

$$Adv_{\mathcal{A}}^{\mathit{UF}}(k) \leq q_2 Adv_{\mathcal{B}}^{\mathit{CDH}}(k) + \frac{q_S(q_S + q_2) + 1}{q} \,.$$

The following lemma will be useful:

Lemma 1. *Suppose $C^* = (T^*, c^*, \sigma^*)$ is a ciphertext computed using sender key pair (pk_S^*, sk_S^*) and receiver key pair (pk_R^*, sk_R^*). If $(\boldsymbol{pk_S}, pk_R^*, C)$ satisfies $\texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R^*, C) \neq \perp$ and there exists j such that $(pk_S^{(j)}, T^{(j)}, c^{(j)}) = (pk_S^*, T^*, c^*)$, then $(\boldsymbol{pk_S}, pk_R^*, C) \in \langle pk_S^*, pk_R^*, C^* \rangle$.*

Proof: Since $\texttt{Unsigncrypt}(\boldsymbol{pk_S}, pk_R^*, C) \neq \perp$, we must have

$$e(g, \sigma) = \prod_{i=1}^{n} e(g, \sigma^{(i)})$$

where

$$e(g, \sigma^{(i)}) = e(T^{(i)}, H_1(T^{(i)}, c^{(i)}, ID_S^{(i)})) \cdot e(pk_S^{(i)}, H_2(T^{(i)}, c^{(i)}, ID_R^*)) \,.$$

So if we define $C^{(i)} = (T^{(i)}, c^{(i)}, \sigma^{(i)})$ then $C = \texttt{Aggregate}(\boldsymbol{pk_S}, pk_R^*, \boldsymbol{C})$. However, since e is a bilinear map between prime order groups, there exists a unique value $\sigma^{(j)}$ such that

$$e(g, \sigma^{(j)}) = e(T^*, H_1(T^*, c^*, ID_S^*)) \cdot e(pk_S^*, H_2(T^*, c^*, ID_R^*))$$

and so we must have that $\sigma^{(j)} = \sigma^*$. Therefore, $(\boldsymbol{pk_S}, pk_R^*, C) \in \langle pk_S^*, pk_R^*, C^* \rangle$. □

Proof of Confidentiality: Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an attacker against the insider IND-CCA2 security of the signcryption scheme. We assume that starred variables are variables computed during the creation of the challenge ciphertext. Let S_i^b be the event that game G_i^b outputs 1. Let game G_1^b be the $\text{Expt}_{\mathcal{A}}^{\texttt{IND-b}}(k)$ game. We prove our result through game hopping [5, 29].

Let G_2^b be identical to G_1^b except that the game outputs 0 if $\mathcal{A}_1$ queries the H_1, H_2, H_t or unsigncryption oracle with T^* as one of the inputs. Since the attacker does not gain any information about T^* until the challenge phase, the probability that this occurs is bounded by $q_1 + q_2 + q_t + q_U/q$. Thus, $|\Pr[S_2^b] - \Pr[S_1^b]| \leq (q_1 + q_2 + q_t + q_U)/q$.

Let G_3^b be the game in which the unsigncryption oracle returns $\perp$ if $\mathcal{A}_2$ queries the unsigncryption oracle on a public-key vector $\boldsymbol{y_S}$ and a ciphertext $(\boldsymbol{T}, \boldsymbol{c}, \sigma)$ for which there exists i such that $T^{(i)} = T^*$ and $(y_S^{(i)}, c^{(i)}) \neq (y_S^*, c^*)$. Suppose that the unsigncryption oracle on G_3^b returns $\perp$ when G_2^b would have returned some message vector $\boldsymbol{m} \neq \perp$. Intuitively, this means that there has been a forgery on the signature used in the CHK construction. We define an attacker $\mathcal{B}$ against the CDH problem which breaks the scheme if $\mathcal{A}$ makes such a query. The attacker runs as in Figure 8 (where the hash oracles which are not directly defined return a random result when queried on a new value and the attacker is assumed never to query the same hash value twice).

The crux of the proof is in the way in which the attacker $\mathcal{B}$ simulates the H_1 and H_2 oracles. All H_1 oracle queries that do not involve T^* are answered with a random group element g^α. The H_1 query on (T^*, c^*, ID_S^*) is handled in the same way and this allows $\mathcal{B}$ to compute the challenge ciphertext without knowing sk_S^*. All H_1 queries involving T^* are answered with a random group element $U^{*\alpha}$. All H_2 oracle queries are answered with a random group element g^β. This allows the simulation to answer all queries $(\boldsymbol{pk_S}, C)$ for which $T^{(i)} \neq T^*$ for all i. If there exists i with $(T^{(i)}, pk_S^{(i)}, c^{(i)}) = (T^*, pk_S^*, c^*)$ then $(\boldsymbol{pk_S}, pk_R^*, C) \in \langle pk_S^*, pk_R^*, C^* \rangle$ by Lemma 1 which is an illegal query.

If $\mathcal{A}$ submits a valid query $(\boldsymbol{pk_S}, C)$ to the unsigncryption oracle for which there exists some i such that $T^{(i)} = T^*$ but $(pk_S^{(i)}, c^{(i)}) \neq (pk_S^*, c^*)$ then this must be the only only value for which $T^{(i)} = T^*$ (as otherwise the unsigncryption algorithm would return $\perp$). Moreover, we must have that

$$e(g, \sigma) = \prod_{j=1}^{n} e(T^{(j)}, H_1(T^{(j)}, c^{(j)}, ID_S^{(j)})) \cdot e(y_S^{(j)}, H_2(T^{(j)}, c^{(j)}, ID_R^*))$$

which means, by the definition of the H_1 and H_2 oracle, that

$$\sigma = Z^{\alpha_i} y_S^{(i)\beta_i} \prod_{j=1, j\neq i}^{n} T^{(j)\alpha_j} y_S^{(j)\beta_j}$$

```
$\mathcal{B}(T^*, U^*)$:
  $x_R^* \xleftarrow{\$} \mathbb{Z}_q$; $y_R^* \leftarrow g^{x_R^*}$
  Initiate lists H$_1$-LIST and H$_2$-LIST
  $(m_0, m_1, y_S^*, \omega) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_U}(y_R^*)$
    If $\mathcal{A}_1$ queries $\mathcal{O}_U(\boldsymbol{pk_S}, C)$
      $\boldsymbol{m} \xleftarrow{\$} \mathtt{Unsigncrypt}(\boldsymbol{y_S}, x_R^*, C)$
      If $\exists$ (unique) $i$ such that $T^{(i)} = T^*$
        Output $\perp$ and halt
      Return $\boldsymbol{m}$
    If $\mathcal{A}_1$ queries $\mathcal{O}_{H_1}(T, c, ID_S)$
      If $T = T^*$ then output $\perp$ and halt
      $\alpha \xleftarrow{\$} \mathbb{Z}_q$
      Add $(T, c, ID_S, \alpha)$ to H$_1$-LIST
      Return $g^\alpha$
    If $\mathcal{A}_1$ queries $\mathcal{O}_{H_2}(T, c, ID_R)$
      If $T = T^*$ then output $\perp$ and halt
      $\beta \xleftarrow{\$} \mathbb{Z}_q$
      Add $(T, c, ID_R, \beta)$ to H$_2$-LIST
      Return $g^\beta$
  $K^* \leftarrow H_K(e(T^*, H_t(T^*, ID_S))^{x_R^*})$
  $c^* \leftarrow \mathrm{ENC}_{K^*}(m_b)$
  $\alpha^* \xleftarrow{\$} \mathbb{Z}_q$
  Add $(T^*, c^*, ID_S^*, \alpha^*)$ to H$_1$-LIST
  Compute $H_2(T^*, c^*, ID_R^*)$
  Find $(T^*, c^*, ID_R^*, \beta^*)$ on H$_2$-LIST
  $\sigma^* \leftarrow T^{*\alpha^*} y_S^{*\beta^*}$
  $C^* \leftarrow (T^*, c^*, \sigma^*)$
  (continued in next column)

$b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_U}(C^*, \omega)$
  If $\mathcal{A}_1$ queries $\mathcal{O}_U(\boldsymbol{y_S}, C)$
    $\boldsymbol{m} \leftarrow \mathtt{Unsigncrypt}(\boldsymbol{y_S}, x_R^*, C)$
    If $\boldsymbol{m} = \perp$ then return $\perp$
    Parse $C$ as $(\boldsymbol{T}, \boldsymbol{c}, \sigma)$
    If $\forall\ i$ we have $T^{(i)} \neq T^*$
      Return $\boldsymbol{m}$
    Else $\exists$ unique $i$ with $T^{(i)} = T^*$
      $Z \leftarrow \{\sigma / y_S^{(i)\beta_i} \prod_{j=1, j\neq i}^{n} T^{(j)\alpha_j} y_S^{(j)\beta_j}\}^{1/\alpha_i}$ where
          $(T^{(j)}, c^{(j)}, ID_S^{(j)}, \alpha_j) \in$ H$_1$-LIST
          $(T^{(j)}, c^{(j)}, ID_R^*, \beta_j) \in$ H$_2$-LIST
      Output $Z$ and halt
  If $\mathcal{A}_1$ queries $\mathcal{O}_{H_1}(T, c, ID_S)$
    If $T \neq T^*$
      $\alpha \xleftarrow{\$} \mathbb{Z}_q$
      Add $(T, c, ID_S, \alpha)$ to H$_1$-LIST
      Return $g^\alpha$
    Else
      $\alpha \xleftarrow{\$} \mathbb{Z}_q$
      Add $(T^*, c, ID_S, \alpha)$ to H$_1$-LIST
      Return $U^{*\alpha}$
  If $\mathcal{A}_1$ queries $\mathcal{O}_{H_2}(T, c, ID_R)$
    $\beta \xleftarrow{\$} \mathbb{Z}_q$
    Add $(T, c, ID_R, \beta)$ to H$_2$-LIST
    Return $g^\beta$
Output $\perp$
```

Fig. 8. The attacker $\mathcal{B}$ against the CDH problem. The hash oracles H_K and H_t which are not directly defined return a random result when queried on a new value. The attacker is assumed never to query a hash oracle on the same value twice.

where Z is the solution to the CDH problem for (T^*, U^*). Hence, $\mathcal{B}$ can solve the CDH problem and so $|\Pr[S_3^b] - \Pr[S_2^b]| \leq Adv_{\mathcal{B}}^{\texttt{CDH}}(k)$.

Let G_4^b be the game in which outputs $\perp$ if the attacker $\mathcal{A}$ makes a H_K query on the value $e(y_R^*, H_t(T^*, ID_S^*))^{t^*}$ or if $\mathcal{A}$ makes an unsigncryption oracle query that would evaluate H_K on this value. We give an attacker $\mathcal{B}'$ which solves the BDH problem with non-negligible probability if $\mathcal{A}$ makes such a query. The attacker runs as in Figure 9. Note the important role being played by the IND-CCA2 model here – the unsigncryption oracle can only correctly compute the key K on ciphertexts which compute $H_t(T^{(i)}, ID_S^{(i)})$ as g^γ. However, if the ciphertext forces the unsigncryption oracle to compute $H_t(T^*, ID_S^*) = U^*$ then the unsigncryption oracle will fail. However, this will only occur if there exists j such that $(T^{(j)}, ID_S^{(j)}) = (T^*, ID_S^*)$ and by the conditions imposed in G_3^b this cannot occur.

If an attacker queries a "forbidden" H_K query (or causes the unsigncryption oracle to do so) then the H_K oracle will add an entry (X, K) to H$_K$-List where X is the solution to the BDH problem on (T^*, U^*, Y^*). The simulation may break down at this point, but this does not matter since the solution to the problem we require is on H$_K$-List at this point. Thus, if such a query occurs, then $\mathcal{B}'$ outputs the solution to the BDH problem with probability at least $1/(q_U + q_K)$. Hence, $|\Pr[S_3^b] - \Pr[S_4^b]| \leq (q_U + q_K) Adv_{\mathcal{B}'}^{\texttt{BDH}}(k)$.

```
$\mathcal{B}'(T^*, U^*, Y^*)$:
  $y_R^* \leftarrow Y^*$
  Initialise hash oracle lists
  $(m_0, m_1, y_S^*, \omega) \xleftarrow{\$} \mathcal{A}_1(y_R^*)$
  $K^* \xleftarrow{\$} \{0,1\}^k$
  $c^* \leftarrow \texttt{Enc}_{K^*}(m_b)$
  $\alpha^*, \beta^* \xleftarrow{\$} \mathbb{Z}_q$
  Add $(T^*, c^*, ID_S^*, \alpha^*)$ to H$_1$-List
  Add $(T^*, c^*, ID_R^*, \beta^*)$ to H$_2$-List
  $\sigma^* \leftarrow T^{*\alpha^*} y_S^{*\beta^*}$
  $C^* \leftarrow (T^*, c^*, \sigma^*)$
  $b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_U}(C^*, \omega)$
  Randomly choose an entry $(X, K) \in$ H$_K$-List
  Output $X$

$H_1(T, c, ID)$:
  $\alpha \xleftarrow{\$} \mathbb{Z}_q$
  Add $(T, c, ID, \alpha)$ to H$_1$-List
  Return $g^\alpha$

$H_2(T, c, ID)$:
  $\beta \xleftarrow{\$} \mathbb{Z}_q$
  Add $(T, c, ID, \beta)$ to H$_2$-List
  Return $g^\beta$

$H_t(T, ID)$:
  If $(T, ID) = (T^*, ID_S^*)$ then return $U^*$
  Else
    $\gamma \xleftarrow{\$} \mathbb{Z}_q$
    Add $(T, ID, \gamma)$ to H$_t$-List
    Return $g^\gamma$

$H_K(X)$:
  $K \xleftarrow{\$} \{0,1\}^k$
  Add $(X, K)$ to H$_K$-List
  Return $K$

$\texttt{Unsigncrypt}(\boldsymbol{y_S}, C)$:
  Parse $C$ as $(\boldsymbol{T}, \boldsymbol{c}, \sigma)$
  If $\exists\, i$ such that $T^{(i)} = T^*$ then return $\perp$
  If $T^{(i)} = T^{(j)}$ for some $i \neq j$ then return $\perp$
  $P_1 \leftarrow \prod_{i=1}^n e(T^{(i)}, H_1(T^{(i)}, c^{(i)}, ID_S^{(i)}))$
  $P_2 \leftarrow \prod_{i=1}^n e(y_S^{(i)}, H_2(T^{(i)}, c^{(i)}, ID_R^*))$
  If $e(g, \sigma) \neq P_1 \cdot P_2$ then return $\perp$
  For $i = 1, \ldots, n$:
    Compute $H_t(T^{(i)}, ID_S^{(i)})$
    Find $(T^{(i)}, ID_S^{(i)}, \gamma)$ on H$_t$-List
    $K \leftarrow H_K(e(y_R^*, T^{(i)})^\gamma)$
    $m^{(i)} \leftarrow \texttt{Dec}_K(c^{(i)})$
    If $m^{(i)} = \perp$ then output $\perp$
  Return $\boldsymbol{m}$
```

Fig. 9. The attacker $\mathcal{B}'$ against the BDH problem. It is assumed that any oracle query made by the attacker is answered by the appropriate given simulator. It is also assumed that the first stage attacker $\mathcal{A}_1$ does not query the H_1, H_2, H_t or `Unsigncrypt` oracle on any value involving T^* and that the attacker never queries an oracle on the same value twice.

However, this means that the key that is used to compute the challenge ciphertext isn't used for any other purpose in the game G_4^b. We may therefore relate $|\Pr[S_4^0] - \Pr[S_4^1]|$ to the difficulty of breaking the symmetric encryption scheme. We give an attacker $\mathcal{B}''$ against the IND-CPA security of the symmetric scheme in Figure 10. We have that $|\Pr[S_4^0] - \Pr[S_4^1]| \leq Adv_{\mathcal{B}''}^{\mathtt{sym}}(k)$.

```
$\mathcal{B}_1''(1^k)$:
  $x_R^* \xleftarrow{\$} \mathbb{Z}_q$; $y_R^* \xleftarrow{\$} \mathbb{Z}_q$
  $(m_0, m_1, y_S^*, \omega) \xleftarrow{\$} \mathcal{A}_1(y_R^*)$
  Output $(m_0, m_1)$

$\mathcal{B}_2''(c^*)$:
  $t^* \xleftarrow{\$} \mathbb{Z}_q$; $T^* \leftarrow g^{t^*}$
  $\sigma^* \leftarrow H_1(T^*, c^*, ID_S^*)^{t^*} H_2(T^*, c, ID_R^*)^{x_S^*}$
  $C^* \leftarrow (T^*, c^*, \sigma^*)$
  $b' \xleftarrow{\$} \mathcal{A}_2(C^*, \omega)$
  Output $b'$
```

Fig. 10. The attacker $\mathcal{B}''$ against the IND-CPA security of the symmetric encryption scheme. If the attacker queries any oracle, then the response is computed using an oracle with the correct distribution. We assume that $\mathcal{A}_1$ never makes a H_1, H_2, H_t or `Unsigncrypt` query involving T^* (from Game 1), that $\mathcal{A}_2$ never makes a valid `Unsigncrypt` query involving T^* (from Game 2) and that $\mathcal{A}$ never makes a H_K query on the value $e(y_R^*, H_t(T^*, ID_S^*))^{t^*}$ or makes an `Unsigncrypt` query that forces this query to be made.

Therefore, putting all the elements together we have that

$$Adv_{\mathcal{A}}^{\mathtt{IND}}(k) \leq \frac{2(q_1 + q_2 + q_t + q_U)}{q} + 2Adv_{\mathcal{B}}^{\mathtt{CDH}}(k) + 2(q_K + q_U)Adv_{\mathcal{B}'}^{\mathtt{BDH}} + Adv_{\mathcal{B}''}^{\mathtt{sym}}(k)\,.$$

This proves the aggregate signcryption scheme is confidential. □

Proof of Unforgeability: The proof of insider unforgeability is much simpler. It essentially relies on the fact that this is an encrypt-then-sign construction, albeit with certain components aggregated. Suppose $\mathcal{A}$ is an SKI-UF-CMA attacker against the unforgeability of the aggregate signcryption scheme. We describe an attacker $\mathcal{B}$ against the CDH problem in Figure 11.

$\mathcal{B}$ perfectly simulates the oracles for $\mathcal{A}$ unless the signcryption oracle fails (which causes $\mathcal{B}$ to output $\perp$ and halt). This will only occur if H_2 is defined on a query involving $T \xleftarrow{\$} \mathbb{G}$. This occurs with probability at most $q_S(q_S + q_2)/q$. If the simulation is correct and $\mathcal{A}$ outputs a valid forgery, then $\mathcal{B}$ may output $\perp$ if $H_1(T^{(i)}, c^{(i)}, ID_S^{(i)})$ or $H_2(T^{(i)}, c^{(i)}, ID_R^*)$ is not defined. However, in this case, the probability that C is a valid forgery is at most $1/q$ since with probability $1 - 1/q$ we will have that σ is incorrect.

If these two events do not occur, and $\mathcal{B}$ outputs a valid forgery, then there must exist j' such that $pk_S^{(j')} = y_S^*$ and $(y_R^*, m^{(j')})$ was never queried to the unsigncryption oracle. Furthermore, we must have that $H_2(T^{(i)}, c^{(i)}, ID_R^*)$ is defined. This cannot have been defined through a signcryption query, as otherwise we would have that $\mathcal{B}$ has not output a valid forgery. Hence, we have that $(T^{(j')}, c^{(j')}, ID_R^*)$ was the j-th query to the H_2 oracle with probability at least $1/q_2$. If this is the case, then $\mathcal{B}$ solves the CDH problem. Therefore we have that

$$Adv_{\mathcal{A}}^{\mathtt{UF}}(k) \leq q_2 Adv_{\mathcal{B}}^{\mathtt{CDH}}(k) + \frac{q_S(q_S + q_2) + 1}{q}\,.$$

This proves that the aggregate signcryption scheme is unforgeable. □

6 Signcryption in DDFD Systems

In this section, we investigate the efficiency of aggregate signcryption schemes in network systems using a database system as an example. Obviously, aggregate signcryption has the potential to be useful for securing data that needs to be transmitted with data confidentiality and data integrity through a connectionless communication channel. We give a comparison of the bandwidth and efficiency of a number of aggregate signcryption ciphertexts in Table 1 and Table 2 (including

```
$\mathcal{B}(Y^*, U^*)$:
  $y_S^* \leftarrow Y^*$
  $j \xleftarrow{\$} \{1, \dots, q_2\}$
  $(\boldsymbol{y_S}, y_R^*, C^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_S}(y_S^*)$
    If $\mathcal{A}$ queries $\mathcal{O}_U(y_R, m)$
      $t \xleftarrow{\$} \mathbb{Z}_q$; $T \leftarrow g^t$
      $K \leftarrow H_K(e(y_R, H_t(T, ID_S))^t)$
      $c \leftarrow \texttt{Enc}_K(m)$
      If $H_2(T, c, ID_R)$ is defined
        Output $\perp$ and halt
      $\beta \xleftarrow{\$} \mathbb{Z}_q$
      Add $(T, c, ID_R, \perp, \beta)$ to H$_2$-List
      $\sigma \leftarrow H_1(T, c, ID_S^*)^t y_S^{*\beta}$
      Return $(T, c, \sigma)$
    If $\mathcal{A}$ queries $H_1(T, c, ID)$
      $\alpha \xleftarrow{\$} \mathbb{Z}_q$
      Add $(T, c, ID, \alpha)$ to H$_1$-List
      Return $g^\alpha$
    If $\mathcal{A}$ queries $H_2(T, c, ID)$
      If this is not the $j$-th query
        $\beta \xleftarrow{\$} \mathbb{Z}_q$
        Add $(T, c, ID, \beta)$ to H$_2$-List
        Return $g^\beta$
      Else
        Return $U^*$
    If $\mathcal{A}$ queries $H_t(T, ID)$
      $\gamma \xleftarrow{\$} \mathbb{Z}_q$
      Add $(T, ID, \gamma)$ to H$_t$-List
      Return $g^\gamma$
(continued in next column)

  Parse $C$ as $(\boldsymbol{T}, \boldsymbol{c}, \sigma)$
  If $T^{(i)} = T^{(j)}$ for some $i \neq j$ then output $\perp$ and halt
  If $\exists i$ such that $H_1(T^{(i)}, c^{(i)}, ID_S^{(i)})$
    or $H_2(T^{(i)}, c^{(i)}, ID_R^*)$ is not defined
    then output $\perp$ and halt
  $P_1 \leftarrow \prod_{i=1}^n e(T^{(i)}, H_1(T^{(i)}, c^{(i)}, ID_S^{(i)}))$
  $P_2 \leftarrow \prod_{i=1}^n e(y_S^{(i)}, H_2(T^{(i)}, c^{(i)}, ID_R^*))$
  If $e(g, \sigma) \neq P_1 P_2$ then output $\perp$
  For $i = 1, \dots, n$
    Compute $H_t(T^{(i)}, ID_S^{(i)})$
    Find $(T^{(i)}, ID_S^{(i)}, \gamma) \in$ H$_t$-List
    $K \leftarrow H_K(e(T^{(i)}, y_R^*)^\gamma)$
    $m^{(i)} \leftarrow \texttt{Dec}_K(c^{(i)})$
    If $m^{(i)} = \perp$ then output $\perp$ and halt
  Find $j'$ such that $pk_S^{(j')} = y_S^*$ and $(y_R^*, m^{(j')})$
    was not queried to $\mathcal{O}_S$. If no such query exists
    then output $\perp$
  If $(T^{(j')}, c^{(j')}, ID_R^*)$ not the $j$-th query to $H_2$ oracle
    Output $\perp$ and halt
  For $i = 1, \dots, n$
    Find $(T^{(i)}, c^{(i)}, ID_S^{(i)}, \alpha_i) \in$ H$_1$-List
  For $i = 1, \dots, n$ and $i \neq j$
    Find $(T^{(i)}, c^{(i)}, ID_R^*), \beta_i) \in$ H$_2$-List
  $Z_1 \leftarrow \prod_{i=1}^n T^{(i)\alpha_i}$
  $Z_2 \leftarrow \prod_{i=1, i \neq j}^n y_S^{(i)\beta_i}$
  Output $\sigma / Z_1 Z_2$
```

Fig. 11. The CDH attacker for the SKI-UF-CMA unforgeability of the aggregate signcryption scheme. We assume that the H_K oracle is simulated in the obvious way (by returning consistent random values). We also assume that the H_1, H_2 and H_t oracles are consistent when queried with the same inputs twice even if the hash function's response has been defined by another oracle.

Table 1. Comparison of Bandwidth Use by Aggregate Signcryption Solutions

Name	**Confidentiality**	**Unforgeability**	**1-overhead**	**n-overhead**	**DoD**
ECIES + Schnorr	insider IND-RCCA	insider UF-SKI-CMA	$4q + \|MAC\|$	$4nq + n\|MAC\|$	no
Zheng	outsider IND-CCA2	insider UF-SKI-CMA	$2q$ bits	$2nq$ bits	no
LQ1	insider IND-CCA2	insider UF-CMA	$4q$ bits	$4nq$ bits	no
LQ2	insider IND-CCA2	insider UF-CMA	$3q + 1$ bits	$3nq + n$ bits	no
ECIES + BGLS	insider IND-RCCA	insider UF-SKI-CMA	$4q + \|MAC\|$	$2nq + n\|MAC\| + 2q$	no
Dent	insider IND-SKI-CCA2	insider UF-SKI-CMA	$4q$	$2q(n + 1)$	yes

The term "n-overhead" refers to the amount of extra data that needs to be transmitted in an aggregate of n signcryption ciphertexts. In other words, it is the length of the aggregate signcryption ciphertext *minus* the lengths of the individual messages. The first "block" of the table consists of signcryption schemes which "aggregate" their ciphertexts simply by presenting all the original ciphertexts as a sequence. The second "block" of the table consists of "pure" aggregate signcryption schemes. For Diffie-Hellman-based schemes we assume the scheme is implemented on a group $\mathbb{G}$ of order q in which group elements can be represented using $2q$ bits (as is the case for elliptic curve groups). A typical value for q is 160-bits. We assume that IND-CPA symmetric encryption does not incur any overhead. We assume a MAC algorithm that produces MAC tags of size $|MAC|$ (e.g. $|MAC| = 80$). The ECIES+Schnorr scheme is the trivial signcryption scheme formed by combining the ECIES encryption scheme [1] and the Schnorr signature scheme [27]. Zheng's signcryption scheme is given in [30]. The LQ1 and LQ2 signcryption schemes were given by Libert and Quisquater and are described in [21] and [22] respectively. The ECIES+BGLS scheme is the aggregate signcryption scheme formed by combining the ECIES encryption scheme [1] and the BGLS aggregate signature scheme [11] using the method in Section 4. The Dent aggregate signcryption scheme is the direct construction given in Section 5. All schemes are proven secure in the random oracle model.

Table 2. Comparison of Efficiency of Aggregate Signcryption Schemes

Name	**Signcryption (Operations)**	**(n, m)-Aggregation (Operations)**	**n-Unsigncryption (Operations)**
ECIES + Schnorr	3 exp	trivial	$3n$ exp*
Zheng	1 exp	trivial	3n exp*
LQ1	3 exp	trivial	n exp/$2n$ pair
LQ2	2 exp	trivial	$2n$ exp/$2n$ pair
ECIES + BGLS	3 exp	†	n exp/$n + 1$ pair
Dent	4 exp/1 pair	$3n$ pair	n exp/$n + 3$ pair

The efficiency of the same schemes as in Table 1 expressed as the number of exponentiations and pairing operations required (since these are the dominant operations). The (n, m)-aggregation cost refers to the cost of aggregating n signcryption ciphertexts that already contain m messages. The n-unsigncryption cost refers to the cost of decryption of a signcryption ciphertext containing n messages. * serves to remind the reader that these schemes do not need to be implemented on pairing friendly elliptic curves and so may be faster than the equivalent operation for other schemes. † denotes that the operation requires either a number of multiplications or $2n$ pairings, depending on whether the intermediate nodes verify the BGLS signatures before passing on the aggregate ciphertext or not. If the scheme verifies the BGLS signatures, then a weak form of DoD security can be achieved.

a comparison with a number of traditional signcryption schemes where "aggregation" occurs by trivially combining all the ciphertexts into one sequence).

It is interesting to note that the optimal choice of aggregate signcryption scheme depends on the security needs of the application. If the application only requires outsider confidentiality, then a naïve use of Zheng's signcryption scheme is the most efficient (where "aggregation" occurs by concatenating the individual ciphertexts without any attempt to combine them). If insider confidentiality is required and the majority of communication is not likely to require aggregation, then the LQ2 scheme (with naïve aggregation) is most efficient. However, the Dent scheme of Section 5 becomes more efficient in any situation where two or more ciphertexts need to be aggregated.

To illustrate this point more fully, we give an example of an implementation of aggregate signcryption securing communication in a dynamic, distributed, federated database (DDFD) system. We use the Gaian database [6] as our example as this system requires a large number of secure connectionless communications.

6.1 The Gaian Database System

The Gaian database is a distributed database in which data is held by nodes in a "logical graph" [6]. The system is federated in the sense that it uses a store-local-query-anywhere (SLQA) structure which allows requests for data not held in the local database to be obtained from remote nodes. The physical network on which the database graph is implemented is assumed to be dynamic and lacking any pre-specified structure. Since the structure of the logical and physical graphs cannot be assumed, a database node in the logical graph performs a query operation by flooding the network with requests for data and collating the results. All database nodes are required to respond to requests for information even if the database does not hold any relevant data.

The novel approach of the Gaian database system is in how the logical graph "grows" with the addition of new nodes. The system uses a preferential attachment system whereby a new node is linked to m existing nodes in the graph through the use of a preferential attachment rule that is designed to give rise to a logical graph that shares similar properties to a scale-free graph [8]; most notably the small diameter that is associated with scale free graphs. A small diameter implies that the "query flood" operation is relatively efficient. The basic operation of the preferential attachment system means that a new database node u links to the first m nodes that return a "connection message". Each existing node v with degree less than v_{max} sends a connection message after a delay of t seconds where t is randomly chosen in the range $[0, t_0 F(v)]$, t_0 is an implementation-defined constant and F is the "fitness function". Proposed fitness functions include $F(v) = 1/deg(v)$ [6] and $F(v) = d(u,v)/deg(v)$ where $d(u,v)$ is the distance (in hops) between u and v in the *physical* network [12]. The constant v_{max} is meant to stop the logical graph containing nodes which are "too critical" and which could be the target for denial of service attacks.

6.2 Security Requirements

The Gaian system is designed to be used in situations which include data retrieval by palm-top devices through wireless communication in active military conflict scenarios; hence, communication between nodes may be subject to eavesdropping and data manipulation attacks[4]. Furthermore, one must consider the possibility that these devices will fall into malicious hands, which corresponds to the existence of malicious nodes in the logical graph. With regards to efficiency, it is generally accepted that processing power and storage capacity is increasing faster than communications capacity in these scenarios and that communication solutions should concentrate on achieving high bandwidth efficiency [19].

We believe that aggregate signcryption is a natural candidate for fulfilling these security requirements:

[4] Jamming attacks are also a distinct possibility; however, the provision of an availability service is not the purpose of this paper and so we will ignore this threat.

- **Structure-free**: The aggregate signcryption scheme only assumes that nodes have asymmetric key-pairs and authentic copies of the public keys of other nodes. This can easily be guaranteed at the point of manufacture or service. No implicit network structure has to be assumed.
- **Confidentiality**: Aggregate signcryption provides confidential message transmission. Owing to the possibility of device compromise and the strong confidentiality requirements that may exist for the data, forward secrecy is an important requirement for the signcryption scheme and hence the use of an insider-secure IND-CCA2 signcryption scheme is recommended.
- **Integrity Protection/Origin Authentication**: Aggregate signcryption provides data origin authentication and integrity protection. Again, owing to the possibility of device compromise, insider UF-CMA security would be recommended. Of course, aggregate signcryption does not automatically provide freshness guarantees; however, these could be built into the query request/response protocol through the use of nonces – see (e.g.) key establishment protocols from signcryption schemes [14].
- **DoD Security**: While not explicitly a requirement for the communication system, DoD security may provide a useful service. The DoD security guarantee ensures that a compromised node cannot create a signcryption ciphertext that will be correctly aggregated, but will corrupt the data provided by other nodes. In other words, compromising a node allows an attacker to send malicious data from that node, but does not give the attacker the ability to prevent the legitimate data from other nodes from being received[5].

Moreover, aggregate signcryption aims to provide these services whilst consuming the minimum amount of bandwidth.

6.3 Efficiency Comparison

Aggregate signcryption has the potential to improve bandwidth efficiency for nodes who are responding to “flood” requests for data. The requests themselves cannot be compressed as they are intended for different recipients. The most obvious candidates for securing the messaging function in the Gaian database are the LQ2 signcryption scheme and the Dent aggregate signcryption scheme. (Zheng’s signcryption scheme only provides outsider security and so does not meet the security requirements of the application.)

We modelled the communication requirements for the data responses using a computer simulation. The simulation generated 1000 Gaian logical networks[6] each consisting of n nodes and computed the communication overhead for each node to receive data from every other nodes in the network using each of the different aggregate signcryption schemes in Table 1. Nodes were numerically labelled in the order in which they joined the network and if two equal-length paths existed to transmit the data, then the data was passed to nodes with lower node labels in preference to nodes with higher node labels. Each new node formed $m = 2$ attachments using the fitness function $F(u) = 1/\deg(u)$ and no “cap” v_{max} was placed on the number of attachments a node could make.

The average communication overhead for each communication type is shown in Figure 12 with number of nodes along the x-axis and communication overhead in bits along the y-axis. With the exception of the Zheng signcryption scheme (which does not have sufficient security for this application) the direct construction of Section 5 achieves clear bandwidth savings of up to 43% against a naive combination of ECIES encryption and Schnorr signatures and of 14% against the next best scheme (the LQ2 algorithm).

Although it is not obvious from the graph, there is an interesting interplay between efficiency of the Dent construction and the LQ2 scheme. For networks with fewer nodes, the LQ2 scheme is more efficient. The new scheme becomes more efficient in networks with more than 60 nodes. This is because networks with fewer nodes are likely to form simple networks with lots of nodes that

[5] This guarantee, of course, only applies at an algorithmic level. The compromised node may still prevent legitimate data from being received by, for example, blocking the communication channel.

[6] Since the logical network must be embedded in the physical network, it serves as a “minimum” description of the network over which data will be passed.

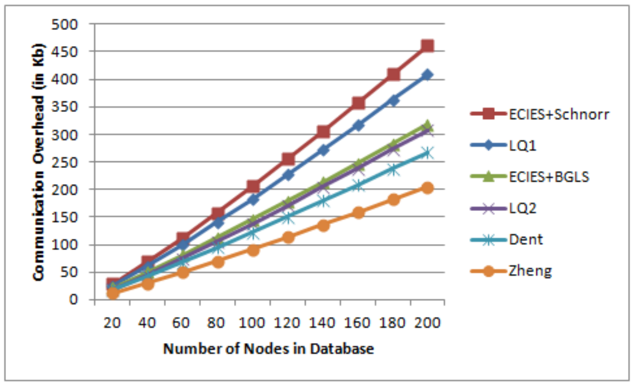

Fig. 12. Communication Overhead of Signcryption Schemes in the Gaian Database

are only one edge away from each other. This means that there will be a lot of ciphertexts which only contain a single message and the LQ2 scheme is more efficient in this situation.

This also explains the LQ2 scheme is more bandwidth efficient than the new scheme when considering the overhead required to send data to the “best” node in the graph (i.e. the node that requires the least overhead). This node is likely to be central in the graph and the graph is likely to resemble a “star” network. The LQ2 scheme is likely to be more efficient in this case. In contrast, the new scheme is significantly more efficient when we consider the overhead required to send data to the “worst” node in the graph (i.e. the node that requires the most overhead).

Lastly, we recall that we have only considered the bandwidth cost for a node to *receive* data. It may be thought that the advantages of using the new scheme disappear when considering the need to both send requests for data and receive responses, since no aggregation can occur when sending requests for data. However, we note that bandwidth savings may still be achieved and (as a worst-case solution) we may use different algorithms to send requests and to receive responses. For example, we may use LQ2 to send requests for data and the new scheme to receive responses. The “code cost” of using two algorithms may be reasonable if the bandwidth saving is sufficiently large.

7 Acknowledgements

I would like to acknowledge and thank Greg Neven for some initial discussions about the feasibility of aggregate signcryption. This paper has been the product of lethargic work for a period of at least four years and I would like to acknowledge the various institutions with whom I have worked during this time. I would like to thank the Information Security Group at Royal Holloway for giving my a position in the department for ten years and I would like to thank the Computer Science Department at City University of New York for allowing me to visit on numerous occasions.

References

1. M. Abdalla, M. Bellare, and P. Rogaway. The oracle Diffie-Hellman assumption and an analysis of DHIES. In D. Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 143–158. Springer-Verlag, 2001.
2. J. H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In L. Knudsen, editor, *Advance 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. *Journal of Cryptology*, 20(2):203–235, 2007.
4. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *Proceedings of the 38th Symposium on Foundations of Computer Science*, IEEE, 1997.

5. M. Bellare and A. Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In M. Franklin, editor, *Advances in Cryptology – Crypto 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 273–289. Springer-Verlag, 2004.
6. G. Bent, P. Dantressangle, D. Vyvyan, A. Mowshowitz, and V. Mitsou. A dynamic distributed federtaed database. In *Proc. Second Annual Conference of the ITA*, 2008.
7. A. Boldyreva, C. Gentry, A. O'Neill, and D. H. Yum. Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. In *ACM Conference on Computer and Communications Security – ACM CCS '07*, pages 276–285. ACM Press, 2007.
8. B. Bollobas and O. Riordan. The diameter of a scale free random graph. *Combinatorika*, 24(1):5–34, 2004.
9. D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 36(5):1301–1328, 2007.
10. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *Advances in Cryptology – Crypto 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer-Verlag, 2001.
11. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In E. Biham, editor, *Advances in Cryptology – Eurocrypt 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 416–432. Springer-Verlag, 2003.
12. E. Bulut and B. K. Szymanski. On alignment of physical and logical network in GaianDB network. Rensselaer Polytechnic Institute, Dept. of Computer Science, Report CS-11-10, 2010.
13. R. Canetti, H. Krawczyk, and J. B. Nielsen. Relaxing chosen-ciphertext security. In D. Boneh, editor, *Advances in Cryptology – Crypto 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 565–582. Springer-Verlag, 2003.
14. A. W. Dent and Y. Zheng, editors. *Practical Signcryption*. Springer-Verlag, 2010.
15. W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22:644–654, 1976.
16. O. Eikemeier, M. Fischlin, J.-F. Götzmann, A. Lehmann, D. Schröder, P. Schröder, and D. Wagner. History-free aggregate message authentication codes. In J. Garay and R. De Prisco, editors, *Security in Communication Networks – SCN 2010*, volume 6280 of *Lecture Notes in Computer Science*, pages 309–328. Springer-Verlag, 2010.
17. U. Feige, D. Lapidot, and A. Shamir. Multiple noninteractive zero knowledge proofs under general assumptions. *SAIM Journal on Computing*, 29(1):1–28, 1999.
18. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Science*, 28:270–299, 1984.
19. International Technical Alliance (ITA). BPP11 Technical Areas. 2010.
20. E. Kiltz. Chosen-ciphertext security from tag-based encryption. In S. Halevi and T. Rabin, editors, *Theory of Cryptography – TCC 2006*, volume 3876 of *Lecture Notes in Computer Science*, pages 581–600. Springer-Verlag, 2006.
21. B. Libert and J.-J. Quisquater. Efficient signcryption with key privacy from gap Diffie-Hellman groups. In F. Bao, editor, *Public Key Cryptography – PKC 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 187–200. Springer-Verlag, 2004.
22. B. Libert and J.-J. Quisquater. Improved signcryption from q-Diffe-Hellman problems. In C. Blundo and S. Cimato, editors, *Security in Communication Networks – SCN 2004*, volume 3352 of *Lecture Notes in Computer Science*, pages 220–234. Springer-Verlag, 2004.
23. J. K. Liu, M. H. Au, and W. Susilo. Self-generated-certificate public key cryptography and certificateless signature/encryption scheme in the standard model. In *Proc. ACM Symposium on Information, Computer and Communications Security*. ACM Press, 2007.
24. H. Lu and Q. Xie. An efficient certificaless aggregate signcryption scheme. In *International Conference on Electronics, Communications and Control*, pages 132–135. IEEE Press, 2011.
25. M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proc. 22nd ACM Symposium on the Theory of Computing – STOC '90*, pages 427–437. ACM Press, 1990.
26. C. Rackoff and D. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In J. Feigenbaum, editor, *Advances in Cryptology – Crypto '91*, volume 576 of *Lecture Notes in Computer Science*, pages 434–444. Springer-Verlag, 1991.
27. C. P. Schnorr. Efficient signature generation for smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.
28. S. Sharmilla Deva Selvi, S. Sree Vivek, J. Shiriam, S. Kalaivani, and C. Pandu Rangan. Identity based aggregate signcryption schemes. In B. Roy and N. Sendrier, editors, *Progress in Cryptology –*

Indocrypt 2009, volume 5922 of *Lecture Notes in Computer Science*, pages 378–397. Springer-Verlag, 2009.
29. V. Shoup. Sequences of games: A tool for taming complexity in security proofs. Available from `http://eprint.iacr.org/2004/332/`, 2004.
30. Y. Zheng. Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost(signature) + cost(encryption). In B. Kaliski, editor, *Advances in Cryptology – Crypto '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 165–179. Springer-Verlag, 1997.

A Commit-then-Encrypt-and-Sign

If (ComSetup, Commit, Open) is a commitment scheme, (SigGen, Sign, Aggregate, Ver) is an aggregate signature scheme, and (EncGen, Enc, Dec) is a public-key encryption scheme, then we may construct a $\mathcal{C}t\mathcal{E}\&\mathcal{S}$ aggregate signcryption scheme as in Figure 13. This scheme is attractive since the encryption and signature algorithms can be run in parallel in both signcryption and unsigncryption.

```
Setup($1^k$):
  $param \xleftarrow{\$} \texttt{ComSetup}(1^k)$
  Output $param$

Gen$_S$($1^k$):
  $(pk_S, sk_S) \xleftarrow{\$} \texttt{SigGen}(1^k)$
  Output $(pk_S, sk_S)$

Gen$_R$($1^k$):
  $(pk_R, sk_R) \xleftarrow{\$} \texttt{EncGen}(1^k)$
  Output $(pk_R, sk_R)$

Aggregate($\boldsymbol{pk_S}, pk_R, \boldsymbol{C}$):
  Parse $\boldsymbol{C}$ as $((c^{(1)}, \chi^{(1)}, \sigma^{(1)}), \ldots, (c^{(n)}, \chi^{(n)}, \sigma^{(n)}))$
  $\sigma \xleftarrow{\$} \texttt{SigAgg}(\boldsymbol{pk_S}, \boldsymbol{\sigma})$
  Output $(\boldsymbol{c}, \boldsymbol{\chi}, \sigma)$

Signcrypt($sk_S, pk_R, m$):
  $(c, d) \xleftarrow{\$} \texttt{Commit}(param, m)$
  $m' \leftarrow (ID_S, d)$
  $\chi \leftarrow \texttt{Enc}(pk_R, m')$
  $m'' \leftarrow (ID_R, c)$
  $\sigma \xleftarrow{\$} \texttt{Sign}(sk_S, m'')$
  Output $C \leftarrow (c, \chi, \sigma)$

Unsigncrypt($\boldsymbol{pk_S}, sk_R, C$):
  Parse $C$ as $(\boldsymbol{c}, \boldsymbol{\chi}, \sigma)$
  $\boldsymbol{m''} \leftarrow ((ID_R, c^{(1)}), \ldots, (ID_R, c^{(n)}))$
  If $\texttt{Ver}(\boldsymbol{pk_S}, \boldsymbol{m''}, \sigma) = \perp$ then output $\perp$
  For $1 \leq i \leq n$:
    $m'^{(i)} \leftarrow \texttt{Dec}(sk_R, \chi^{(i)})$
    Parse $m'^{(i)}$ as $(ID_S^{(i)}, d^{(i)})$
    If $pk_S^{(i)}$ is not the public key of $ID_S^{(i)}$ then output $\perp$
    $m^{(i)} \leftarrow \texttt{Open}(param, c^{(i)}, d^{(i)})$
    If $m^{(i)} = \perp$ then output $\perp$
  Output $\boldsymbol{m}$
```

Fig. 13. The Commit-then-Encrypt-and-Sign Construction

Theorem 4. *The following results hold:*

1. *If $\mathcal{A}$ is a PPT attacker against the UF-CMA unforgeability of the signcryption scheme that makes at most q_s queries to the signcryption oracle, then there exists a PPT attacker $\mathcal{B}$ against the UF-CMA unforgeability of the signature scheme and a PPT attacker $\mathcal{B}'$ against the relaxed binding property of the commitment scheme such that $Adv^{\texttt{UF}}_{\mathcal{A}}(k) \leq Adv^{\texttt{UF}}_{\mathcal{B}}(k) + q_s Adv^{\texttt{Bind}}_{\mathcal{B}'}(k)$.*
2. *If $\mathcal{A}$ is a PPT attacker against the IND-RCCA security of the signcryption scheme and the set $SPK(k)$ is polynomial-time recognisable, then there exists a PPT attacker $\mathcal{B}$ against the IND-CCA2 security of the public-key encryption scheme and a PPT attacker $\mathcal{B}'$ against the hiding property of the commitment scheme such that $Adv^{\texttt{RCCA}}_{\mathcal{A}}(k) \leq 2Adv^{\texttt{CCA}}_{\mathcal{B}}(k) + Adv^{\texttt{Hide}}_{\mathcal{B}'}(k)$.*

Again, since we cannot determine whether an encryption χ is valid or not, this construction does not give DoD security.

B Security Proofs for Generic Constructions

B.1 Encrypt-then-Sign

Let us re-cap the definition of the $\mathcal{E}t\mathcal{S}$ construction. If ($\texttt{SigGen}, \texttt{Sign}, \texttt{SigAgg}, \texttt{Ver}$) is an aggregate signature scheme and ($\texttt{EncGen}, \texttt{Enc}, \texttt{Dec}$) is a public key encryption scheme, then we may construct an $\mathcal{E}t\mathcal{S}$ aggregate signcryption scheme is given below. This scheme does not make use of a $\texttt{Setup}$ algorithm.

```
$\texttt{Gen}_S(1^k)$:
  $(pk_S, sk_S) \xleftarrow{\$} \texttt{SigGen}(1^k)$
  Output $(pk_S, sk_S)$

$\texttt{Gen}_R(1^k)$:
  $(pk_R, sk_R) \xleftarrow{\$} \texttt{EncGen}(1^k)$
  Output $(pk_R, sk_R)$

$\texttt{Signcrypt}(sk_S, pk_R, m)$:
  $m' \leftarrow (ID_S, m)$
  $\chi \xleftarrow{\$} \texttt{Enc}(pk_R, m')$
  $m'' \leftarrow (ID_R, \chi)$
  $\sigma \xleftarrow{\$} \texttt{Sign}(sk_S, m'')$
  Output $C \leftarrow (\chi, \sigma)$
```

```
$\texttt{Aggregate}(pk_R, \boldsymbol{C})$:
  Parse $\boldsymbol{C}$ as $((\chi^{(1)}, \sigma^{(1)}), \ldots, (\chi^{(n)}, \sigma^{(n)}))$
  $\sigma \xleftarrow{\$} \texttt{SigAgg}(\boldsymbol{pk_S}, \boldsymbol{\sigma})$
  Output $(\boldsymbol{\chi}, \sigma)$

$\texttt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R, C)$:
  Parse $C$ as $(\boldsymbol{\chi}, \sigma)$
  $\boldsymbol{m''} \leftarrow ((ID_R, \chi^{(1)}), \ldots, (ID_R, \chi^{(n)}))$
  If $\texttt{Ver}(\boldsymbol{pk_S}, \boldsymbol{m''}, \sigma) = \perp$ then output $\perp$
  For $1 \le i \le n$:
    $m'^{(i)} \leftarrow \texttt{Dec}(sk_R, \chi^{(i)})$
    Parse $m'^{(i)}$ as $(ID_S^{(i)}, m^{(i)})$
    If $pk_S^{(i)}$ is not the public key of $ID_S^{(i)}$ then output $\perp$
  Output $\boldsymbol{m}$
```

Theorem 1. *The following results hold:*

1. *If $\mathcal{A}$ is a PPT attacker against the insider SKI-UF-CMA unforgeability of the signcryption scheme, then there exists a PPT attacker $\mathcal{B}$ against the UF-CMA unforgeability of the signature scheme such that $Adv_{\mathcal{A}}^{\texttt{UF}}(k) \le Adv_{\mathcal{B}}^{\texttt{UF}}(k)$.*
2. *If $\mathcal{A}$ is a PPT attacker against the IND-RCCA security of the signcryption scheme and the set $SKP(k)$ is polynomial-time recognisable, then there exists a PPT attacker $\mathcal{B}$ against the IND-RCCA security of the encryption scheme such that $Adv_{\mathcal{A}}^{\texttt{RCCA}}(k) \le Adv_{\mathcal{B}}^{\texttt{RCCA}}(k)$.*

Proof We begin with the unforgeability game and directly describe the attacker $\mathcal{B}$. We assume that ID_S^* is the identity of the entity with public key pk_S^* (and similar for ID_R^* and pk_R^*)). $\mathcal{B}$ proceeds as follows:

```
$\mathcal{B}^{\mathcal{O}'_S}(pk^*)$:                              ▷ $\mathcal{O}'_S$ is the aggregate signature oracle
  $pk_S^* \leftarrow pk^*$
  $(\boldsymbol{pk_S}, pk_R^*, C^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_S}(pk_S^*)$        ▷ $\mathcal{O}_S$ is the aggregate signcryption oracle
    If $\mathcal{A}$ queries $\mathcal{O}_S(pk_R, m)$
      $m' \leftarrow (ID_S^*, m)$
      $c \xleftarrow{\$} \texttt{Enc}(pk_R, m')$
      $m'' \leftarrow (ID_R, c)$
      Query $\sigma \leftarrow \mathcal{O}'_S(m'')$
      Return $(c, \sigma)$
  Parse $C^*$ as $(\boldsymbol{c}, \sigma^*)$
  $\boldsymbol{m^*} \leftarrow ((ID_R^*, c^{(1)}), \ldots, (ID_R^*, c^{(n)}))$
  Output $(\boldsymbol{pk_S}, \boldsymbol{m^*}, \sigma^*)$
```

If $\mathcal{A}$ wins the insider unforgeability game, then we have that (1) pk_R^* is a valid receiver key pair (with corresponding private key sk_R^*); (2) $\texttt{Ver}(\boldsymbol{pk_S}, \boldsymbol{m^*}, \sigma^*) = \top$; (3) $\texttt{Dec}(sk_R^*, c^{(i)}) = (ID_S^{(i)}, m^{(i)})$ for some $m^{(i)}$ and all $1 \le i \le n$; and (4) there exists i^* such that $pk_S^{(i^*)} = pk^*$ and $(pk_R^*, m^{(i^*)})$ was never queried to the signcryption oracle.

Since σ^* is a valid signature, $\mathcal{B}$ wins the unforgeability game provided that there exists i such that $pk_S^{(i)} = pk^*$ and the signature oracle was never queried on $(ID_R^*, c^{(i)})$. We claim that this is satisfied when $i = i^*$. It certainly true that $pk_S^{(i^*)} = pk^*$ and, since ID_R^* uniquely identifies the public key pk_R^*, we have that the only way that $\mathcal{B}$ would possibly query the signature oracle on $(ID_R^*, c^{(i^*)})$ is if $\mathcal{A}$ queried the signcryption oracle on $(pk_R^*, m^{(i^*)})$, which was disallowed. Hence, $\mathcal{B}$ breaks the signature scheme whenever $\mathcal{A}$ breaks the signcryption scheme.

For confidentiality, we again directly construct an attacker $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$. For simplicity we will suppress the state information variable ω and simply assume that all necessary information is passed between attackers. $\mathcal{B}$ runs as follows:

```
$\mathcal{B}_1^{\mathcal{O}_D}(pk^*)$:
  $pk_R^* \leftarrow pk^*$
  $(m_0, m_1, pk_S^*, sk_S^*) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_U}(pk_R^*)$
  $m_i' \leftarrow (ID_S^*, m_i)$ for $i \in \{0,1\}$
  Output $(m_0', m_1')$

$\mathcal{B}_2^{\mathcal{O}_D}(c^*)$:
  $m'' \leftarrow (ID_R^*, c^*)$
  $\sigma^* \xleftarrow{\$} \mathtt{Sign}(sk_S^*, m'')$
  $C^* \leftarrow (c^*, \sigma^*)$
  If $|m_0| \neq |m_1|$ then $C^* \leftarrow \perp$
  If $(pk_S^*, sk_S^*) \notin SPK(k)$ then $C^* \leftarrow \perp$
  $b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_U}(C^*)$
  Output $b'$
```

```
$\mathcal{O}_U(\boldsymbol{pk_S}, C)$:
  Parse $C$ as $(\boldsymbol{c}, \sigma)$
  $\boldsymbol{m''} \leftarrow ((ID_R, c^{(1)}), \ldots, (ID_R, c^{(n)}))$
  If $\mathtt{Ver}(\boldsymbol{pk_S}, \boldsymbol{m''}, \sigma) = \perp$ then return $\perp$
  For $1 \leq i \leq n$:
    Query $m'^{(i)} \leftarrow \mathcal{O}_D(c^{(i)})$
    If $m'^{(i)} = \mathtt{test}$
      If $pk_S^{(i)} = pk_S^*$ then return $\mathtt{test}$
      Else return $\perp$
    Parse $m'^{(i)}$ as $(ID_S^{(i)}, m^{(i)})$
    If $pk_S^{(i)}$ is not the public key of $ID_S^{(i)}$
      Return $\perp$
  Return $\boldsymbol{m}$
```

$\mathcal{B}$ wins if $\mathcal{A}$ correctly identifies the bit b and $\mathcal{B}$ correctly simulates all of the oracles to which $\mathcal{A}$ has access. The challenge signcryption ciphertext is clearly constructed correctly. The $\mathcal{O}_U$ oracle clearly returns the correct result in all situations except perhaps if the decryption oracle $\mathcal{O}_D$ returns test. In this situation, the decryption of $c^{(i)}$ is either (ID_S^*, m_0) or (ID_S^*, m_1). Thus, if $pk_S^{(i)} \neq pk_S^*$ then the $\mathcal{O}_U$ oracle correctly returns $\perp$. If $pk_S^{(i)} = pk_S^*$ then the signcryption algorithm would return $m^{(i)} \in \{m_0, m_1\}$ and so the $\mathcal{O}_U$ oracle correctly returns test. Hence, $Adv_{\mathcal{A}}^{\mathtt{RCCA}}(k) = Adv_{\mathcal{B}}^{\mathtt{RCCA}}(k)$. □

B.2 Commit-then-Encrypt-and-Sign

Let us recap the $\mathcal{C}t\mathcal{E}\&\mathcal{S}$ scheme. If $(\mathtt{ComSetup}, \mathtt{Commit}, \mathtt{Open})$ is a commitment scheme, $(\mathtt{SigGen}, \mathtt{Sign}, \mathtt{Aggregate}, \mathtt{Ver})$ is an aggregate signature scheme, and $(\mathtt{EncGen}, \mathtt{Enc}, \mathtt{Dec})$ is a public-key encryption scheme, then we may construct a $\mathcal{C}t\mathcal{E}\&\mathcal{S}$ aggregate signcryption scheme as below

```
$\mathtt{Setup}(1^k)$:
  $param \xleftarrow{\$} \mathtt{ComSetup}(1^k)$
  Output $param$

$\mathtt{Gen}_S(1^k)$:
  $(pk_S, sk_S) \xleftarrow{\$} \mathtt{SigGen}(1^k)$
  Output $(pk_S, sk_S)$

$\mathtt{Gen}_R(1^k)$:
  $(pk_R, sk_R) \xleftarrow{\$} \mathtt{EncGen}(1^k)$
  Output $(pk_R, sk_R)$

$\mathtt{Aggregate}(\boldsymbol{pk_S}, pk_R, \boldsymbol{C})$:
  Parse $\boldsymbol{C}$ as $((c^{(1)}, \chi^{(1)}, \sigma^{(1)}), \ldots, (c^{(n)}, \chi^{(n)}, \sigma^{(n)}))$
  $\sigma \xleftarrow{\$} \mathtt{SigAgg}(\boldsymbol{pk_S}, \boldsymbol{\sigma})$
  Output $(\boldsymbol{c}, \boldsymbol{\chi}, \sigma)$
```

```
$\mathtt{Signcrypt}(sk_S, pk_R, m)$:
  $(c, d) \xleftarrow{\$} \mathtt{Commit}(param, m)$
  $m' \leftarrow (ID_S, d)$
  $\chi \leftarrow \mathtt{Enc}(pk_R, m')$
  $m'' \leftarrow (ID_R, c)$
  $\sigma \xleftarrow{\$} \mathtt{Sign}(sk_S, m'')$
  Output $C \leftarrow (c, \chi, \sigma)$

$\mathtt{Unsigncrypt}(\boldsymbol{pk_S}, sk_R, C)$:
  Parse $C$ as $(\boldsymbol{c}, \boldsymbol{\chi}, \sigma)$
  $\boldsymbol{m''} \leftarrow ((ID_R, c^{(1)}), \ldots, (ID_R, c^{(n)}))$
  If $\mathtt{Ver}(\boldsymbol{pk_S}, \boldsymbol{m''}, \sigma) = \perp$ then output $\perp$
  For $1 \leq i \leq n$:
    $m'^{(i)} \leftarrow \mathtt{Dec}(sk_R, \chi^{(i)})$
    Parse $m'^{(i)}$ as $(ID_S^{(i)}, d^{(i)})$
    If $pk_S^{(i)}$ is not the public key of $ID_S^{(i)}$ then output $\perp$
    $m^{(i)} \leftarrow \mathtt{Open}(param, c^{(i)}, d^{(i)})$
    If $m^{(i)} = \perp$ then output $\perp$
  Output $\boldsymbol{m}$
```

Theorem 2. *The following results hold:*

1. *If $\mathcal{A}$ is a PPT attacker against the UF-CMA unforgeability of the signcryption scheme that makes at most q_s queries to the signcryption oracle, then there exists a PPT attacker $\mathcal{B}$ against the UF-CMA unforgeability of the signature scheme and a PPT attacker $\mathcal{B}'$ against the relaxed binding property of the commitment scheme such that $Adv^{\texttt{UF}}_{\mathcal{A}}(k) \leq Adv^{\texttt{UF}}_{\mathcal{B}}(k) + q_s Adv^{\texttt{Bind}}_{\mathcal{B}'}(k)$.*
2. *If $\mathcal{A}$ is a PPT attacker against the IND-RCCA security of the signcryption scheme and the set $SPK(k)$ is polynomial-time recognisable, then there exists a PPT attacker $\mathcal{B}$ against the IND-CCA2 security of the public-key encryption scheme and a PPT attacker $\mathcal{B}'$ against the hiding property of the commitment scheme such that $Adv^{\texttt{RCCA}}_{\mathcal{A}}(k) \leq 2Adv^{\texttt{CCA}}_{\mathcal{B}}(k) + Adv^{\texttt{Hide}}_{\mathcal{B}'}(k)$.*

Proof We will start with the integrity property. The principle is simple: if $\mathcal{A}$ produces a forged ciphertext, then it must either produce a signature on a new value c (in which case we can break the unforgeability of the signature scheme) or it must produce a new decommitment d (in which case we can break the binding property of the signature scheme). We describe the attackers $\mathcal{B}$ and $\mathcal{B}'$ directly. $\mathcal{B}$ is an attacker against the UF-CMA security of the signature scheme and runs as follows:

```
$\mathcal{B}^{\mathcal{O}'_S}(pk^*)$:                                    $\triangleright$ $\mathcal{O}'_S$ is the aggregate signature oracle
  $param \xleftarrow{\$} \texttt{Setup}(1^k)$
  $pk^*_S \leftarrow pk^*$
  $(\boldsymbol{pk_S}, pk^*_R, sk^*_R, C) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_S}(param, pk^*_S)$   $\triangleright$ $\mathcal{O}_S$ is the aggregate signcryption oracle
    If $\mathcal{A}$ queries $\mathcal{O}_S(pk_R, m)$
      $(c, d) \xleftarrow{\$} \texttt{Commit}(param, m)$
      $m' \leftarrow (ID^*_S, d)$; $m'' \leftarrow (ID_R, c)$
      $\chi \xleftarrow{\$} \texttt{Enc}(pk_R, m')$
      Query $\sigma \leftarrow \mathcal{O}'_S(m'')$
      Return $(\chi, c, \sigma)$
  Parse $C$ as $(\boldsymbol{\chi}, \boldsymbol{c}, \sigma^*)$
  $\boldsymbol{m^*} \leftarrow ((ID^*_R, c^{(1)}), \ldots, (ID^*_R, c^{(n)}))$
  Output $(\boldsymbol{pk_S}, \boldsymbol{m^*}, \sigma^*)$
```

$\mathcal{B}'$ is an attacker against the binding property of the commitment scheme. We assume that the algorithms pass sufficient state between their parts to run without requiring it to be explicitly written. $\mathcal{B}'$ runs as follows:

```
$\mathcal{B}'_1(param)$:
  $j \leftarrow 1$; $j^* \xleftarrow{\$} \{1, \ldots, q_s\}$
  $(pk^*_S, sk^*_S) \xleftarrow{\$} \texttt{SigGen}(1^k)$
  $(\boldsymbol{pk_S}, pk^*_R, sk^*_R, C^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_S}(param, pk^*_S)$
    If $\mathcal{A}$ queries $\mathcal{O}_S(pk_R, m)$
      If $j \neq j^*$
        $j = j + 1$
        $(c, d) \xleftarrow{\$} \texttt{Commit}(param, m)$
        $m' \leftarrow (ID^*_S, d)$; $m'' \leftarrow (ID_R, c)$
        $\chi \xleftarrow{\$} \texttt{Enc}(pk_R, m')$
        $\sigma \xleftarrow{\$} \texttt{Sign}(sk^*_S, m'')$
        Return $(\chi, c, \sigma)$
      Else if $j = j^*$
        Pause $\mathcal{A}$'s execution
        Output $m^* \leftarrow m$
```

```
$\mathcal{B}'_2(c^*, d^*)$
  $m' \leftarrow (ID^*_S, d^*)$; $m'' \leftarrow (ID_R, c^*)$
  $\chi \xleftarrow{\$} \texttt{Enc}(pk_R, m')$
  $\sigma \xleftarrow{\$} \texttt{Sign}(sk^*_S, m'')$
  $C \leftarrow (\chi, c^*, \sigma)$
  Resume $\mathcal{A}^{\mathcal{O}_S}(param, pk^*_S)$ by returning $C$
    If $\mathcal{A}$ queries $\mathcal{O}_S(pk_R, m)$
      $(c, d) \xleftarrow{\$} \texttt{Commit}(param, m)$
      $m' \leftarrow (ID^*_S, d)$; $m'' \leftarrow (ID_R, c)$
      $\chi \xleftarrow{\$} \texttt{Enc}(pk_R, m')$
      $\sigma \xleftarrow{\$} \texttt{Sign}(sk^*_S, m'')$
      Return $(\chi, c, \sigma)$
  Parse $C^*$ as $(\boldsymbol{\chi}, \boldsymbol{c}, \sigma)$
  For $1 \leq i \leq n$ such that $c^{(i)} = c^*$:
    $(ID_S, d^{(i)}) \xleftarrow{\$} \texttt{Dec}(sk^*_R, \chi^{(i)})$
    If $\texttt{Open}(c^*, d^*) \neq \texttt{Open}(c^*, d^{(i)}) \neq \perp$
      Output $d^{(i)}$
  Otherwise output $\perp$
```

If $\mathcal{A}$ wins the unforgeability game, then it must have output a tuple $(\boldsymbol{pk_S}, pk_R^*, sk_R^*, C^*)$ such that (1) (pk_R^*, sk_R^*) is a valid receiver key pair; (2) $\texttt{Ver}(\boldsymbol{pk_S}, \boldsymbol{m''}, \sigma^*) = \top$ where $\boldsymbol{m''} = ((ID_R^*, c^{*(1)}), \ldots, (ID_R^*, c^{*(n)}))$; (3) $(ID_S^{(i)}, d^{(i)}) \leftarrow \texttt{Dec}(sk_R^*, \chi^{(i)})$ where $ID_S^{(i)}$ is the identity associated with $pk_S^{(i)}$ for $1 \leq i \leq n$; (4) $m^{(i)} \leftarrow \texttt{Open}(c^{(i)}, d^{(i)})$ satisfies $m^{(i)} \neq \perp$ for $1 \leq i \leq n$; (5) there exists i^* such that $pk_S^{(i^*)} = pk^*$ and $\mathcal{A}$ never queries $\mathcal{O}_S(pk_R^*, m^{(i^*)})$.

Suppose $\mathcal{A}$ outputs a valid forgery $(\boldsymbol{pk_S}, C^*)$ for a receiver key pair (pk_R^*, sk_R^*) where $C^* = (\boldsymbol{\chi^*}, \boldsymbol{c^*}, \sigma)$. We define E to be the event that the signcryption oracle was queried on (pk_R^*, m) and returned a signcryption ciphertext oracle of the form $(\chi, c^{*(i^*)}, \sigma)$ for some m, χ and σ.

If E does not occur, then we claim that the aggregate signature $(\boldsymbol{pk_S}, \boldsymbol{m^*}, \sigma^*)$ which is output by $\mathcal{B}$ is a valid forgery. This is because the only reason that the $\mathcal{B}$ would query $(ID_R^*, c^{*(i^*)})$ to the signature oracle is if it had received a signcryption query on $(pk_R^*, m^{(i^*)})$ and such a query is forbidden. Hence, $\Pr[\text{Expt}_{\mathcal{A}}^{\texttt{UF}}(k) = 1 \wedge \neg E] = \Pr[\text{Expt}_{\mathcal{B}}^{\texttt{UF}}(k) = 1]$.

If E does occur then $m^{(i)}$ cannot be equal to the message m which was queried to the signcryption oracle (as such a query is forbidden). Let $(ID_S^*, d) \leftarrow \texttt{Dec}(sk_R^*, \chi)$ and $(ID_S^*, d^*) \leftarrow \texttt{Dec}(sk_R^*, \chi^{*(i)})$. Then we have that $\texttt{Open}(c^{*(i)}, d) = m \neq m^{(i)} = \texttt{Open}(c^{*(i)}, d^*)$. As long as $\mathcal{B}'$ correctly chooses the value j^* which corresponds to the signcryption oracle query which returns $c^{*(i^*)}$ then $\mathcal{B}'$ breaks the relaxed binding property of the commitment scheme. Hence, $\Pr[\text{Expt}_{\mathcal{A}}^{\texttt{UF}}(k) \wedge E] \leq q_s \Pr[\text{Expt}_{\mathcal{B}'}^{\texttt{bind}}(k) = 1]$. The stated result follows from combining these two results.

We now consider confidentiality and prove our result via a sequence of games. We parameterise a game G_i^b by the bit b and let S_i^b to be the event that the attacker outputs $b' = 1$. Let G_1^b be the game be the game defined by $\text{Expt}_{\mathcal{A}}^{\texttt{IND-b}}(k)$. Hence, $Adv_{\mathcal{A}}^{\texttt{IND}}(k) = |\Pr[S_1^1] - \Pr[S_1^0]|$. We assume that "starred" variables refer to those variables computed for the challenge ciphertext.

In Game G_2^b we change how the unsigncryption oracle works. If $\mathcal{A}_2$ queries the unsigncryption oracle on $(\boldsymbol{pk_S}, (\boldsymbol{\chi}, \boldsymbol{c}, \sigma))$, then the oracle responds as follows:

```
$\mathcal{O}_U(\boldsymbol{pk_S}, (\boldsymbol{\chi}, \boldsymbol{c}, \sigma))$:
  $test \leftarrow 0$
  $\boldsymbol{m''} \leftarrow ((ID_R^*, c^{(1)}), \ldots, (ID_R^*, c^{(n)}))$
  If $\texttt{Ver}(\boldsymbol{pk_S}, \boldsymbol{m''}, \sigma) = \perp$ then return $\perp$
  For $1 \leq i \leq n$:
    If $\chi^{(i)} = \chi^*$
      If $(pk_S^{(i)}, c^{(i)}) \neq (pk_S^*, c^*)$ then return $\perp$
      Else $test \leftarrow 1$
    Else
      $m'^{(i)} \leftarrow \texttt{Dec}(sk_R^*, \chi^{(i)})$
      Parse $m'^{(i)}$ as $(ID_S^{(i)}, d^{(i)})$
      If $pk_S^{(i)}$ is not the public key of $ID_S^{(i)}$ then return $\perp$
      $m^{(i)} \leftarrow \texttt{Open}(c^{(i)}, d^{(i)})$
      If $m^{(i)} = \perp$ then return $\perp$
  If $test = 1$ then return $\texttt{test}$
  If $\exists i$ such that $pk_S^{(i)} = pk_S^*$ and $m^{(i)} \in \{m_0, m_1\}$ then return $\texttt{test}$
  Else return $\boldsymbol{m}$
```

We claim that the action of the unsigncryption oracle in G_2^b is identical to that in G_1^b. This is clearly true for any ciphertext which does not have a component $\chi^{(i)} = \chi^*$. If $\chi^{(i)} = \chi^*$ then $m'^{(i)} = (ID_S^*, c^*)$. Hence, the oracle should return $\perp$ if $pk_S^{(i)} = pk_S^*$ or if $c^{(i)} \neq c^*$ (since there is at most one commitment for every decommitment). However, if $(pk_S^{(i)}, c^{(i)}) = (pk_S^*, c^*)$ then $m^{(i)} = m_b$ and the oracle should return $\texttt{test}$ (assuming the other components correct decrypt). Thus, $\Pr[S_1^b] = \Pr[S_2^b]$.

In game G_3^b, the challenge signcryption ciphertext is computed using a random decommitment, as follows:

```
$(c^*, d) \xleftarrow{\$} \texttt{Commit}(m_b)$
$d^* \xleftarrow{\$} \{0,1\}^{|d|}$
$m'^* \leftarrow (ID_S^*, d^*)$
$\chi^* \xleftarrow{\$} \texttt{Enc}(pk_R^*, m'^*)$
$m''^* \leftarrow (ID_R^*, c^*)$
$\sigma^* \leftarrow \texttt{Sign}(sk_S^*, m''^*)$
$C^* \leftarrow (\chi^*, c^*, \sigma^*)$
```

We describe an attacker $\mathcal{B}$ against the IND-CCA2 security of the encryption scheme such that $|\Pr[S_3^b] - \Pr[S_2^b]| \leq Adv_{\mathcal{B}}^{\texttt{IND}}(k)$. $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ runs as follows:

```
$\mathcal{B}_1^{\mathcal{O}_D}(pk^*)$:
  $param \xleftarrow{\$} \texttt{ComSetup}(1^k)$
  $pk_R^* \leftarrow pk^*$
  $(m_0, m_1, pk_S^*, sk_S^*) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_U}(param, pk_R^*)$
  $(c^*, d_0^*) \xleftarrow{\$} \texttt{Commit}(param, m_b)$
  $d_1^* \xleftarrow{\$} \{0,1\}^{|d_0^*|}$
  $m_0'^* \leftarrow (ID_S^*, d_0^*)$; $m_1'^* \leftarrow (ID_S^*, d_1^*)$
  Output $(m_0'^*, m_1'^*)$

$\mathcal{B}_2^{\mathcal{O}_D}(\chi^*)$:
  $\sigma^* \xleftarrow{\$} \texttt{Sign}(sk_S^*, (ID_R^*, c^*))$
  $C^* \leftarrow (\chi^*, c^*, \sigma^*)$
  If $|m_0| \neq |m_1|$ then $C^* \leftarrow \perp$
  If $(pk_S^*, sk_S^*) \notin SPK(k)$ then $C^* \leftarrow \perp$
  $b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_U}(C^*)$
  Output $b'$
```

```
$\mathcal{O}_U(\boldsymbol{pk_S}, (\boldsymbol{\chi}, \boldsymbol{c}, \sigma))$:
  $test \leftarrow 0$
  $\boldsymbol{m''} \leftarrow ((ID_R^*, c^{(1)}), \ldots, (ID_R^*, c^{(n)}))$
  If $\texttt{Ver}(\boldsymbol{pk_S}, \boldsymbol{m''}, \sigma) = \perp$ then return $\perp$
  For $1 \leq i \leq n$:
    If $\chi^{(i)} = \chi^*$
      If $(pk_S^{(i)}, c^{(i)}) \neq (pk_S^*, c^*)$ then return $\perp$
      Else $test \leftarrow 1$
    Else
      Query $m'^{(i)} \leftarrow \mathcal{O}_D(\chi^{(i)})$
      Parse $m'^{(i)}$ as $(ID_S^{(i)}, d^{(i)})$
      If $pk_S^{(i)}$ is not the public key of $ID_S^{(i)}$ then return $\perp$
      $m^{(i)} \leftarrow \texttt{Open}(c^{(i)}, d^{(i)})$
      If $m^{(i)} = \perp$ then return $\perp$
  If $test = 1$ then return $\texttt{test}$
  If $\exists\, i$ such that $pk_S^{(i)} = pk_S^*$ and $m^{(i)} \in \{m_0, m_1\}$
    Return $\texttt{test}$
  Else return $\boldsymbol{m}$
```

We note that $\mathcal{B}$ correctly simulates the unsigncryption oracle as it will never query $\mathcal{O}_D(\chi^*)$. Hence,

$$Adv_{\mathcal{B}}^{\texttt{IND}}(k) = |\Pr[\text{Expt}_{\mathcal{A}}^{\texttt{IND-1}}(k) = 1] - \Pr[\text{Expt}_{\mathcal{A}}^{\texttt{IND-0}}(k) = 1]| = |\Pr[S_3^b] - \Pr[S_2^b]|\,.$$

Lastly, we show that there exists an attacker $\mathcal{B}'$ against the hiding property of the commitment scheme such that $|\Pr[S_3^1] - \Pr[S_3^0]| \leq Adv_{\mathcal{B}'}^{\texttt{hide}}(k)$.

```
$\mathcal{B}_1'(param)$:
  $(pk_R^*, sk_R^*) \xleftarrow{\$} \texttt{EncGen}(1^k)$
  $(m_0, m_1, pk_S^*, sk_S^*) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_U}(param, pk_R^*)$
  Output $(m_0, m_1)$

$\mathcal{B}_2'(c^*)$
  If $c^* = \perp$ or $(pk_S^*, sk_S^*) \notin SPK(k)$
    $C^* \leftarrow \perp$
  Else
    $(c', d') \leftarrow \texttt{Commit}(m_0)$
    $d^* \xleftarrow{\$} \{0,1\}^{|d'|}$
    $m'^* \leftarrow (ID_S^*, d^*)$
    $\chi^* \xleftarrow{\$} \texttt{Enc}(pk_R^*, m'^*)$
    $m''^* \leftarrow (ID_R^*, c^*)$
    $\sigma^* \xleftarrow{\$} \texttt{Sign}(sk_S^*, m''^*)$
    $C^* \leftarrow (\chi^*, c^*, \sigma^*)$
  $b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_U}(C^*)$
  Output $b'$
```

```
$\mathcal{O}_U(\boldsymbol{pk_S}, (\boldsymbol{\chi}, \boldsymbol{c}, \sigma))$:
  $test \leftarrow 0$
  $\boldsymbol{m''} \leftarrow ((ID_R^*, c^{(1)}), \ldots, (ID_R^*, c^{(n)}))$
  If $\texttt{Ver}(\boldsymbol{pk_S}, \boldsymbol{m''}, \sigma) = \perp$ then return $\perp$
  For $1 \leq i \leq n$:
    If $\chi^{(i)} = \chi^*$
      If $(pk_S^{(i)}, c^{(i)}) \neq (pk_S^*, c^*)$ then return $\perp$
      Else $test \leftarrow 1$
    Else
      $m'^{(i)} \leftarrow \texttt{Dec}(sk_R^*, \chi^{(i)})$
      Parse $m'^{(i)}$ as $(ID_S^{(i)}, d^{(i)})$
      If $pk_S^{(i)}$ is not the public key of $ID_S^{(i)}$ then return $\perp$
      $m^{(i)} \leftarrow \texttt{Open}(c^{(i)}, d^{(i)})$
      If $m^{(i)} = \perp$ then return $\perp$
  If $test = 1$ then return $\texttt{test}$
  If $\exists\, i$ such that $pk_S^{(i)} = pk_S^*$ and $m^{(i)} \in \{m_0, m_1\}$
    Return $\texttt{test}$
  Else return $\boldsymbol{m}$
```

Again, the unsigncryption oracle simulation is perfect. Furthermore,

$$Adv_{\mathcal{B}'}^{\texttt{hide}}(k) = |\Pr[\textsc{Expt}_{\mathcal{B}'}^{\texttt{hid-1}}(k) = 1] - \Pr[\textsc{Expt}_{\mathcal{B}'}^{\texttt{hid-0}}(k) = 0]| = |\Pr[S_3^1] - \Pr[S_3^0]|\,.$$

Combining inequalities gives the stated result. $\square$