Improved Identity-Based Signcryption

Liqun Chen[1] and John Malone-Lee[2]

[1] Hewlett-Packard Laboratories, Filton Road,
Stoke Gifford, Bristol, BS34 8QZ, UK.
liqun.chen@hp.com
[2] University of Bristol, Department of Computer Science,
Woodland Road, Brsitol, BS8 1UB, UK.
malone@cs.bris.ac.uk

Abstract. Identity-based cryptography is form of public-key cryptography that does not require users to pre-compute key pairs and obtain certificates for their public keys. Instead, public keys can be arbitrary identifiers such as email addresses. This means that the corresponding private keys are derived, at any time, by a trusted private key generator.

The idea of signcryption is to provide a method to encrypt and sign data together in a way that is more efficient than using an encryption scheme combined with a signature scheme.

We present an identity-based signcryption solution that we believe is the most efficient, provably-secure scheme of its type proposed to date. Our scheme admits proofs of security in the random oracle model under the bilinear Diffie-Hellman assumption using the definitions proposed by Boyen.

1 Introduction

Two of the most important services offered by cryptography are those of providing private and authenticated communications. Much research has been done into creating encryption schemes to meet highly developed notions of privacy [3, 16]. Similarly, designing unforgeable signature schemes to give authenticity and non-repudiation is also a well studied problem [10]. It is possible to combine encryption schemes and signature schemes, using methods such as those described in [1], to obtain private and authenticated communications.

In 1997, Zheng proposed a primitive that he called *signcryption* [20]. The idea of a signcryption scheme is to combine the functionality of an encryption scheme with that of a signature scheme. It must provide privacy; signcryptions must be unforgeable; and there must be a method to settle repudiation disputes. This must be done in a more efficient manner than a composition of an encryption scheme with a signature scheme. Along with the concept, Zheng also proposed an efficient, discrete logarithm based scheme.

The first formal security treatment for signcryption appeared in [1]. This work formalised notions of privacy and unforgeability. Subsequently, several provably secure signcryption schemes have been designed, for example [12].

The concept *identity-based cryptography* was proposed by Shamir in 1984 [18]. The idea of an identity-based system is that public keys can be derived from arbitrary strings. This means that if a user has a string corresponding to its identity, this string can be used to derive the user's public key. For this to work there is a *trusted authority* (TA henceforth) that generates private keys using some master key related to the global parameters for the system. In [18] Shamir proposed an identity-based signature scheme, but for many years identity-based encryption remained an open problem. The problem was solved nearly two decades after it was originally proposed [5, 9]. In [9] Cocks proposed a solution based on quadratic residuosity and in [5] Boneh and Franklin gave a scheme using bilinear pairings on elliptic curves. It is pairings on elliptic curves that have become the most popular building block for identity-based cryptography and many schemes have been designed using this primitive.

The idea of *identity-based signcryption* was first proposed by Malone-Lee in [13] along with a security model. This model dealt with notions of privacy and unforgeability. A weakness in the scheme from [13] was subsequently pointed out by Libert and Quisquater in [11] where a new scheme was proposed. The new scheme came with proofs of security in the model of [13]. This model was developed by Boyen in [6]. Three new security notions were added: *ciphertext unlinkability*, *ciphertext authentication* and *ciphertext anonymity*. We discuss these notions in Section 3. Boyen also proposed a scheme in [6] and analysed it in the enhanced model.

We take the model from [6] as the starting point for this work. We describe a scheme that admits security proofs in this model. We show that our scheme compares favourably with other provably-secure signcryption schemes in the literature.

The paper proceeds as follows. In Section 2 we formally define what we mean by identity-based signcryption. Section 3 recalls the security model from [6]. We present our scheme in Section 4 and provide security results for it in Section 5. A comparison is made with existing schemes in Section 6. The paper ends with some concluding remarks.

2 Identity-Based Signcryption

Before formally defining what we mean by identity-based signcryption we describe the notation that we will use throughout the paper.

Notation Let S be a set. We write $v \leftarrow S$ to denote the action of sampling from the uniform distribution on S and assigning the result to the variable v. If S contains one element s we use $v \leftarrow s$ as shorthand for $v \leftarrow \{s\}$. If $\mathcal{A}$ is an algorithm we denote the action of running $\mathcal{A}$ on input I and assigning the resulting output to the variable v by $v \leftarrow \mathcal{A}(I)$.

If E is an event defined in some probability space, we denote the probability that E occurs by $\mathbf{Pr}[E]$ (assuming the probability space is understood from the context). Let $\mathbb{Z}_q$ denote the non-negative integers modulo q and let $\mathbb{Z}_q^*$ denote the corresponding multiplicative group.

An identity-based signcryption scheme consists of the following six algorithms: **Setup**, **Extract**, **Sign**, **Encrypt**, **Decrypt** and **Verify**. We describe the functions of each below.

- **Setup**: On input of a security parameter 1^k the TA uses this algorithm to produce a pair $(\texttt{params}, s)$, where `params` are the global public parameters for the system and s is the master secret key. The public parameters include a global public key Q_{TA}. We will assume that `params` are publicly known so that we do not need to explicitly provide them as input to other algorithms.
- **Extract**: On input of an identity ID_U and the master secret key s, the TA uses this algorithm to compute a secret key S_U corresponding to ID_U.
- **Sign**: User A with identity ID_A and secret key S_A uses this algorithm with input (m, S_A) to produce a signature σ on m valid under the public key derived from ID_A. It also produces some ephemeral data r.
- **Encrypt**: On input of $(S_A, ID_B, m, \sigma, r)$, ID_A uses this algorithm to produce a ciphertext c. This is the encryption of m, and ID_A's signature on m, which can be decrypted using the secret key of the user with identity ID_B.
- **Decrypt**: User B with identity ID_B and secret key S_B uses this algorithm with input (c, S_B) to produces (m, ID_A, σ) where m is a message and σ is a purported signature by ID_A on m.
- **Verify**: On input of (m, ID_A, σ), this algorithm outputs $\top$ if σ is ID_A's signature on m and it outputs $\perp$ otherwise.

The above algorithms have the following consistency requirement. If

$$(m, \sigma, r) \leftarrow \textbf{Sign}(m, S_A),\ c \leftarrow \textbf{Encrypt}(S_A, ID_B, m, \sigma, r) \text{ and}$$
$$(\hat{m}, I\hat{D}_A, \hat{\sigma}) \leftarrow \textbf{Decrypt}(c, S_B),$$

then we must have

$$I\hat{D}_A = ID_A,\ m = \hat{m} \text{ and } \top \leftarrow \textbf{Verify}(\hat{m}, I\hat{D}_A, \hat{\sigma}).$$

Note that in some models for signcryption [20] and identity-based signcryption [13, 11], the **Sign** and **Encrypt** algorithms are treated as one "signcryption" algorithm, as are the **Decrypt** and **Verify** algorithms. Our scheme supports a separation and so we stick with the above definition as in [6]. One advantage of this approach, where it is possible, is that it makes non-repudiation of messages a straightforward consequence of unforgeability. This follows from the fact that after decryption there is a publicly verifiable signature that can be forwarded to a third party.

3 Security Notions

In this section we review the security model for identity-based signcryption proposed in [6]. This model uses the notions of *insider security* and *outsider security* from [1]. Informally insider security is security against a legitimate user of the scheme while outsider security is security against an outside third party. Where appropriate, this makes insider security a stronger notion. We will comment on the significance of the distinction at relevant points in this section.

3.1 Ciphertext Authentication

A scheme offering ciphertext authentication provides the guarantee to the recipient of a signed and encrypted message that the message was encrypted by the same person who signed it. This means that the ciphertext must have been encrypted throughout the transmission and so it cannot have been the victim of a successful man-in-the-middle attack. It also implies that the signer chose the recipient for its signature.

We define this notion via a game played by a challenger and an adversary.

Game

- **Initial**: The challenger runs $\mathbf{Setup}(1^k)$ and gives the resulting `params` to the adversary. It keeps s secret.
- **Probing**: The challenger is probed by the adversary who makes the following queries.
  - **Sign/Encrypt**: The adversary submits a sender identity, a receiver identity and a message to the challenger. The challenger responds with the signature of the sender on the message, encrypted under the public key of the receiver.
  - **Decrypt/Verify**: The adversary submits a ciphertext and a receiver identity to the challenger. The challenger decrypts the ciphertext under the secret key of the receiver. It then verifies that the resulting decryption is a valid message/signature pair under the public key of the decrypted identity. If so the challenger returns the message, its signature and the identity of the signer, otherwise it returns $\perp$.
  - **Extract**: The adversary submits an identity to the challenger. The challenger responds with the secret key of that identity.
- **Forge**: The adversary returns a recipient identity ID_B and a ciphertext c. Let (m, ID_A, σ) be the result of decrypting c under the secret key corresponding to ID_B. The adversary wins if $ID_A \neq ID_B$; $\mathbf{Verify}(m, ID_A, \sigma) = \top$; no extraction query was made on ID_A, or ID_B; and c did not result from a sign/encrypt query with sender ID_A and recipient ID_B.

Definition 1. *Let $\mathcal{A}$ denote an adversary that plays the game above. If the quantity $\mathbf{Adv}[\mathcal{A}] = \mathbf{Pr}[\mathcal{A}$ wins$]$ is negligible we say that the scheme in question is existentially ciphertext-unforgeable against outsider chosen-message attacks, or AUTH-IBSC-CMA secure.*

Here we have an example of outsider security since the adversary is not able to extract the secret key corresponding to ID_B. This models the true adversarial scenario where an attack would be re-encrypting a signed message using a public key with unknown secret key.

3.2 Message Confidentiality

The accepted notion of security with respect to confidentiality for public key encryption is *indistinguishability of encryptions under adaptive chosen ciphertext attack*, as formalised in [16]. The notion of security defined in the game below is a natural adaptation of this notion to the identity-based signcryption setting.

Game

- **Initial**: The challenger runs $\mathbf{Setup}(1^k)$ and gives the resulting `params` to the adversary. It keeps s secret.
- **Phase 1**: The challenger is probed by the adversary who makes queries as in the game of Section 3.1. At the end of Phase 1 the adversary outputs two identities $\{ID_A, ID_B\}$ and two messages $\{m_0, m_1\}$. The adversary must not have made an extract query on ID_B.
- **Challenge**: The challenger chooses a bit b uniformly at random. It signs m_b under the secret key corresponding to ID_A and encrypts the result under the public key of ID_B to produce c. The challenger returns c to the adversary.
- **Phase 2**: The adversary continues to probe the challenger with the same type of queries that it made in Phase 1. It is not allowed to extract the private key corresponding to ID_B and it is not allowed to make a decrypt/verify query for c under ID_B.
- **Response**: The adversary returns a bit b'. We say that the adversary *wins* if $b' = b$.

Definition 2. *Let $\mathcal{A}$ denote an adversary that plays the game above. If the quantity $\mathbf{Adv}[\mathcal{A}] = |\mathbf{Pr}[b' = b] - \frac{1}{2}|$ is negligible we say that the scheme in question is semantically secure against adaptive chosen-ciphertext attack, or IND-IBSC-CCA2 secure.*

Note that Definition 2 deals with insider security since the adversary is assumed to have access to the private key of the sender of a signcrypted message. This means that confidentiality is preserved even if a sender's key is compromised.

3.3 Signature Non-Repudiation

A signcryption scheme offering non-repudiation prevents the sender of a signcrypted message from disavowing its signature. Note that non-repudiation is not as straightforward for signcryption as it is for digital signature schemes since we are dealing with encrypted data. As a consequence, by default, only the intended recipient of a signcryption can verify.

We define the notion of non-repudiation via the following game played by a challenger and an adversary.

Game

- **Initial**: The challenger runs $\mathbf{Setup}(1^k)$ and gives the resulting `params` to the adversary. It keeps s secret.
- **Probing**: The challenger is probed by the adversary who makes queries as in the game of Section 3.1.
- **Forge**: The adversary returns a recipient identity ID_B and a ciphertext c. Let (m, ID_A, σ) be the result of decrypting c under the secret key corresponding to ID_B. The adversary wins if $ID_A \neq ID_B$; $\mathbf{Verify}(m, ID_A, \sigma) = \top$; no extraction query was made on ID_A; no sign/encrypt query $(m, ID_A, ID_{B'})$ was responded to with a ciphertext whose decryption under the private key of $ID_{B'}$ is (m, ID_A, σ).

This model is a natural adaptation of existential unforgeability (EUF) under adaptive chosen message attack, the accepted notion of security for digital signature schemes [10].

Definition 3. *Let $\mathcal{A}$ denote an adversary that plays the game above. If the quantity $\mathbf{Adv}[\mathcal{A}] = \mathbf{Pr}[\mathcal{A}$ wins$]$ is negligible we say that the scheme in question is existentially unforgeable against insider chosen-message attacks, or EUF-IBSC-CMA secure.*

In Definition 3 we allow the adversary access to the secret key of the recipient of the forgery. It is this that gives us insider security. Also note that the adversary's advantage is with respect to its success in forging the signature within the ciphertext. This is indeed the correct definition for non-repudiation in this context because it is the signature and not the ciphertext that contains it that is forwarded to a third party in the case of a dispute.

3.4 Ciphertext Anonymity

Ciphertext anonymity is the property that ciphertexts contain no third-party extractable information that helps to identify the sender of the ciphertext or the intended recipient. It is defined via the following game.

Game

- **Initial**: The challenger runs $\mathbf{Setup}(1^k)$ and gives the resulting `params` to the adversary. It keeps s secret.
- **Phase 1**: The challenger is probed by the adversary who makes queries as in the game of Section 3.1. At the end of Phase 1 the adversary outputs a message m; two sender identities $\{ID_{A_0}, ID_{A_1}\}$; and two recipient identities $\{ID_{B_0}, ID_{B_1}\}$. The adversary must not have made an extract query on either of $\{ID_{B_0}, ID_{B_1}\}$.
- **Challenge**: The challenger chooses two bits $(b, \hat{b})$ uniformly at random. It signs m under the secret key S_{A_b} corresponding to ID_{A_b}. It then encrypts the result under the public key of $ID_{B_{\hat{b}}}$ to produce a ciphertext c. The challenger returns c to the adversary.
- **Phase 2**: The adversary continues to probe the challenger with the same type of queries that it made in Phase 1. It is not allowed to extract the private key corresponding to ID_{B_0} or ID_{B_1} and it is not allowed to make a decrypt/verify query for c under ID_{B_0} or under ID_{B_1}.
- **Response**: The adversary returns two bits $(b', \hat{b}')$. We say that the adversary wins if $b = \hat{b}$ or $b' = \hat{b}'$.

Definition 4. *Let $\mathcal{A}$ denote an adversary that plays the game above. If the quantity $\mathbf{Adv}[\mathcal{A}] = |\mathbf{Pr}[b' = b \vee \hat{b}' = \hat{b}] - \frac{3}{4}|$ is negligible we say that the scheme in question is ciphertext-anonymous against insider adaptive chosen-ciphertext attack, or ANON-IBSC-CCA2 secure.*

Note that in the equivalent definition from [6] the adversary only wins if $b = \hat{b}$ and $b' = \hat{b}'$. It is stated there that the scheme is ANON-IBSC-CCA2 secure if the quantity $\mathbf{Adv}[\mathcal{A}] = |\mathbf{Pr}[b' = b \wedge \hat{b}' = \hat{b}] - \frac{1}{4}|$ is negligible. The two definitions are clearly equivalent. We prefer our formulation because it explicitly states that the adversary should not be able to guess either of the bits. The intuition is that it gains no information about the sender of a message or the intended recipient. Definition 4 follows from the fact that the adversary is always able to guess at least one of the bits correctly with probability 3/4.

An additional security definition dubbed ciphertext unlinkability is described in [6]. Informally this notion means that Alice is able to deny having sent a given ciphertext to Bob, even if the ciphertext decrypts under Bob's secret key to a message bearing Alice's signature. This property is demonstrated for the scheme in [6] by showing that given a message signed by Alice, Bob is able to create a valid ciphertext addressed to himself for that message. It is easily verified that our scheme also has this property.

4 The Scheme

In this section we describe how our identity-based signcryption scheme works. We will refer to the scheme as IBSC henceforth.

Before explaining our scheme we must briefly summarise the mathematical primitives necessary for pairing based cryptography. We require two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of large prime order q. These groups must be such that there exists a non-degenerate, efficiently computable map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. This map must be bilinear i.e. for all $P_1, P_2 \in \mathbb{G}_1$ and all $a, b \in \mathbb{Z}_q^*$ we have $\hat{e}(aP_1, bP_2) = \hat{e}(P_1, P_2)^{ab}$. A popular construction for such groups uses supersingular elliptic curves over finite fields. The bilinear map is realised using a modification of the Tate pairing or the Weil pairing. For details of such instantiations see [2, 5].

We also require three hash functions $H_0 : \{0,1\}^{k_1} \rightarrow \mathbb{G}_1$, $H_1 : \{0,1\}^{k_0+n} \rightarrow \mathbb{Z}_q^*$ and $H_2 : \mathbb{G}_2 \rightarrow \{0,1\}^{k_0+k_1+n}$. Here k_0 is the number of bits required to represent an element of $\mathbb{G}_1$; k_1 is the number of bits required to represent an identity; and n is the number of bits of a message to be signed and encrypted.

Setup
Establish parameters $\mathbb{G}_1$, $\mathbb{G}_2$, q, $\hat{e}$, $H_0 : \{0,1\}^{k_1} \rightarrow \mathbb{G}_1$, $H_1 : \{0,1\}^{k_0+n} \rightarrow \mathbb{Z}_q^*$ and $H_2 : \mathbb{G}_2 \rightarrow \{0,1\}^{k_0+k_1+n}$ as described above; choose P such that $\langle P \rangle = \mathbb{G}_1$; choose $s \leftarrow \mathbb{Z}_q^*$ and compute the global public key $Q_{TA} \leftarrow sP$.

Extract
To extract the private key for user U with $ID_U \in \{0,1\}^{k_1}$: Compute the public key $Q_U \leftarrow H_0(ID_U)$ and the secret key $S_U \leftarrow sQ_U$.

Sign
For user A with identity ID_A to sign $m \in \{0,1\}^n$ with private key S_A corresponding to public key $Q_A \leftarrow H_0(ID_A)$: Choose $r \leftarrow \mathbb{Z}_q^*$; compute $X \leftarrow rQ_A$,

$h_1 \leftarrow H_1(X||m)$ and $Z \leftarrow (r + h_1)S_A$; return the signature (X, Z) and forward (m, r, X, Z) to **Encrypt**.

Encrypt
For user A with identity ID_A to encrypt m using r, X, Z output by **Sign** for receiver ID_B: Compute $Q_B \leftarrow H_0(ID_B)$, $w \leftarrow \hat{e}(rS_A, Q_B)$ and $y \leftarrow H_2(w) \oplus (Z||ID_A||m)$; return ciphertext (X, y).

Decrypt
For user B with identity ID_B to decrypt (X, y) using $S_B = sH_0(ID_B)$: Compute $w \leftarrow \hat{e}(X, S_B)$ and $Z||ID_A||m \leftarrow y \oplus H_2(w)$; forward message m, signature (X, Z) and purported sender ID_A to **Verify**.

Verify
To verify user A's signature (X, Z) on message m where A has identity ID_A: Compute $Q_A \leftarrow H_0(ID_A)$ and $h_1 \leftarrow H_1(X||m)$; if $\hat{e}(Z, P) = \hat{e}(Q_{TA}, X + h_1 Q_A)$, return $\top$; else, return $\perp$.

Note that, as was the case in [6], the key setup used by our scheme is that proposed in [17], and the signing algorithm is that proposed in [7]. Also, the encryption is done in a manner similar to the BasicIdent scheme from [5]. The integrity checking necessary for security against adaptive adversaries comes from the signature in our case.

5 Security Results

In this section we state our security results. Owing to space constraints, we only provide a proof of the ciphertext authentication property here. The proofs of the other properties may be found in the full version of the paper [8].

All our results are relative to the *bilinear Diffie-Hellman* (BDH) problem. Informally, using the notation of Section 4, this is the problem of computing $\hat{e}(P, P)^{abc}$ from (P, aP, bP, cP) where a, b, c are chosen at random from $\mathbb{Z}_q^*$ and P generates $\mathbb{G}_1$. For further details see [5].

To prove our results we model H_0, H_1 and H_2 as random oracles [3]. We assume that the adversary makes q_i queries to H_i for $i = 0, 1, 2$. The number of sign/encrypt and decrypt/verify queries made by the adversary are denoted q_s and q_d respectively.

Ciphertext Authentication

Theorem 1. *If there is an AUTH-IBSC-CMA adversary $\mathcal{A}$ of IBSC that succeeds with probability ϵ, then there is a simulator $\mathcal{B}$ running in polynomial time that solves the BDH problem with probability at least*

$$\epsilon \cdot \left(1 - \frac{q_s(q_1 + q_2 + 2q_s)}{q}\right) \cdot \frac{1}{q_0(q_0 - 1)(q_s + q_d)(q_2 + q_s)}.$$

Proof. We will show how an AUTH-IBSC-CMA adversary $\mathcal{A}$ of IBSC may be used to construct a simulator $\mathcal{B}$ that solves the BDH problem for (P, aP, bP, cP).

We now describe the construction of the simulator $\mathcal{B}$. The simulator runs $\mathcal{A}$ with trusted third party public key $Q_{TA} \leftarrow cP$. It also creates algorithms to respond to queries made by $\mathcal{A}$ during its attack. To maintain consistency between queries made by $\mathcal{A}$, the simulator keeps the following lists: L_i for $i = 0, 1, 2$ of data for query/response pairs to random oracle H_i; L_s of signcryptions generated by the simulator; and L_d of some of the queries made by $\mathcal{A}$ to the decrypt/verify oracle. We will see in the construction of the sign/encrypt simulator that the list L_s stores other information that will be useful to $\mathcal{B}$. Its use will become apparent in the subsequent analysis, as will the use of L_d.

Simulator: $H_0(ID_U)$
At the beginning of the simulation choose i_a, i_b uniformly at random from $\{1, \ldots, q_0\}$ $(i_a \neq i_b)$. We respond to the i-th query made by $\mathcal{A}$ as follows (assuming $\mathcal{A}$ does not make repeat queries).

- If $i = i_a$ then respond with $H_0(ID_U) \leftarrow aP$ and set $ID_A \leftarrow ID_U$.
- If $i = i_b$ then respond with $H_0(ID_U) \leftarrow bP$ and set $ID_B \leftarrow ID_U$.
- Else choose $x \leftarrow \mathbb{Z}_q^*$; compute $Q_U \leftarrow xP$; compute $S_U \leftarrow xQ_{TA}$; store (ID_U, Q_U, S_U, x) in L_0 and respond with Q_U.

Simulator: $H_1(X||m)$

- If $(X||m, h_1) \in L_1$ for some h_1, return h_1.
- Else choose $h_1 \leftarrow \mathbb{Z}_q^*$; add $(X||m, h_1)$ to L_1; return h_1.

Simulator: $H_2(w)$

- If $(w, h_2) \in L_2$ for some h_2, return h_2.
- Else choose $h_2 \leftarrow \{0, 1\}^{k_0+k_1+n}$; add (w, h_2) to L_2; return h_2.

Simulator: **Extract**(ID_U)
We assume that $\mathcal{A}$ queries $H_0(ID_U)$ before it makes the extraction query ID_U.

- If $ID_U = ID_A$ or $ID_U = ID_B$, abort the simulation.
- Else search L_0 for the entry (ID_U, Q_U, S_U, x) corresponding to ID_U and return S_U.

Simulator: **Sign/Encrypt**(m, ID_1, ID_2)
We will assume that $\mathcal{A}$ makes the queries $H_0(ID_1)$ and $H_0(ID_2)$ before it makes a sign/encrypt query using these identities. We have five cases to consider.

Case 1: $ID_1 \neq ID_A$ and $ID_1 \neq ID_B$

- Find the entry (ID_1, Q_1, S_1, x) in L_0; choose $r \leftarrow \mathbb{Z}_q^*$; compute $X \leftarrow rQ_1$; compute $h_1 \leftarrow H_1(X||m)$ (where H_1 is the simulator above); compute $Z \leftarrow (r + h_1)S_1$; compute $Q_2 \leftarrow H_0(ID_2)$ (where H_0 is the simulator above); compute $w \leftarrow \hat{e}(rS_1, Q_2)$; compute $y \leftarrow H_2(w) \oplus (Z||ID_1||m)$ (where H_2 is the simulator above); return (X, y).

Case 2: $ID_1 = ID_A$, $ID_2 \neq ID_A$ and $ID_2 \neq ID_B$

– Choose $r, h_1 \leftarrow \mathbb{Z}_q^*$; compute $X \leftarrow rP - h_1Q_A$; compute $Z \leftarrow rQ_{TA}$; add $(X||m, h_1)$ to L_1; find the entry (ID_2, Q_2, S_2, x) in L_0; compute $w \leftarrow \hat{e}(X, S_2)$; compute $y \leftarrow H_2(w) \oplus (Z||ID_A||m)$ (where H_2 is the simulator above); return (X, y).

Case 3: $ID_1 = ID_B$, $ID_2 \neq ID_A$ and $ID_2 \neq ID_B$
Use the simulation of Case 2 replacing (ID_A, Q_A) with (ID_B, Q_B).

Case 4: $ID_1 = ID_A$ and $ID_2 = ID_B$

– Follow the first four steps of Case 2; choose $h_2 \leftarrow \{0,1\}^{k_0+k_1+n}$; compute $y \leftarrow h_2 \oplus Z||ID_A||m$; add $(ID_A, ID_B, X, y, Z, m, r, h_1, h_2)$ to L_s; return (X, y).

Case 5: $ID_1 = ID_B$ and $ID_2 = ID_A$
Use the simulation of Case 4 swapping (ID_A, Q_A, ID_B) with (ID_B, Q_B, ID_A).

Decrypt/Verify:$(X, y), ID_2$
We assume that $\mathcal{A}$ makes the query $H_0(ID_2)$ before making a decryption query for ID_2. We have the following three cases to consider.

Case 1: $ID_2 \neq ID_A$ and $ID_2 \neq ID_B$

– Find the entry (ID_2, Q_2, S_2, x) in L_0; compute $w = \hat{e}(X, S_2)$; initialise $b \leftarrow 1$.
– If $w \in L_2$, compute $Z||ID_1||m \leftarrow y \oplus H_2(w)$, else $b \leftarrow 0$.
– If $b = 1$ and $ID_1 \in L_0$, let $Q_1 \leftarrow H_0(ID_1)$, else $b \leftarrow 0$.
– If $b = 1$ and $X||m \in L_1$, let $h_1 \leftarrow H_1(X||m)$, else $b \leftarrow 0$.
– If $b = 1$ and $\hat{e}(Z, P) = \hat{e}(Q_{TA}, X + h_1Q_1)$, return m, (X, Z) and ID_1, else step through the list L_s as follows.
  - If the current entry has the form $(ID_A, ID_B, X', y, Z, m', r, h_1', h_2)$ then test if $\hat{e}(X', Q_B) = \hat{e}(X, xP)$. If so continue, else move on to the next element of L_s and begin again.
  - Else if the current entry has the form $(ID_B, ID_A, X', y, Z, m', r, h_1', h_2)$ then test $\hat{e}(X', Q_A) = \hat{e}(X, xP)$. If so continue, else move on to the next element of L_s and begin again.
  - Compute $Z||ID_1||m \leftarrow y \oplus h_2$.
  - If $ID_1 = ID_2$ move to the next element in L_s and begin again.
  - If $ID_1 \in L_0$ let $Q_1 \leftarrow H_0(ID_1)$, else move to the next element in L_s.
  - If $X||m \in L_1$ let $h_1 \leftarrow H_1(X||m)$, else move to the next element in L_s.
  - Check that $\hat{e}(Z, P) = \hat{e}(Q_{TA}, X + h_1Q_1)$, if so return m, (X, Z) and ID_1, if not move on to the next element in L_s and begin again.
– If no message has been returned, return $\perp$.

Case 2: $ID_2 = ID_B$

– If $(ID_A, ID_B, X, y, Z, m, r, h_1, h_2) \in L_s$ for some m, return m, (X, Z), ID_A.

- Else, add $(X, y), ID_B$ to L_d and step through the list L_2 with entries (w, h_2) as follows.
  - Compute $Z||ID_1||m \leftarrow y \oplus h_2$.
  - If $ID_1 = ID_A$ or $ID_1 = ID_B$, move to the next element in L_2 and begin again.
  - If $ID_1 \in L_0$ let $Q_1 \leftarrow H_0(ID_1)$ and find S_1 in L_0 , else move to the next element in L_2 and begin again.
  - If $X||m \in L_1$ let $h_1 \leftarrow H_1(X||m)$, else move on to the next element in L_2 and begin again.
  - Check that $w = \hat{e}(Z - h_1 S_1, Q_B)$ and if not move on to the next element in L_2 and begin again.
  - Check that $\hat{e}(Z, P) = \hat{e}(Q_{TA}, X + h_1 Q_1)$, if so return m, (X, Z) and ID_1, else move on to the next element in L_2 and begin again.
- If no message has been returned after stepping through the list L_2, step through the list L_s as follows.
  - If the current entry has the form $(ID_A, ID_B, X', y, Z, m', r, h'_1, h_2)$ then check that $X' = X$. If so continue, else move on to the next element of L_s and begin again.
  - Else if the current entry has the form $(ID_B, ID_A, X', y, Z, m', r, h'_1, h_2)$ then check that $\hat{e}(X', Q_A) = \hat{e}(X, Q_B)$. If so continue, if not move on to the next element of L_s and begin again.
  - Compute $Z||ID_1||m \leftarrow y \oplus h_2$.
  - If $ID_1 = ID_B$, move to the next element in L_s and begin again.
  - If $ID_1 \in L_0$ let $Q_1 \leftarrow H_0(ID_1)$, else move to the next element in L_s.
  - If $X||m \in L_1$ let $h_1 \leftarrow H_1(X||m)$, else move to the next element in L_s.
  - Check that $\hat{e}(Z, P) = \hat{e}(Q_{TA}, X + h_1 Q_1)$, if so return m, (X, Z) and ID_1, else move on to the next element in L_s and begin again.
- If no message has been returned, return $\perp$.

Case 3: $ID_2 = ID_A$
Use the simulation of Case 2 replacing (ID_B, Q_B, ID_A) with (ID_A, Q_A, ID_B).

Once $\mathcal{A}$ has been run, $\mathcal{B}$ does one of two things.

1. With probability $q_s/(q_s + q_d)$ choose a random element from L_s and a random element (w, h_2) from L_2. We call this event Ch_1 in the analysis below (Ch for choice). The significance of the probability will become apparent in the subsequent analysis we only mention here that we are assuming $|L_s| = q_s$ at the end of our simulation. This is the worst case scenario.
   - If the chosen element has form $(ID_A, ID_B, X, y, Z, m, r, h_1, h_2)$, compute
   $$B = \big(w/\hat{e}(rbP, cP)\big)^{-1/h_1}.$$
   - If the chosen element has form $(ID_B, ID_A, X, y, Z, m, r, h_1, h_2)$, compute
   $$B = \big(w/\hat{e}(raP, cP)\big)^{-1/h_1}.$$

2. With probability $q_d/(q_s + q_d)$ choose a random element from L_d and a random element (w, h_2) from L_2. We call this event Ch_2 in the analysis below. Again, the significance this probability will become apparent in the subsequent analysis. As above, we are assuming $|L_d| = q_d$ at the end of our simulation. This is the worst case scenario.
   - If the chosen element from L_d has the form $(X, y), ID_B$ compute $y \oplus h_2$. If $y \oplus h_2$ has the form $Z||ID_A||m$ for some Z, m, compute

     $$B = \left(w/\hat{e}(Z, bP)\right)^{-1/h_1}.$$

     If $y \oplus h_2$ does not have this form $\mathcal{B}$ has failed.
   - If the chosen element from L_d has the form $(X, y), ID_A$ compute $y \oplus h_2$. If $y \oplus h_2$ has the form $Z||ID_B||m$ for some Z, m, compute

     $$B = \left(w/\hat{e}(Z, aP)\right)^{-1/h_1}.$$

     If $y \oplus h_2$ does not have this form $\mathcal{B}$ has failed.

The rational for these probabilities and computations will become apparent in the discussion of equations (1), (2), (4) and (5) below.

Let us now analyse our simulation. The simulations for the random oracles and the extraction queries are trivial. The simulation of the sign/encrypt queries uses standard techniques. We make some remarks about the decrypt/verify simulation since this is less obvious. We will treat each case separately.

Case 1: In this case the simulator $\mathcal{B}$ knows the secret key of the receiver and so it is able to compute the correct ephemeral encryption key. The first six steps in this case are therefore those that would be followed in genuine decryption and verification. The reason that it does not stop at this point is that the sign/encrypt simulator implicitly defines $H_2(w)$ for values of w that are unknown to the simulator. It must check that the ephemeral encryption key w that it has computed is not one of these values. For example, suppose that there is an entry of the form $(ID_A, ID_B, X', y, Z, m, r, h_1', h_2)$ in L_s. Referring back to the construction of the sign/encrypt simulator, it needs to know if

$$\hat{e}(X', S_B) = \hat{e}(X, S_2).$$

The simulator knows that $S_2 = xQ_{TA} = xcP$ and it know $S_B = bQ_{TA} = bcP = cQ_B$ so this test becomes

$$\hat{e}(X', Q_B) = \hat{e}(X, xP).$$

Case 2: In this case the simulator $\mathcal{B}$ does not know the secret key of the receiver and so it is unable to compute the ephemeral encryption key $\hat{e}(X, S_B)$. The first loop, through the list L_2, determines whether the H_2 value of the ephemeral encryption key is in L_2 itself i.e. for each w in L_2 it wants to know if $w = \hat{e}(X, S_B)$. Since by construction $Q_{TA} = cP$ this test becomes $w = \hat{e}(cX, Q_B)$

and, under the assumption that the ciphertext is correctly formed, it becomes $\hat{e}(Z - h_1 S_1, Q_B)$. Note that if the ciphertext is not correctly formed the simulator does not care whether the value of $H_2(w)$ is defined since it is correct to reject. The final test in this loop is just the standard test for verification.

The second loop, through L_s, determines whether the value of $H_2(w)$ that $\mathcal{B}$ is looking for has been determined by the sign/encrypt simulator. If it is searching L_s for an entry of form $(ID_A, ID_B, X', y, Z, m, r, h'_1, h_2)$ then the receiver identities are the same in this entry and in the decrypt/verify query that we are trying to respond to. The check is then simply on the values of X and X'.

If $\mathcal{B}$ is looking at an entry of L_s of the form $(ID_B, ID_A, X', y, Z, m, r, h'_1, h_2)$ then the receivers identities are not the same in this entry and in the decrypt/verify query that it is trying to respond to. The check that it wishes to perform is $\hat{e}(X', S_A) = \hat{e}(X, S_B)$. This is clearly equivalent to the check $\hat{e}(X', Q_A) = \hat{e}(X, Q_B)$.

Case 3: The analysis is identical to that of Case 2 with A and B reversed.

Let us now consider how our simulation could fail i.e. describe events that could cause $\mathcal{A}$'s view to differ when run by $\mathcal{B}$ from its view in a real attack. We call such an event an error and denote it ER.

It is clear that the simulations for H_0 and H_1 are indistinguishable from real random oracles. Let us now consider the H_2 simulator. The important point here is that H_2 is not only defined at points where the H_2 simulator is called by $\mathcal{A}$ or by the simulator itself. It is also defined at certain points implicitly by the sign/encrypt simulator. For example, suppose that the sign/encrypt simulator responds to a query m, ID_A, ID_B. In this case it adds an entry $(ID_A, ID_B, X, y, Z, m, r, h_1, h_2)$ to L_s. This implicitly defines $H_2(\hat{e}(X, S_B)) = h_2$ although it is not actually able to compute $\hat{e}(X, S_B)$. If the H_2 simulator is subsequently called with $w = \hat{e}(X, S_B)$ it will not recognise it and so it will not return h_2. We denote such events H-ER. However, if such an event occurs we have

$$w = \hat{e}(X, S_B) = \hat{e}(rP - h_1 Q_A, S_B)$$

from which it is possible to compute

$$\hat{e}(P,P)^{abc} = \hat{e}(Q_A, S_B) = \big(w/\hat{e}(rQ_B, Q_{TA})\big)^{-1/h_1} = \big(w/\hat{e}(rbP, cP)\big)^{-1/h_1}. \quad (1)$$

Similarly if the H_2 simulator is called with w that is implicitly defined by an entry $(ID_B, ID_A, X, y, Z, m, r, h_1, h_2) \in L_s$ we can compute.

$$\hat{e}(P,P)^{abc} = \hat{e}(Q_B, S_A) = \big(w/\hat{e}(rQ_A, Q_{TA})\big)^{-1/h_1} = \big(w/\hat{e}(raP, cP)\big)^{-1/h_1}. \quad (2)$$

Let us now consider how the simulation for sign/encrypt could fail. We denote such an event S-ER. The most likely failure will be caused by the sign/encrypt simulator responding to a query of the form Case 4 or Case 5 (see simulator). Since we do not know how often each case will occur we will be conservative and assume that each query will be one of these, 4 say. The only possibilities for introducing an error here are defining $H_1(X||m)$ when it is already defined or defining $H_2(\hat{e}(X, S_B))/H_2(\hat{e}(X, S_A))$ when it is already defined. Since X takes its value uniformly at random in $\langle P \rangle$, the chance of one of these events occurring is at most $(q_1 + q_2 + 2q_s)/q$ for each query. The $2q_s$ comes from the fact that the signing simulator adds elements to L_1 and L_2. Therefore, over the whole simulation, the chance of an error introduced in this way is at most

$$q_s(q_1 + q_2 + 2q_s)/q. \tag{3}$$

We now turn our attention to the decrypt/verify simulator. An error in this simulator is denoted D-ER. It is clear that this simulator never accepts an invalid encryption. What we have to worry about is the possibility that it rejects a valid one. This can only occur with non-negligible probability in Case 2 or Case 3. Suppose that we are trying to decrypt $(X, y), ID_B$ (i.e. Case 2). An error will only occur if while stepping through L_2 there is an entry (w, h_2) such that $Z||ID_A||m \leftarrow y \oplus h_2$ and (X, y) is a valid encryption of m from ID_A to ID_B. In this case we must have

$$w = \hat{e}(Z - h_1 S_A, Q_B) = \hat{e}(Z, Q_B) \cdot \hat{e}(-h_1 S_A, Q_B) = \hat{e}(Z, bP) \cdot \hat{e}(-h_1 acP, bP),$$

where $h_1 = H_1(X||m)$. From the above we can compute

$$\hat{e}(P, P)^{abc} = \big(w/\hat{e}(Z, bP)\big)^{-1/h_1}. \tag{4}$$

Suppose now that we are trying to decrypt $(X, y), ID_A$ (i.e. Case 3). An error will only occur if while stepping through L_2 there is an entry (w, h_2) such that $Z||ID_B||m \leftarrow y \oplus h_2$ and (X, y) is a valid encryption of m from ID_B to ID_A. In this case we must have

$$w = \hat{e}(Z - h_1 S_B, Q_A) = \hat{e}(Z, Q_A) \cdot \hat{e}(-h_1 S_B, Q_A) = \hat{e}(Z, aP) \cdot \hat{e}(-h_1 bcP, aP),$$

from which we can compute

$$\hat{e}(P, P)^{abc} = \big(w/\hat{e}(Z, aP)\big)^{-1/h_1}. \tag{5}$$

The final simulator is the extract simulator. Note that the adversary will only succeed in its task with non-negligible probability if it queries H_0 with the two identities under which the encrypted and signed message it produces is supposed to be valid. Looking at the H_0 simulator we see that it chooses two H_0 queries made by the adversary and responds to these with group elements from the BDH instance that it is trying to solve. The simulator hopes that these will be the identities for the adversary's encrypted and signed message. This will be the case with probability at least

$$1/q_0(q_0 - 1). \tag{6}$$

If this is not the case we say that an error has occurred in the extract simulator because, if the adversary tried to extract the private key for these identities, the simulator would abort. An error in the extract simulator is denoted E-ER.

Once $\mathcal{A}$ has been run by the simulator $\mathcal{B}$, there are two courses of action: Ch_1 and Ch_2 (as described above). If Ch_1 has been chosen, we denote the event that $\mathcal{B}$ selects the correct elements to solve the BDH problem from L_s and H_2

by CG_1 (under the assumption that there are such correct elements in the lists at the end of the simulation). Likewise if Ch_2 has been chosen, we denote the event that $\mathcal{B}$ selects the correct elements from L_d and H_2 by CG_2.

With the events described above we have

$$\begin{aligned}\mathbf{Adv}[\mathcal{B}] \geq\ & \mathbf{Pr}[\neg\mathsf{E\text{-}ER} \wedge \mathsf{H\text{-}ER} \wedge \neg\mathsf{S\text{-}ER} \wedge \mathsf{Ch}_1 \wedge \mathsf{CG}_1] \\ & + \mathbf{Pr}[\mathsf{D\text{-}ER} \wedge \neg\mathsf{E\text{-}ER} \wedge \neg\mathsf{H\text{-}ER} \wedge \neg\mathsf{S\text{-}ER} \wedge \mathsf{Ch}_2 \wedge \mathsf{CG}_2]. \end{aligned} \tag{7}$$

Also,

$$\begin{aligned}& \mathbf{Pr}[\neg\mathsf{E\text{-}ER} \wedge \mathsf{H\text{-}ER} \wedge \neg\mathsf{S\text{-}ER} \wedge \mathsf{Ch}_1 \wedge \mathsf{CG}_1] \\ &= \mathbf{Pr}[\neg\mathsf{E\text{-}ER} \wedge \neg\mathsf{S\text{-}ER}] \cdot \mathbf{Pr}[\mathsf{Ch}_1 \wedge \mathsf{CG}_1] \cdot \mathbf{Pr}[\mathsf{H\text{-}ER}], \end{aligned} \tag{8}$$

and,

$$\begin{aligned}& \mathbf{Pr}[\mathsf{D\text{-}ER} \wedge \neg\mathsf{E\text{-}ER} \wedge \neg\mathsf{H\text{-}ER} \wedge \neg\mathsf{S\text{-}ER} \wedge \mathsf{Ch}_2 \wedge \mathsf{CG}_2] \\ &= \mathbf{Pr}[\mathsf{D\text{-}ER}] \cdot \mathbf{Pr}[\neg\mathsf{E\text{-}ER} \wedge \neg\mathsf{H\text{-}ER} \wedge \neg\mathsf{S\text{-}ER}] \cdot \mathbf{Pr}[\mathsf{Ch}_2 \wedge \mathsf{CG}_2]. \end{aligned} \tag{9}$$

Note that, in the event $\neg\mathsf{E\text{-}ER} \wedge \neg\mathsf{H\text{-}ER} \wedge \neg\mathsf{S\text{-}ER}$, the adversary $\mathcal{A}$ is run by $\mathcal{B}$ in exactly the same way that it would be run in a real attack until the event $\mathsf{D\text{-}ER}$ occurs. Moreover, in the event $\neg\mathsf{E\text{-}ER} \wedge \neg\mathsf{H\text{-}ER} \wedge \neg\mathsf{S\text{-}ER}$, $\mathcal{A}$ winning and $\mathsf{D\text{-}ER}$ are equivalent. This means that (9) becomes

$$\begin{aligned}& \mathbf{Pr}[\mathsf{D\text{-}ER} \wedge \neg\mathsf{E\text{-}ER} \wedge \neg\mathsf{H\text{-}ER} \wedge \neg\mathsf{S\text{-}ER} \wedge \mathsf{Ch}_2 \wedge \mathsf{CG}_2] \\ &= \epsilon \cdot \mathbf{Pr}[\neg\mathsf{E\text{-}ER} \wedge \neg\mathsf{S\text{-}ER}] \cdot \mathbf{Pr}[\mathsf{Ch}_2 \wedge \mathsf{CG}_2] \cdot \mathbf{Pr}[\neg\mathsf{H\text{-}ER}]. \end{aligned} \tag{10}$$

From the definitions of Ch_1, CG_1, Ch_2 and CG_2 above it is clear that

$$\mathbf{Pr}[\mathsf{Ch}_1 \wedge \mathsf{CG}_1] = \frac{q_s}{q_s + q_d} \cdot \frac{1}{q_s(q_2 + q_s)} = \frac{1}{(q_s + q_d)(q_2 + q_s)} \text{ and} \tag{11}$$

$$\mathbf{Pr}[\mathsf{Ch}_2 \wedge \mathsf{CG}_2] = \frac{q_d}{q_s + q_d} \cdot \frac{1}{q_d(q_2 + q_s)} = \frac{1}{(q_s + q_d)(q_2 + q_s)}. \tag{12}$$

Note that we are assuming a worst case scenario here i.e. $|L_s| = q_s$ and $|L_d| = q_d$. We will make this assumption throughout the remaining analysis without further comment. ¿From the fact that $\mathbf{Pr}[\mathsf{H\text{-}ER}] + \mathbf{Pr}[\neg\mathsf{H\text{-}ER}] = 1$, (7), (8), (10), (11) and (12) we have

$$\begin{aligned}\mathbf{Adv}[\mathcal{B}] &\geq (\mathbf{Pr}[\mathsf{H\text{-}ER}] + \epsilon \cdot \mathbf{Pr}[\neg\mathsf{H\text{-}ER}]) \cdot \mathbf{Pr}[\neg\mathsf{E\text{-}ER} \wedge \neg\mathsf{S\text{-}ER}] \cdot \frac{1}{(q_s + q_d)(q_2 + q_s)} \\ &\geq \epsilon \cdot (\mathbf{Pr}[\mathsf{H\text{-}ER}] + \mathbf{Pr}[\neg\mathsf{H\text{-}ER}]) \cdot \mathbf{Pr}[\neg\mathsf{E\text{-}ER} \wedge \neg\mathsf{S\text{-}ER}] \cdot \frac{1}{(q_s + q_d)(q_2 + q_s)} \\ &= \epsilon \cdot \mathbf{Pr}[\neg\mathsf{E\text{-}ER} \wedge \neg\mathsf{S\text{-}ER}] \cdot \frac{1}{(q_s + q_d)(q_2 + q_s)}. \end{aligned} \tag{13}$$

Finally, by the independence of $\mathsf{E\text{-}ER}$ and $\mathsf{S\text{-}ER}$, using (3), (6) and (13) we have

$$\mathbf{Adv}[\mathcal{B}] \geq \epsilon \cdot \left(1 - \frac{q_s(q_1 + q_2 + 2q_s)}{q}\right) \cdot \frac{1}{q_0(q_0 - 1)(q_s + q_d)(q_2 + q_s)}. \tag{14}$$

□

Message Confidentiality

Theorem 2 describes the security of our scheme under Definition 2. We provide a proof in the full version of the paper[8].

Theorem 2. *If there is an IND-IBSC-CCA2 adversary $\mathcal{A}$ of IBSC that succeeds with probability ϵ, then there is a simulator $\mathcal{B}$ running in polynomial time that solves the BDH problem with probability at least*

$$\epsilon \cdot \left(1 - \frac{q_s(q_1 + q_s)}{q}\right) \cdot \frac{1}{q_0 q_2}.$$

Signature Non-Repudiation

In Theorem 3 we state the security result for our scheme under Definition 3. The proof will be found in the full version of the paper[8].

Theorem 3. *If there is an EUF-IBSC-CMA adversary $\mathcal{A}$ of IBSC that succeeds with probability ϵ, then there is a simulator $\mathcal{B}$ running in polynomial time that solves the BDH problem with probability at least*

$$\epsilon \cdot \left(1 - \frac{q_s(q_1 + q_s)}{q}\right)^2 \cdot \frac{1}{4q_0^2(q_1 + q_s)^2}.$$

Ciphertext Anonymity

Our final security result is Theorem 4. This deals with security under Definition 4. The proof appears in the full version of the paper[8].

Theorem 4. *If there is an ANON-IBSC-CCA2 adversary $\mathcal{A}$ of IBSC that succeeds with probability ϵ, then there is a simulator $\mathcal{B}$ running in polynomial time that solves the BDH problem with probability at least*

$$\epsilon \cdot \left(1 - \frac{q_s(q_1 + q_2 + 2q_s)}{q}\right) \cdot \frac{1}{q_0(q_0 - 1)(2 + q_s)(q_2 + q_s)}.$$

6 Performance and Security Comparison

We compare our scheme with other schemes appearing in the literature in Table 1. We assume that all schemes are implemented with the same $\mathbb{G}_1$, $\mathbb{G}_2$, $\hat{e}$ and q as defined in Section 4.

The 1, 2, 3 and 4 in the "security" column refer to security under Definition 1, 2, 3 and 4 respectively. A y means that the scheme provably meets the definition, a n means that the scheme is not secure under the definition, and a ? means that the status is unknown.

In the "ciphertext size" column we let n_1 be the number of bits required to represent an element of $\mathbb{G}_1$, n_q be the number of bits required to represent an element of $\mathbb{F}_q$, n_{id} be the number of bits required to represent an identity, and

scheme	security				ciphertext size	sign/encrypt ops.			decrypt/verify ops.		
	1	2	3	4		$\mathbb{G}_1$	$\mathbb{G}_2$	$\hat{e}$	$\mathbb{G}_1$	$\mathbb{G}_2$	$\hat{e}$
[6]	y	y	y	y	$2n_1 + n_{id} + m$	3	1	1	2	0	4
[11]	?	y	y	n	$n_1 + n_q + m$	2	2	2	0	2	4
[13]	y	n	y	n	$2n_1 + m$	3	0	1	0	1	4
[14]	?	?	?	n	$3n_1 + m$	3	1	0	1	0	2
[15]	?	?	?	n	$2n_1 + m$	2	1	1	0	1	3
[19]	?	y	y	?	$2n_1 + n_{id} + m$	4	0	1	1	0	3
ours	y	y	y	y	$2n_1 + n_{id} + m$	3	0	1	1	0	3

Table 1. A comparison between various schemes in the literature

m be the number of bits in the message being signcrypted. The ciphertext size is therefore measured in bits.

In the "sign/encrypt ops." and "decrypt/verify ops." columns, the sub-columns $\mathbb{G}_1$, $\mathbb{G}_2$ and $\hat{e}$ hold the number of multiplications in $\mathbb{G}_1$, exponentiations in $\mathbb{G}_2$ and computations of $\hat{e}$ respectively.

Note that the scheme in [14] has a slight computational overhead for computing public keys when compared to the other schemes we have mentioned. This is not reflected in the table above.

7 Conclusions

We have proposed an identity-based signcryption scheme that is the most efficient among the provably secure schemes of its type proposed to date. Our scheme admits a full security analysis in the model of Boyen [6].

Our security analysis, like the security analysis for all provably secure identity-based signcryption schemes, requires the random oracle model [3]. Techniques have recently been developed for designing identity-based encryption schemes with provable security in the standard model [4]. It would be interesting to know if these, or other, techniques can be applied to identity based signcryption.

References

1. J. H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In *Advances in Cryptology - EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 83–107. Springer-Verlag, 2002.
2. P. S. L. M. Barreto, H. Y. Kim, B. Lynn, and M. Scott. Efficient algorithms for paring-based cryptosystems. In *Advances in Cryptology - CRYPTO 2002*, volume 2442 of *LNCS*, pages 354–368. Springer-Verlag, 2002.
3. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In 1^{st} *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

4. D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *LNCS*, pages 443–459. Springer-Verlag, 2004.
5. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In *Advances in Cryptology - CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer-Verlag, 2001.
6. X. Boyen. Multipurpose identity-based signcryption: A swiss army knife for identity-based cryptography. In *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *LNCS*, pages 382–398. Springer-Verlag, 2003.
7. J. C. Cha and J. H. Cheon. An identity-based signature from gap Diffie-Hellman groups. In *Public Key Cryptography - PKC 2003*, volume 2567 of *LNCS*, pages 18–30. Springer-Verlag, 2003.
8. L. Chen and J. Malone-Lee. Improved identity-based sincryption. Cryptology ePrint Archive, Report 2004/114, 2004. `http://eprint.iacr.org/`.
9. C. Cocks. An identity-based encryption scheme based on quadratic residues. In *Cryptography and Coding*, volume 2260 of *LNCS*, pages 360–363. Springer-Verlag, 2001.
10. S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.
11. B. Libert and J. J. Quisquater. New identity-based signcryption schemes from pairings. In *IEEE Information Theory Workshop 2003*. Full version available at `http://eprint.iacr.org/2003/023/`.
12. B. Libert and J. J. Quisquater. Efficient signcryption with key privacy from gap Diffie-Hellman groups. In *Public Key Cryptography - PKC 2004*, volume 2947 of *LNCS*, pages 187–200. Springer-Verlag, 2004.
13. J. Malone-Lee. Identity-based signcryption. Cryptology ePrint Archive, Report 2002/098, 2002. `http://eprint.iacr.org/`.
14. Noel McCullagh and Paulo S. L. M. Barreto. Efficient and forward-secure identity-based signcryption. Cryptology ePrint Archive, Report 2004/117, 2004. `http://eprint.iacr.org/`.
15. D. Nalla and K. C. Reddy. Signcryption scheme for identity-based cryptosystems. Cryptology ePrint Archive, Report 2003/066, 2003. `http://eprint.iacr.org/`.
16. C. Rackoff and D. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Advances in Cryptology - CRYPTO '91*, volume 576 of *LNCS*, pages 433–444. Springer-Verlag, 1992.
17. R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairings. In *Symposium on Cryptography and Information Security*, 2000.
18. A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology - CRYPTO '84*, volume 0193 of *LNCS*, pages 47–53. Springer-Verlag, 1984.
19. T. H. Yuen and V. K. Wei. Fast and proven secure blind identity-based signcryption from pairings. Cryptology ePrint Archive, Report 2004/121, 2004. `http://eprint.iacr.org/`.
20. Y. Zheng. Digital signcryption or how to achieve cost(signature & encryption) $<<$ cost(signature) + cost(encryption). In *Advances in Cryptology - CRYPTO '97*, volume 1294 of *LNCS*, pages 165–179. Springer-Verlag, 1997.