Provably Secure ID-based Broadcast Signcryption (IBBSC) Scheme

S. Sharmila Deva Selvi and S. Sree Vivek and Ragavendran Gopalakrishnan and Naga Naresh Karuturi and C. Pandu Rangan

Department of Computer Science and Engineering
Indian Institute of Technology

Abstract. With the advent of mobile and portable devices such as cell phones and PDAs, wireless content distribution has become a major means of communications and entertainment. In such applications, a central authority needs to deliver encrypted data to a large number of recipients in such a way that only a privileged subset of users can decrypt it. A broadcasting news channel may face this problem, for example, when a large number of people subscribe to a daily exclusive news feature. This is exactly the kind of problem that *broadcast encryption* attempts to efficiently solve. On top of this, especially in the current digital era, junk content or spam is a major turn off in almost every Internet application. If all the users who subscribe to the news feed receive meaningless noise or any unwanted content, then the broadcaster is going to lose them. This results in the additional requirement that subscribers have source authentication with respect to their broadcaster. *Broadcast signcryption*, which enables the broadcaster to simultaneously encrypt and sign the content meant for a specific set of users in a single logical step, provides the most efficient solution to the dual problem of confidentiality and authentication. Efficiency is a major concern, because mobile devices have limited memory and computational power and wireless bandwidth is an extremely costly resource. While several alternatives exist in implementing broadcast signcryption schemes, identity-based (ID-based) schemes are arguably the best suited because of the unique advantage that they provide — any unique, publicly available parameter of a user can be his public key, which eliminates the need for a complex public key infrastructure. In ASIAN 2004, Mu et al. [25] propose what they call an ID-based authenticated broadcast encryption scheme, which is also a broadcast signcryption scheme, as the security goals are the same. They claim that their scheme provides message authentication and confidentiality and formally prove that the broadcaster's secret is not compromised, but in this paper, we demonstrate that even without knowing the broadcaster's secret, it is possible for a legal user to impersonate the broadcaster. We demonstrate this by mounting a universal forgeability attack — any valid user, on receiving and decrypting a valid ciphertext from a broadcaster, can generate a valid ciphertext on any message on behalf of that broadcaster for the same set of legal receivers to which the broadcaster signcrypted the earlier message, without knowing any secrets. Following this, we propose a new ID-based broadcast signcryption (IBBSC) scheme, and formally prove its security under the strongest existing security models for broadcast signcryption (IND-CCA2 and EUF-CMA2).

Keywords. Signcryption, Cryptanalysis, ID-based Cryptosystem, Broadcast Encryption, Provable Security, Random Oracle, Bilinear Pairing.

1 Introduction

With the advent of mobile and portable devices such as cell phones and PDAs used in wireless networks, accessing multimedia content through these devices in the wireless network is increasingly popular. On the other hand, a wireless network is much easier to eavesdrop than a wired network. Therefore, the need to securely deliver multimedia content to the user over a wireless network is becoming more important and critical. Furthermore, wireless communication is a good way to broadcast messages to many users in one go. The most efficient way to broadcast information securely is through a *broadcast encryption scheme* in which a broadcaster can send secure information to dynamic selective recipients such that no other recipients outside the set could recover the secret information. Normally, a broadcast encryption scheme is used to distribute a session key to many users and the intended users would be able to recover the session key, which is used to decrypt encrypted multimedia content sent by a broadcaster.

In many applications, it is also desirable that the users have source authentication with respect to their broadcaster. This results in the need for *authenticated broadcast encryption*, otherwise known as *broadcast signcryption*. The efficiency of a broadcast signcryption scheme is mainly measured by three parameters — length of transmission messages, storage cost, and computational overhead at a user device. All these parameters are extremely important to mobile devices as they have limited memory and computational power as compared to a personal computer, and wireless bandwidth is extremely costly. Identity-based (ID-based) schemes are the most suited for meeting these restrictions, because of the unique advantage that they provide — the public key of a user can be computed from any publicly available parameter of that user that is unique to him, thereby eliminating the complex public key infrastructure that would otherwise have to be employed.

Our Contribution. We give the general framework of an IBBSC scheme and define the formal security models for confidentiality and authentication for IBBSC (which we call IND-IBBSC-CCA2 and EUF-IBBSC-CMA2 respectively). In ASIAN 2004, Mu et al. [25] propose an ID-based broadcast signcryption scheme[1], which they claim provides message authentication and confidentiality. They have proven

[1] Though they call their scheme as an authenticated broadcast encryption scheme, it achieves the same security goals as broadcast signcryption and hence, their scheme is also a broadcast signcryption scheme. The term ‘broadcast signcryption’ was coined much later by Fagen Li et al. [21].

their scheme secure against two types of attacks called *insider attack* and *outsider attack*. In the former, they show that a legal user of the system cannot find the secret key of the broadcaster and in the latter, they show that an external adversary cannot decrypt and recover the encrypted message. Coming to authentication, though they claim their scheme to be unforgeable, they do not prove it formally. In this paper, we demonstrate a universal forgeability attack on their scheme — any legal user, on receiving and decrypting a valid ciphertext from a broadcaster, can generate a valid ciphertext on any message on behalf of that broadcaster for the same set of legal receivers to which the broadcaster signcrypted the earlier message, without knowing any secrets. Following this, we propose a new ID-based broadcast signcryption (IBBSC) scheme, and formally prove its security (confidentiality and unforgeability) under the strongest existing security models for broadcast signcryption (IND-CCA2 and EUF-CMA2 respectively).

Organization. The rest of this paper is organized as follows. In Section 2, we review the underlying cryptographic concepts that are involved, like ID-basaed cryptography, signcryption, broadcast encryption, bilinear pairings and related computational problems, the general framework of ID-based broadcast signcryption (IBBSC) schemes and the formal security models for IBBSC. Next, in Section 3, we review the ID-based authenticated broadcast encryption scheme of Mu et al. [25]. We present our insider universal forgeability attack on this scheme in Section 4. Following this, in Section 5, we lay out the details of our proposed IBBSC scheme, following the general framework of IBBSC schemes. In Sections 6, 7 and 8, we present the formal proofs of correctness, confidentiality and unforgeability of our scheme in the strongest existing security models for IBBSC. In Section 9, we discuss the efficiency of our scheme, following which, in Section 10, we provide a small extension of our scheme. Finally, in Section 11, we conclude the discussion and pose some interesting open problems.

2 Preliminaries

2.1 Identity-based Cryptography

The concept of an Identity-based (ID-based) cryptosystem was introduced by Shamir in 1984 [29]. The distinguishing characteristic of *ID-based cryptography* is the ability to use any string as a public key. In particular, this string maybe the email address, telephone number, or any publicly available parameter of a user that is unique to him. The corresponding private key can only be derived by a trusted Private Key Generator (PKG) who keeps a master secret which is involved in deriving the private keys. Though Shamir proposed an ID-based signature scheme in [29], he left open the quest for an ID-based encryption scheme (IBE). Several schemes that were proposed later were unsatisfactory in different aspects. The first practical IBE scheme was introduced by Boneh and Franklin in 2001 [7]. Since 2001, several schemes have been introduced [13, 30, 10, 6, 5, 4, 17].

There are several advantages offered by ID-based cryptography. If there are only a finite number of users, after all users have been issued with keys, the master secret key can be destroyed. This can take place because in the basic ID-based cryptosystem, keys once issued are always valid. Also, as public keys are derived from identities, IBE eliminates the need for a public key distribution infrastructure (PKI). The authenticity of the public keys is guaranteed implicitly as long as the transport of the private keys to the corresponding user is kept secure.

2.2 Signcryption

To avoid forgery and ensure confidentiality of the contents of a letter, for centuries it has been a common practice for the sender of the letter to sign his name on it and then seal it in an envelope, before handing it over to a deliverer. Public key cryptography, discovered nearly three decades ago, has revolutionized the way for people to conduct secure and authenticated communications. It is now possible for people who have never met before to communicate with one another in a secure and authenticated way over an open and insecure network such as the Internet. In doing so, this same two-step approach has been followed. Namely, before a message is sent out, the sender of the message would sign it using a digital signature scheme, and then encrypt the message (and the signature) using a private key encryption algorithm under a randomly chosen message encryption key. The random message encryption key would then be encrypted using the recipient's public key. This traditional two-step approach is called *signature-then-encryption*.

Signature generation and encryption consume machine cycles, and also introduce 'expanded' bits to an original message. Symmetrically, a comparable amount of computation time is generally required for signature verification and decryption. Hence, the cost of a cryptographic operation on a message is typically measured in the message expansion rate and the computational time invested by both the sender and the recipient. With the standard *signature-then-encryption* approach, the cost for delivering a message in a secure and authenticated way is essentially the sum of the cost for digital signature and that for encryption.

In 1997 [32], Yuliang Zheng presented these concerns and raised an important question as to whether it is possible to transfer a message of arbitrary length in a *secure* and *authenticated* way with an expense less than that required by the *signature-then-encryption* approach. Zheng presented a positive answer to this question, which led to the dawn of *signcryption* which simultaneously fulfills both the functions of digital signature and public key encryption in a logically single step, and with a cost *significantly* smaller than that required by *signature-then-encryption*. He proceeds to define a *signcryption scheme* as consisting of a pair of (polynomial time) algorithms (S, U), where S is called the *signcryption algorithm* and U is called the *unsigncryption algorithm* (also most commonly known as *designcryption algorithm*). S in general is probabilistic, while U is most likely to be deterministic. (S, U) satisfy the following conditions.

1. ***Unique Unsigncryptability.*** Given a message m of arbitrary length, the algorithm S signcrypts m and outputs a signcrypted text c. On input c, the algorithm U unsigncrypts c and recovers the original message unambiguously.
2. ***Security.*** (S, U) fulfill, simultaneously, the properties of a secure encryption scheme and those of a secure digital signature scheme. These properties mainly include *confidentiality* of message contents, *unforgeability*, and *non-repudiation*.
3. ***Efficiency.*** The computational cost, which includes the computational time involved both in signcryption and unsigncryption, and the communication overhead or added redundant bits, of the scheme is smaller than that required by the best currently known *signature-then-encryption* scheme with comparable parameters.

Zheng's discovery went on to revolutionize the cryptographic research community and in a short span of a decade, *signcryption* has become an exploding research area. Since 1997, several efficient signcryption schemes have been proposed [2, 33, 16, 28, 26, 20, 31, 24]. The first example of formal security proof in a formal security model was published in 2002 [1]. However, none of these schemes were ID-based. Malone-Lee [23] proposed the first method that achieved ID-based signcryption. Libert and Quisquater [22] pointed out that [23] is not semantically secure because the signature of the message is visible in the signcrypted message. Chow et al. [12] designed an ID-based signcryption scheme that provides both public verifiability and forward security. Boyen [9] presented an ID-based signcryption scheme that provides not only public verifiability and forward security but also ciphertext unlinkability and anonymity. In [11], Chen and Malone-Lee improved the scheme of Boyen [9] in terms of efficiency. In [3], Barreto et al. constructed the most efficient ID-based signcryption scheme to date.

2.3 Broadcast Encryption

Consider a scenario where there is a center and a set of users. The center provides the users with prearranged keys when they join the system. At some point the center wishes to broadcast a message (e.g., a key to decipher a video clip) to a *dynamically changing* privileged subset of the users in such a way that non-members of the privileged class cannot learn the message. Naturally, the non-members are curious about the contents of the message that is being broadcast, and may try to learn it.

The obvious solution is to give every user his own key and transmit an individually encrypted message to every member of the privileged class. This requires a very long transmission (the number of members in the class times the length of the message). Another simple solution is to provide every possible subset of users with a key, that is, give every user the keys corresponding to the subsets it belongs to. This requires every user to store a huge number of keys.

Amos Fiat and Moni Naor, in 1993 [15], analyzed this problem and proposed a solution which results in efficiency in both measures — transmission length and storage at the users end, without compromising the computational efficiency involved in carrying out the scheme. Their framework is called *broadcast encryption*. Apart from the normal security requirements of a two-party cryptosystem where there is one sender and one receiver, an additional property that is desired from any secure broadcast encryption scheme is *collusion resistance*. This means that even if all the non-privileged users collude in an attempt to learn the plaintext, they should not be able to do so.

Since its introduction by Fiat and Naor [15], the problem received significant attention, and many of its variants have been studied; many broadcast encryption systems have been proposed [27, 19, 18, 8, 14]. The best known fully collusion resistant systems are the schemes of Boneh, Gentry and Waters [8] which achieve $O(\sqrt{n})$-size ciphertexts and public key; or, constant size ciphertexts, $O(n)$-size public key and constant size private keys.

2.4 Bilinear Pairing

Let $\mathbb{G}_1$ be an additive cyclic group generated by P, with prime order q, and $\mathbb{G}_2$ be a multiplicative cyclic group of the same order q. A bilinear pairing is a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ with the following properties.

- ***Bilinearity.*** For all $P, Q, R \in \mathbb{G}_1$,
  - $\hat{e}(P + Q, R) = \hat{e}(P, R)\hat{e}(Q, R)$
  - $\hat{e}(P, Q + R) = \hat{e}(P, Q)\hat{e}(P, R)$
  - $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$
- ***Non-Degeneracy.*** There exist $P, Q \in \mathbb{G}_1$ such that $\hat{e}(P, Q) \neq I_{\mathbb{G}_2}$, where $I_{\mathbb{G}_2}$ is the identity element of $\mathbb{G}_2$.
- ***Computability.*** There exists an efficient algorithm to compute $\hat{e}(P, Q)$ for all $P, Q \in \mathbb{G}_1$.

2.5 Computational Assumptions

In this section, we review the computational assumptions related to bilinear maps that are relevant to the protocol we discuss.

Bilinear Diffie-Hellman Problem (BDHP) Given $(P, aP, bP, cP) \in \mathbb{G}_1^4$ for unknown $a, b, c \in \mathbb{Z}_q^*$, the BDH problem in $\mathbb{G}_1$ is to compute $\hat{e}(P, P)^{abc}$.

Definition. The advantage of any probabilistic polynomial time algorithm $\mathcal{A}$ in solving the BDH problem in $\mathbb{G}_1$ is defined as

$$Adv_{\mathcal{A}}^{BDH} = Pr\left[\mathcal{A}(P, aP, bP, cP) = \hat{e}(P, P)^{abc} \mid a, b, c \in \mathbb{Z}_q^*\right]$$

The *BDH Assumption* is that, for any probabilistic polynomial time algorithm $\mathcal{A}$, the advantage $Adv_{\mathcal{A}}^{BDH}$ is negligibly small.

Decisional Bilinear Diffie-Hellman Problem (DBDHP) Given $(P, aP, bP, cP, \alpha) \in \mathbb{G}_1^4 \times \mathbb{G}_2$ for unknown $a, b, c \in \mathbb{Z}_q^*$, the DBDH problem in $\mathbb{G}_1$ is to decide if $\alpha = \hat{e}(P, P)^{abc}$.

Definition. The advantage of any probabilistic polynomial time algorithm $\mathcal{A}$ in solving the DBDH problem in $\mathbb{G}_1$ is defined as

$$Adv_{\mathcal{A}}^{DBDH} = \left| Pr\left[\mathcal{A}(P, aP, bP, cP, \hat{e}(P, P)^{abc}) = 1\right] - Pr\left[\mathcal{A}(P, aP, bP, cP, \alpha) = 1\right] \right|$$

The *DBDH Assumption* is that, for any probabilistic polynomial time algorithm $\mathcal{A}$, the advantage $Adv_{\mathcal{A}}^{DBDH}$ is negligibly small.

Computational Diffie-Hellman Problem (CDHP) Given $(P, aP, bP) \in \mathbb{G}_1^3$ for unknown $a, b \in \mathbb{Z}_q^*$, the CDH problem in $\mathbb{G}_1$ is to compute abP.

Definition. The advantage of any probabilistic polynomial time algorithm $\mathcal{A}$ in solving the CDH problem in $\mathbb{G}_1$ is defined as

$$Adv_{\mathcal{A}}^{CDH} = Pr\left[\mathcal{A}(P, aP, bP) = abP \mid a, b \in \mathbb{Z}_q^*\right]$$

The *CDH Assumption* is that, for any probabilistic polynomial time algorithm $\mathcal{A}$, the advantage $Adv_{\mathcal{A}}^{CDH}$ is negligibly small.

2.6 Framework of ID-based Broadcast Signcryption (IBBSC)

A generic ID-based broadcast signcryption scheme for sending a single message from a broadcaster to t users consists of the following probabilistic polynomial time algorithms.

1. ***Setup***(k)***.*** Given a security parameter k, the Private Key Generator (PKG) generates the public parameters *params* and master secret key *msk* of the system.
2. ***Keygen***(ID_A)***.*** Given an identity ID_A, the PKG, using the public parameters *params* and the master secret key *msk*, computes the corresponding private key S_A and transmits it to A in a secure way.
3. ***Signcrypt***$(m, ID_A, \mathcal{L} = \{ID_1, ID_2, \ldots, ID_t\}, S_A)$***.*** To send a message m to t users with identities $(ID_1, ID_2, \ldots, ID_t)$, the broadcaster A with identity ID_A and private key S_A runs this algorithm to obtain the signcrypted ciphertext σ.
4. ***Designcrypt***$(\sigma, ID_A, ID_i, S_i)$***.*** When user i with identity ID_i and private key S_i receives the signcrypted ciphertext σ from his broadcaster A with identity ID_A, he runs this algorithm to obtain either the plain text m or $\perp$ according as whether σ was a valid signcryption from identity ID_A to identity ID_i or not.

For consistency, we require that if $\sigma = Signcrypt\,(m, ID_A, (ID_1, ID_2, \ldots, ID_t), S_A)$, then $m = Designcrypt\,(\sigma, ID_A, ID_i, S_i)$ for all $1 \leq i \leq t$.

2.7 Security Model for ID-based Broadcast Signcryption

The two security properties that are desired out of any IBBSC scheme are *message confidentiality* and *unforgeability.* We formally extend the existing strongest security notions for encryption and digital signatures (IND-CCA2 and IND-CMA2 respectively) to IBBSC below.

Indistinguishability under Adaptive Chosen Ciphertext Attack for IBBSC (IND-IBBSC-CCA2) An ID-based broadcast signcryption scheme is semantically secure against adaptive chosen ciphertext attack (IND-IBBSC-CCA2) if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game.

1. The challenger $\mathcal{C}$ runs $Setup(k)$ and sends the system public parameters $params$ to the adversary $\mathcal{A}$.
2. In the first phase, $\mathcal{A}$ makes polynomially bounded number of queries to the following oracles.
   (a) **Keygen Oracle** — $\mathcal{A}$ produces an identity ID and queries for the secret key of ID. The *Keygen Oracle* returns S_{ID} to $\mathcal{A}$.
   (b) **Signcrypt Oracle** — $\mathcal{A}$ produces a message m, broadcaster identity ID_A and a list of receiver identities $ID_1, ID_2, \ldots, ID_t$. $\mathcal{C}$ returns $\sigma = Signcrypt\,(m, ID_A, \{ID_1, ID_2, \ldots, ID_t\}, S_A)$, the signcrypted ciphertext, to $\mathcal{A}$, where the secret key S_A is computed from $Keygen(ID_A)$.
   (c) **Designcrypt Oracle** — $\mathcal{A}$ produces a broadcaster identity ID_A, receiver identity ID_i and a signcryption σ. The challenger $\mathcal{C}$ returns to $\mathcal{A}$, the result of $Designcrypt\,(\sigma, ID_A, ID_i, S_i)$, where the secret key S_i is computed from $Keygen(ID_i)$. The result returned is $\perp$ if σ is an invalid signcrypted ciphertext from ID_A to ID_i.
3. $\mathcal{A}$ produces two messages m_0 and m_1 of equal length from the message space $\mathcal{M}$, an arbitrary broadcaster identity ID_A, and the set of identities of the receivers of that broadcaster $\mathcal{L} = \{ID_1, ID_2, \ldots, ID_t\}$. The adversary must not have queried any of the t receivers' secret keys. The challenger $\mathcal{C}$ flips a coin, sampling a bit $b \leftarrow \{0, 1\}$ and obtains the challenge signcrypted ciphertext by running $Signcrypt\,(m_b, ID_A, \{ID_1, ID_2, \ldots, ID_t\}, S_A)$, which is returned to $\mathcal{A}$.
4. $\mathcal{A}$ is allowed to make polynomially bounded number of new queries as in Step 2 with the restrictions that it should not query the *Designcryption Oracle* for the designcryption of σ^* or the *Keygen Oracle* for the secret keys of $ID_1, ID_2, \ldots, ID_t$.
5. Finally, $\mathcal{A}$ outputs a bit b' and wins the game if $b' = b$.

We mention that this model of security takes into account collusion resistance too, because we provide the adversary with the secret keys of every user of the system except the ones he attacks.

Existential Unforgeability under Adaptive Chosen Message Attack for IBBSC (EUF-IBBSC-CMA2) An ID-based broadcast signcryption scheme is existentially unforgeable under adaptive chosen message attack (EUF-IBBSC-CMA2) if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game.

1. The challenger $\mathcal{C}$ runs *Setup*(k) and sends the system public parameters *params* to the adversary $\mathcal{A}$.
2. In the first phase, $\mathcal{A}$ makes polynomially bounded number of queries to the following oracles.
   (a) **Keygen Oracle** — $\mathcal{A}$ produces an identity ID and queries for the secret key of ID. The *Keygen Oracle* returns S_{ID} to $\mathcal{A}$.
   (b) **Signcrypt Oracle** — $\mathcal{A}$ produces a message m, broadcaster identity ID_A and a list of receiver identities $ID_1, ID_2, \ldots, ID_t$. $\mathcal{C}$ returns $\sigma = Signcrypt\left(m, ID_A, \{ID_1, ID_2, \ldots, ID_t\}, S_A\right)$, the signcrypted ciphertext, to $\mathcal{A}$, where the secret key S_A is computed from $Keygen(ID_A)$.
   (c) **Designcrypt Oracle** — $\mathcal{A}$ produces a broadcaster identity ID_A, receiver identity ID_i and a signcryption σ. The challenger $\mathcal{C}$ returns to $\mathcal{A}$, the result of $Designcrypt\left(\sigma, ID_A, ID_i, S_i\right)$, where the secret key S_i is computed from $Keygen(ID_i)$. The result returned is $\perp$ if σ is an invalid signcrypted ciphertext from ID_A to ID_i.
3. $\mathcal{A}$ produces a signcrypted ciphertext σ from an arbitrary broadcaster ID_A to the list of his receivers $\mathcal{L}$ and wins the game if the private key of broadcaster ID_A was not queried and $\perp$ is not returned by $Designcrypt(\sigma, ID_A, ID_i, S_i)$ for any $ID_i \in \mathcal{L}$ and σ is not the output of a previous query to the *Signcrypt Oracle*.

We mention that this model of security takes into account collusion resistance too, because we allow the adversary to query for the secret keys of any entity.

3 Review of Mu's Scheme

Mu et al.'s IBBSC scheme [25] consists of the three algorithms *Keygen* (which includes *Setup* as well), *Encrypt* and *Decrypt*, which we describe below.
KeyGen(k, n_b, n)**.** Here, k is a security parameter, n_b is the number of broadcasters and n is the number of users in the system.

1. **Broadcaster Setup**
   (a) Select master private keys $s_i \in \mathbb{Z}_q$ for all broadcasters B_i $(i = 1, \ldots, n_b)$.
   (b) Select three strong public one-way hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1, H_2 : \{0,1\}^* \to \{0,1\}^k, H_3 : \{0,1\}^* \to \mathbb{Z}_q$.
   (c) Extract the public keys $Q_{B_i} \leftarrow H_1(ID_{B_i})$, where ID_{B_i} $(i = 1, \ldots, n_b)$ are the unique identifiers of broadcasters.
   (d) Compute the private keys of the broadcasters $S_{B_i} \leftarrow s_i Q_{B_i}$ and $\bar{S}_{B_i} \leftarrow s_i P$.
2. **User Setup**

(a) Select $x_i \in \mathbb{Z}_q, i = 1, \cdots, n$ and assign each of $\{x_i\}$ to a user. Assign $\{x_i\}_{1 \le i \le n}$ to all broadcasters.

Signcrypt$(\mathcal{L}, ID_{B_i}, m)$**.** Here, $\mathcal{L} = \{1, 2, \ldots, t\}$ where $t \le n$ is the number of privileged users to whom broadcaster B_i wishes to send a message $m \in \mathbb{Z}_q$. Without loss of generality, we have assumed it is the first t members that are privileged.

1. Compute the following.
   (a) $\prod_{j=1}^{t}(x - x_j)$, generating a polynomial function $f(x) = \sum_{\ell=0}^{t} c_\ell x^\ell$
   (b) $P_0 \leftarrow r(c_0 P + \bar{S}_{B_i}), P_1 \leftarrow c_1 r P, \ldots, P_t \leftarrow c_t r P$
   (c) $k \leftarrow \hat{e}(P, r^2 S_{B_i})$
   (d) $y \leftarrow m \oplus H_3(k)$
   (e) $X \leftarrow r(r - H_2(y)) S_{B_i}$
2. Broadcast $(y, X, ID_{B_i}, P_0, \ldots, P_t)$

Designcrypt$(y, X, ID_{B_i}, ID_j, x_j, P_0, \ldots, P_t)$**.** Here, $(y, X, ID_{B_i}, P_0, \ldots, P_t)$ is the ciphertext received by a member with identity ID_j whose secret value is x_j.

1. Compute $D \leftarrow \sum_{\ell=0}^{t} x_j^\ell P_\ell$.
2. Compute $k \leftarrow \hat{e}(P, X) \cdot \hat{e}(D\ ,\ H_2(y) Q_{B_i})$.
3. Compute $m \leftarrow H_3(k) \oplus y$.

4 Attack on Mu's Scheme

Mu et al. claimed that their scheme provides both confidentiality and unforgeability. They prove the former rigorously, but do not give the formal proof for unforgeability. We show in this section that their scheme is universally forgeable which is a major attack. Once a legitimate user gets a ciphertext from the broadcaster (intended for a set of users) and decrypts it (being a member of the intended set), he can generate a valid ciphertext for any message m^* as if it were generated by the broadcaster for the same set of users. We describe how this attack proceeds in this section.

Let *Alice* be a broadcaster with identity ID_{B_i} of the system and *Eve* be any legitimate user. *Eve* has just received a ciphertext, say $(y, X, ID_{B_i}, P_0, \cdots, P_t)$ and decrypts it (we assume that *Eve* is present in the set $\mathcal{L} = \{1, 2, \ldots, t\}$). If *Eve* wants to generate the ciphertext of any message m^* as if it were generated by *Alice* for the same list $\mathcal{L}$, with identities $ID_1, ID_2, \ldots, ID_t$, *Eve* just has to do the following.

1. As a result of decrypting the ciphertext, *Eve* gets the value of $D = r\bar{S}_{B_i}$.
2. Choose $r^* \in_R \mathbb{Z}_q^*$ and compute the following.
   (a) $P^* = P_0 - D + r^* P = rC_0 p + r^* P$
   (b) $k^* = \hat{e}(r^* P, P)$
   (c) $y^* = m^* \oplus H_3(k)$

(d) $X^* = r^*P - r^*H_2(y)Q_{B_i}$

3. $(y^*, X^*, ID_{B_i}, P_0^*, P_1, P_2, \ldots, P_t)$ is now a valid ciphertext of *Alice* for the message m^* generated by *Eve* for the list of users $\mathcal{L}$ with identities $\{ID_j\}_{1 \le j \le t}$

We now prove that the ciphertext generated by *Eve* is indeed a valid ciphertext from *Alice* to the receivers in $\mathcal{L}$ on the message m^*.
Decrypt$(y^*, X^*, ID_{B_i}, ID_j, x_j, P_0^*, \ldots, P_t)$**.** A receiver with identity ID_j uses his secret value x_j to decrypt the ciphertext $(y^*, X^*, ID_{B_i}, P_0^*, P_1, P_2, \ldots, P_t)$ obtained from *Eve* as follows. He computes the following.

1. $D^* \leftarrow \sum_{\ell=0}^{t} x_j^{\ell} P_{\ell} = r^*P$
2. $k^* \leftarrow \hat{e}(P, X^*) \cdot \hat{e}(D^*\ ,\ H_2(y^*)Q_{B_i}) = \hat{e}(r^*P, P)$
3. $m^* \leftarrow H_3(k) \oplus y^*$

From this it is clear that *Eve* can succeed in generating a ciphertext for an arbitrary message m^* with *Alice* as sender and identities ID_j, $1 \le j \le t$ as receivers without knowing the secret key of *Alice* and only knowing a previous valid ciphertext from *Alice* to this set of users and its decryption.

5 New ID-based Broadcast Signcryption Scheme

In this section, we propose our new IBBSC scheme. We follow the framework of a general ID-based broadcast signcryption scheme that we presented in Section 2.3, with a minor modification. Our proposed scheme includes two more algorithms, *Register* and *Unregister*. (Of course, we also add the corresponding oracles to the security models.) As their names suggest, these algorithms specify the steps to be followed when users who wish to subscribe to a particular broadcaster join the system and when those wishing to unsubscribe leave the system respectively. Our scheme uses the same idea of a polynomial built out of secret values of users that was used in the schemes of Mu et al. [25]. We also mention that in our scheme, whenever a broadcaster signcrypts a message, he does not do it for any subset of his subscribers but to all of them. This restriction is easily removed by minor modifications to our scheme, as will be explained later. Moreover, our scheme is customized for the practical use of broadcast signcryption in the real world, where, in the case of satellite TV, there is one central authority and several local broadcasters and users subscribe to a broadcaster closest to them to receive content. It is also reasonable to assume that broadcasters always broadcast content to all their subscribers. We now describe the algorithms that comprise our scheme.
1. *Setup*(k) Let k be the security parameter of the system. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of prime order q (where $|q| = k$) and let P be the generator of $\mathbb{G}_1$ and $\hat{e}$ be a bilinear map defined as $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Let n_0, n_1, n_2 and n_3 denote the number of bits required to represent an identity, an element of $\mathbb{G}_1$, an element of $\mathbb{G}_2$ and a message respectively. Consider four hash functions $H_1 : \{0,1\}^{n_0} \rightarrow \mathbb{G}_1$, $H_2 : \{0,1\}^{n_2} \rightarrow \mathbb{Z}_q^*$, $H_3 : \{0,1\}^{n_0+2n_1+n_3} \rightarrow \mathbb{Z}_q^*$, $H_4 : \{0,1\}^{n_1} \rightarrow$

$\{0,1\}^{2n_1+n_3}$. The PKG chooses its secret key $s \in \mathbb{Z}_q^*$ and sets the public key $P_{pub} = sP$. The public parameters of the system are $\langle \mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, \hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2, H_1, H_2, H_3, H_4 \rangle$.
2. ***Keygen***(ID_A) The public key and private key of broadcaster B_j are computed from his identity ID_{B_j} as $Q_{B_j} = H_1\left(ID_{B_j}\right)$ and $S_{B_j} = sQ_{B_j}$ respectively. Similarly, the public key and private key of user i are computed from his identity ID_i as $Q_i = H_1(ID_i)$ and $S_i = sQ_i$ respectively.
3. ***Register***(ID_B, ID_i) When a member i with identity ID_i is subscribing or registering to a broadcaster B with identity ID_B, he chooses a secret value $y_i \in_R \mathbb{Z}_q^*$ (or uses an existing secret value, if available) and sends securely (y_iQ_i, y_iQ_B) to the broadcaster B. Then, B can verify if the member is actually registering for himself by checking if the tuple (R, T) which he received satisfies $\hat{e}(R, Q_B) \stackrel{?}{=} \hat{e}(T, Q_i)$. The broadcaster B will then compute $\alpha_i = \hat{e}(R, S_B)$ and $x_i = H_2(\alpha_i)$, add the entry (ID_i, x_i) to his subscriber list $\mathcal{L}_B$ and update his subscriber polynomial as $f_B(x) \leftarrow f_B(x) \cdot (x - x_i)$. For all broadcasters B, $f_B(x)$ is initially set to 1.
4. ***Unregister***(ID_B, ID_i) When a member i with identity ID_i is unsubscribing or unregistering from a broadcaster B with identity ID_B, he sends an unsubscribe request to the broadcaster. The broadcaster will then look up ID_i in his subscriber list $\mathcal{L}_B$ and update his subscriber polynomial as $f_B(x) \leftarrow f_B(x) \cdot (x - x_i)^{-1}$. He also removes the entry (ID_i, x_i) from the list $\mathcal{L}_B$.
5. ***Signcrypt***$(m, ID_{B_j}, f_B(x), S_{B_j})$ When broadcaster B wants to send a message m to his subscribers $1, 2, \ldots, t$, he does the following.

1. Choose a random $r \in_R \mathbb{Z}_q^*$ and compute $P_1 = rc_1P$, $P_2 = rc_2P$, $\ldots$, $P_t = rc_tP$, where c_i is the coefficient of x^i in $f_B(x)$.
2. Compute the following.
   (a) $X = rQ_B$
   (b) $h_3 = H_3(ID_B \| P_1 \| X \| m)$
   (c) $Z = (r + h_3) S_B$
   (d) $\beta = \hat{e}(Z, P)$
   (e) $U = H_2(\beta) P$
   (f) $P_0 = rc_0P + U$, where c_0 is the constant term of the polynomial $f_B(x)$
   (g) $y = (m \| X \| Z) \oplus H_4(U)$
3. Broadcast the signcryption $\sigma_B = (ID_B, y, P_0, P_1, \ldots, P_t)$.

5. ***Designcrypt***(σ_B, ID_i, x_i) To designcrypt an incoming signcryption σ_B, the member with identity ID_i uses his precomputed secret $x_i = H_2(\alpha_i)$, where $\alpha_i = \hat{e}(y_iQ_B, S_i)$. Here, S_i is the secret key of this member and y_i is his secret value that is known only to him. He does the following.

1. Compute $U' = \sum\limits_{j=0}^{t} x_i^j P_j$.
2. Retrieve $m' \| X' \| Z' = y \oplus H_4(U')$.
3. Compute $\beta' = \hat{e}(Z', P)$.
4. Check if $U' \stackrel{?}{=} H_2(\beta') P$.

5. Check if $\beta' \stackrel{?}{=} \hat{e}\left(X' + h_3' Q_B, sP\right)$, where $h_3' = H_3\left(ID_B \| P_1 \| X' \| m'\right)$.
6. If the checks succeed, return m'. Else, return $\perp$.

In the forthcoming sections, we formally prove the correctness, confidentiality and unforgeability of our scheme.

6 Correctness of our IBBSC Scheme

In this section, we proceed to prove that our proposed scheme is indeed consistent and correct. If $\sigma_B = (ID_B, y, P_0, P_1, \ldots, P_t)$ is a valid signcrypted ciphertext from broadcaster B to his subscribers with identities $ID_1, ID_2, \ldots, ID_t$, then ***Designcrypt***(σ_B, ID_i, x_i) will do the following. Here, ID_i is the identity of the receiver who invokes the *Designcrypt* algorithm with his precomputed secret $x_i = H_2\left(\alpha_i\right)$, where $\alpha_i = \hat{e}\left(y_i Q_B, S_i\right)$ and S_i is the secret key of this member and y_i is his secret value that is known only to himself.

1. Compute $U' = \sum_{j=0}^{t} x_i^j P_j = r\left(\sum_{j=0}^{t} c_j x_i^j\right) P + U = U$.
2. Retrieve $m' \| X' \| Z' = y \oplus H_4\left(U'\right) = y \oplus H_4\left(U\right) = m \| X \| Z$.
3. Compute $\beta' = \hat{e}(Z', P) = \hat{e}(Z, P) = \beta$.
4. **Check 1**, $U' \stackrel{?}{=} H_2\left(\beta'\right) P$, succeeds because,
$$U' = U = H_2\left(\beta\right) P = H_2\left(\beta'\right) P$$
5. **Check 2**, $\beta' \stackrel{?}{=} \hat{e}\left(X' + h_3' Q_B, sP\right)$, succeeds because,
$$\begin{aligned} \beta' &= \beta \\ &= \hat{e}(Z, P) \\ &= \hat{e}\left(\left(r + h_3\right) S_B, P\right) \\ &= \hat{e}\left(\left(r + h_3\right) Q_B, sP\right) \\ &= \hat{e}\left(X + h_3 Q_B, sP\right) \\ &= \hat{e}\left(X' + h_3' Q_B, sP\right) \end{aligned}$$
because $X = X'$ and $h_3 = H_3\left(ID_B \| P_1 \| X \| m\right) = H_3\left(ID_B \| P_1 \| X' \| m'\right) = h_3'$ □

7 Proof of Confidentiality of our IBBSC Scheme

Theorem. *Our ID-based broadcast signcryption scheme is secure against any IND-IBBSC-CCA2 adversary $\mathcal{A}$ under the random oracle model if DBDHP is hard in $\mathbb{G}_1$.*

Proof. The challenger $\mathcal{C}$ receives an instance (P, aP, bP, cP, γ) of the DBDH problem. His goal is to determine whether $\gamma = \hat{e}(P, P)^{abc}$. Suppose there exists an IND-IBBSC-CCA2 adversary $\mathcal{A}$ for our proposed scheme. We show that $\mathcal{C}$ can use $\mathcal{A}$ to solve the DBDH problem. $\mathcal{C}$ will set the random oracles $\mathcal{O}_{H_1}$, $\mathcal{O}_{H_2}$, $\mathcal{O}_{H_3}$, $\mathcal{O}_{H_4}$, $\mathcal{O}_{KeyExtract}$, $\mathcal{O}_{Register}$, $\mathcal{O}_{Unregister}$, $\mathcal{O}_{Signcrypt}$ and $\mathcal{O}_{Designcrypt}$. The answers to the oracles $\mathcal{O}_{H_1}$, $\mathcal{O}_{H_2}$, $\mathcal{O}_{H_3}$, and $\mathcal{O}_{H_4}$ are randomly selected; therefore, to maintain consistency, $\mathcal{C}$ will maintain three lists $L_1 = \langle ID_i, Q_i, d_i, \mathcal{S}_i\rangle$, $L_2 = \langle \alpha, x\rangle$, $L_3 = \langle ID_i, P_1, X, m, h_3\rangle$, and $L_4 = \langle U, u\rangle$. The reasons for and meanings of the elements of these lists will become clear during the discussion of the corresponding oracles. We assume that $\mathcal{A}$ will ask for $H_1(ID)$ before ID is used in any key extraction, registration, unregistration, signcryption and designcryption queries. First, the adversary $\mathcal{A}$ outputs a list $\mathcal{L} = \{ID_1, ID_2, \ldots, ID_t\}$ of the members whom he proposes to attack, and the identity ID_B of the broadcaster who signcrypts the message to these members. Then, the challenger $\mathcal{C}$ gives $\mathcal{A}$ the system parameters $params$, consisting of P and $P_{pub} = aP$. The descriptions of the oracles follow.

Oracle $\mathcal{O}_{\mathbf{H_1}}(\mathbf{ID_i})$***.*** $\mathcal{C}$ checks if there exists a tuple $(ID_i, Q_i, d_i, \mathcal{S}_i)$ in L_1. If such a tuple exists, $\mathcal{C}$ answers with Q_i. Otherwise, $\mathcal{C}$ does the following.

1. If $ID_i \notin \mathcal{L}$, choose a new[2] $d_i \in_R \mathbb{Z}_q^*$ and set $Q_i = d_iP$.
2. If $ID_i \in \mathcal{L}$, choose a new $d_i \in_R \mathbb{Z}_q^*$ and set $Q_i = d_ibP$.
3. If ID_i is an identity of a broadcaster, add the tuple $(ID_i, Q_i, d_i, \{1\})$ to L_1 and return Q_i.
4. If ID_i is not an identity of a broadcaster, add the tuple (ID_i, Q_i, d_i, ϕ) to L_1 and return Q_i.

If ID_i is a broadcaster's ID, we use $\mathcal{S}_i$ to denote the set of coefficients of the subscriber polynomial (which is initially just the constant term 1). Otherwise, if it is a member's ID, we use it to store the set of (x_j, ID_{B_j}) values (where x_j is the precomputed secret of the member i and ID_{B_j} is the broadcaster to whom, when registering, $(y_jQ_i, y_jQ_{B_j})$ was sent by the member). Recall that the broadcaster B_j computes $x_j = \hat{e}\left(y_jQ_i, S_{B_j}\right)$.

Oracle $\mathcal{O}_{\mathbf{H_2}}(\alpha)$***.*** $\mathcal{C}$ checks if there exists a tuple (α, h_2) in L_2. If such a tuple exists, $\mathcal{C}$ returns h_2. Otherwise, $\mathcal{C}$ chooses a new $h_2 \in_R \mathbb{Z}_q^*$, adds the tuple (α, h_2) to L_2 and returns h_2.

Oracle $\mathcal{O}_{\mathbf{H_3}}(\mathbf{ID_i}\|\mathbf{P_1}\|\mathbf{X}\|\mathbf{m})$***.*** $\mathcal{C}$ checks if there exists a tuple (ID_i, P_1, X, m, h_3) in L_3. If such a tuple exists, $\mathcal{C}$ returns h_3. Otherwise, $\mathcal{C}$ chooses a new $h_3 \in_R \mathbb{Z}_q^*$, adds the tuple (ID_i, P_1, X, m, h_3) to L_3 and returns h_3.

Oracle $\mathcal{O}_{\mathbf{H_4}}(\mathbf{U})$***.*** $\mathcal{C}$ checks if there exists a tuple (U, h_4) in L_4. If such a tuple exists, $\mathcal{C}$ returns h_4. Otherwise, $\mathcal{C}$ chooses a new $h_4 \in_R \{0,1\}^{2n_1+n_3}$, adds the tuple (U, h_4) to L_4 and returns h_4.

Oracle $\mathcal{O}_{\mathbf{KeyExtract}}(\mathbf{ID_i})$***.*** If L_1 does not contain an entry for ID_i, return $\perp$. Otherwise, $\mathcal{C}$ does the following.

1. If $ID_i \in \mathcal{L}$, return $\perp$.
2. If $ID_i \notin \mathcal{L}$, recover the tuple $(ID_i, Q_i, d_i, \mathcal{S}_i)$ from L_1 and return $S_i = d_iP_{pub} = aQ_i$.

[2] By new, we mean that the random value chosen must not have been already chosen during an earlier execution.

Oracle $\mathcal{O}_{\mathbf{Register}}(\mathbf{ID_{B_j}}, \mathbf{ID_i}, \mathbf{y_iQ_i}, \mathbf{y_iQ_{B_j}})$. If L_1 does not contain an entry for ID_i or ID_{B_j}, then abort. Otherwise, consider the following two cases.
First, when $ID_i \notin \mathcal{L}$, $\mathcal{C}$ checks if $\hat{e}\left(y_iQ_i, Q_{B_j}\right) = \hat{e}\left(Q_i, y_iQ_{B_j}\right)$. If not, then abort. Otherwise, it does the following.

1. Update the tuple $(ID_i, Q_i, d_i, \mathcal{S}_i)$ in L_1 by setting $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \left\{\left(x_i, ID_{B_j}\right)\right\}$, where $x_i = H_2(\alpha_i)$, where $\alpha_i = \hat{e}\left(y_iQ_i, S_{B_j}\right)$, where the broadcaster's secret key is obtained from $\mathcal{O}_{\mathbf{KeyExtract}}(\mathbf{ID_{B_j}})$.
2. Retrieve the tuple $\left(ID_{B_j}, Q_{B_j}, d_{B_j}, \mathcal{S}_{B_j}\right)$ from L_1, where $\mathcal{S}_{B_j} = \{c_0, c_1, \ldots, c_t\}$. Construct the new subscriber polynomial as $f_{B_j}(x) = \left(\sum_{\ell=0}^{t} c_\ell x^\ell\right) \cdot (x - x_i)$. Let the set of new coefficients be $\mathcal{S}'_{B_j} = \left\{c'_0, c'_1, \ldots, c'_{t+1}\right\}$. Update this tuple in L_1 by replacing $\mathcal{S}_{B_j}$ with $\mathcal{S}'_{B_j}$.

Second, if $ID_i \in \mathcal{L}$, the adversary should not be allowed to register the member because then he'll trivially have all the information he needs to designcrypt the signcryption[3]. So, the oracle ignores the last two parameters. It instead retrieves the tuples $(ID_i, Q_i, d_i, \mathcal{S}_i)$ and $(ID_B, Q_B, d_B, \mathcal{S}_B)$ from L_1 and, takes[4] y_i to be $y_i = c \cdot d_i^{-1} \cdot d_B^{-1}$ and executes the same two steps above, with a minor change in the first step. To be clear, $\mathcal{C}$ does the following.

1. Update the tuple $(ID_i, Q_i, d_i, \mathcal{S}_i)$ in L_1 by setting $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \left\{\left(x_i, ID_{B_j}\right)\right\}$, where $x_i = H_2(\gamma)$.
2. Perform Step 2 exactly as in the previous case.

Oracle $\mathcal{O}_{\mathbf{Unregister}}(\mathbf{ID_{B_j}}, \mathbf{ID_i})$. If L_1 does not contain an entry for ID_i or ID_{B_j}, then abort. Otherwise, $\mathcal{C}$ does the following.

1. Update the tuple $(ID_i, Q_i, d_i, \mathcal{S}_i)$ in L_1 by setting $\mathcal{S}_i \leftarrow \mathcal{S}_i - \left\{\left(x_i, ID_{B_j}\right)\right\}$. Temporarily store the value of x_i for use in the next step.
2. Retrieve the tuple $\left(ID_{B_j}, Q_{B_j}, d_{B_j}, \mathcal{S}_{B_j}\right)$ from L_1, where $\mathcal{S}_{B_j} = \{c_0, c_1, \ldots, c_t\}$. Construct the new subscriber polynomial as $f_{B_j}(x) = \left(\sum_{\ell=0}^{t} c_\ell x^\ell\right) \cdot (x - x_i)^{-1}$. Let the set of new coefficients be $\mathcal{S}'_{B_j} = \left\{c'_0, c'_1, \ldots, c'_{t-1}\right\}$. Update this tuple in L_1 by replacing $\mathcal{S}_{B_j}$ with $\mathcal{S}'_{B_j}$.

Oracle $\mathcal{O}_{\mathbf{Signcrypt}}\left(\mathbf{m}, \mathbf{ID_{B_j}}\right)$. On receiving this query, $\mathcal{C}$ checks if there is an entry for ID_{B_j} in L_1 and if the set $\mathcal{S}_{B_j}$ is not singleton. If one or both of these conditions are not satisfied, then $\mathcal{C}$ aborts. Otherwise, $\mathcal{C}$ retrieves the tuple $\left(ID_{B_j}, Q_{B_j}, d_{B_j}, \mathcal{S}_{B_j}\right)$ from L_1, where $\mathcal{S}_{B_j} = \{c_0, c_1, \ldots, c_t\}$, invokes $\mathcal{O}_{\mathbf{KeyExtract}}(\mathbf{ID_{B_j}})$ to compute the secret key S_{B_j} and executes ***Signcrypt***$(m, ID_{B_j}, \mathcal{S}_{B_j}, S_{B_j})$ as usual and returns what the signcryption algorithm returns.

[3] In this game, we are providing the adversary with the secret key of the broadcasters, so even if he does not know the secret key of the subscriber, he can calculate α_i by knowing y_i and the secret key of the broadcaster who sent the signcryption.

[4] Note that the challenger cannot calculate y_i, he just considers y_i to be the value $c \cdot d_i^{-1} \cdot d_B^{-1}$.

Oracle $\mathcal{O}_{\mathbf{Designcrypt}}\left(\sigma_{\mathbf{B_j}}, \mathbf{ID_i}\right)$**.** On receiving the signcryption $\sigma_{B_j} = \left(ID_{B_j}, y, P_0, P_1, \ldots, P_t\right)$, $\mathcal{C}$ first checks if there are entries for ID_{B_j} and ID_i in L_1 and there is a tuple of the form $\left(x_i, ID_{B_j}\right) \in \mathcal{S}_i$. If one or more of these conditions are not satisfied, then $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ executes ***Designcrypt***$(\sigma_{B_j}, ID_i, x_i)$ in the normal way and returns what the designcryption algorithm returns.

After the first query stage, $\mathcal{A}$ outputs two plaintext messages m_0 and m_1 of equal length. Obviously, at this point, all the subscribers of the broadcaster B must be in $\mathcal{L}$. This is because, in our scheme, the broadcaster *always* signcrypts messages for *all* his subscribers. Even if the adversary knows the secret key and secret value of *one* of the subscribers, it is enough for him to be able to designcrypt it. Now, $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, retrieves the tuple $(ID_B, Q_B, d_B, \mathcal{S}_B)$ from L_1, where $\mathcal{S}_B = \{c_0, c_1, \ldots, c_t\}$, computes the secret key S_B from $\mathcal{O}_{\mathbf{KeyExtract}}(\mathbf{ID_B})$ and executes ***Signcrypt***$(m_b, ID_B, \mathcal{S}_B, S_B)$ as usual and returns what the signcryption algorithm returns as the challenge signcryption.

$\mathcal{A}$ can perform queries as above. However, it cannot query the designcryption oracle with the challenge signcryption. At the end of the simulation, $\mathcal{A}$ outputs a bit b' for which he believes that the challenge signcryption is the signcryption of $m_{b'}$ from ID_B to its subscribers. If the relation $b = b'$ holds, then $\mathcal{C}$ outputs 1 as the answer to the DBDH problem. Otherwise, it outputs 0. Since the adversary is denied access to the designcryption oracle with the challenge signcryption, he can recognize which message was signcrypted by seeing the signcryption alone, only if he has computed U, for which he must have computed the value of α_i for some member i who subscribes to the broadcaster B. This means, the α_i that he computes must be the same as the α_i that was used in the construction of the subscriber polynomial. We have,

$$\begin{aligned}
\gamma &= \hat{e}(y_i' Q_B, S_i) \\
&= \hat{e}(c d_i^{-1} d_B^{-1} \cdot d_B P, d_i abP) \\
&= \hat{e}(c d_i^{-1} P, d_i abP) \\
&= \hat{e}(cP, abP) \\
&= \hat{e}(P, P)^{abc}
\end{aligned}$$

So, if there exists a non-trivial adversary who can defeat the signcryption by learning something about the encrypted message, that means there exists an algorithm to solve the CDH problem with non-negligible advantage. Since this is not possible, no adversary can defeat the signcryption this way. Hence, our proposed scheme is secure against any IND-IBBSC-CCA2 attack. □

8 Proof of Unforgeability of our IBBSC Scheme

Theorem. *Our ID-based broadcast signcryption scheme is secure against any EUF-IBBSC-CMA2 adversary* $\mathcal{A}$ *under the random oracle model if CDHP is hard in* $\mathbb{G}_1$*.*

Proof. The challenger $\mathcal{C}$ receives an instance (P, aP, bP) of the CDH problem. His goal is to determine abP. Suppose there exists an EUF-IBBSC-CMA2 adversary $\mathcal{A}$ for our proposed scheme. We show that $\mathcal{C}$ can use $\mathcal{A}$ to solve the CDH problem. $\mathcal{C}$ will set the random oracles $\mathcal{O}_{H_1}$, $\mathcal{O}_{H_2}$, $\mathcal{O}_{H_3}$, $\mathcal{O}_{H_4}$, $\mathcal{O}_{KeyExtract}$, $\mathcal{O}_{Register}$, $\mathcal{O}_{Unregister}$, $\mathcal{O}_{Signcrypt}$ and $\mathcal{O}_{Designcrypt}$. The answers to the oracles $\mathcal{O}_{H_1}$, $\mathcal{O}_{H_2}$, $\mathcal{O}_{H_3}$, and $\mathcal{O}_{H_4}$ are randomly selected; therefore, to maintain consistency, $\mathcal{C}$ will maintain three lists $L_1 = \langle ID_i, Q_i, d_i, \mathcal{S}_i \rangle$, $L_2 = \langle \alpha, x \rangle$, $L_3 = \langle ID_i, P_1, X, m, h_3 \rangle$, and $L_4 = \langle U, u \rangle$. The reasons for and meanings of the elements of these lists will become clear during the discussion of the corresponding oracles. We assume that $\mathcal{A}$ will ask for $H_1(ID)$ before ID is used in any key extraction, registration, unregistration, signcryption and designcryption queries. First, the adversary $\mathcal{A}$ outputs the identity ID_B of the broadcaster whose signcryption he claims to be able to forge. Then, the challenger $\mathcal{C}$ gives $\mathcal{A}$ the system parameters $params$, consisting of P and $P_{pub} = aP$. The descriptions of the oracles follow.

Oracle $\mathcal{O}_{\mathbf{H_1}}(\mathbf{ID_i})$**.** $\mathcal{C}$ checks if there exists a tuple $(ID_i, Q_i, d_i, \mathcal{S}_i)$ in L_1. If such a tuple exists, $\mathcal{C}$ answers with Q_i. Otherwise, $\mathcal{C}$ does the following.

1. If $ID_i \neq ID_B$, choose a new[5] $d_i \in_R \mathbb{Z}_q^*$ and set $Q_i = d_i P$.
2. If $ID_i = ID_B$, set $d_i = \perp$ and $Q_i = bP$.
3. If ID_i is an identity of a broadcaster, add the tuple $(ID_i, Q_i, d_i, \{1\})$ to L_1 and return Q_i.
4. If ID_i is not an identity of a broadcaster, add the tuple (ID_i, Q_i, d_i, ϕ) to L_1 and return Q_i.

If ID_i is a broadcaster's ID, we use $\mathcal{S}_i$ to denote the set of coefficients of the subscriber polynomial (which is initially just the constant term 1). Otherwise, if it is a member's ID, we use it to store the set of (x_j, ID_{B_j}) values (where x_j is the precomputed secret of the member i and ID_{B_j} is the broadcaster to whom, when registering, the member sent $(y_j Q_i, y_j Q_{B_j})$). Recall that the broadcaster B_j computes $x_j = \hat{e}\left(y_j Q_i, S_{B_j}\right)$.

Oracle $\mathcal{O}_{\mathbf{H_2}}(\alpha)$**.** $\mathcal{C}$ checks if there exists a tuple (α, h_2) in L_2. If such a tuple exists, $\mathcal{C}$ returns h_2. Otherwise, $\mathcal{C}$ chooses a new $h_2 \in_R \mathbb{Z}_q^*$, adds the tuple (α, h_2) to L_2 and returns h_2.

Oracle $\mathcal{O}_{\mathbf{H_3}}(\mathbf{ID_i} \| \mathbf{P_1} \| \mathbf{X} \| \mathbf{m})$**.** $\mathcal{C}$ checks if there exists a tuple (ID_i, P_1, X, m, h_3) in L_3. If such a tuple exists, $\mathcal{C}$ returns h_3. Otherwise, $\mathcal{C}$ chooses a new $h_3 \in_R \mathbb{Z}_q^*$, adds the tuple (ID_i, P_1, X, m, h_3) to L_3 and returns h_3.

Oracle $\mathcal{O}_{\mathbf{H_4}}(\mathbf{U})$**.** $\mathcal{C}$ checks if there exists a tuple (U, h_4) in L_4. If such a tuple exists, $\mathcal{C}$ returns h_4. Otherwise, $\mathcal{C}$ chooses a new $h_4 \in_R \{0,1\}^{2n_1+n_3}$, adds the tuple (U, h_4) to L_4 and returns h_4.

Oracle $\mathcal{O}_{\mathbf{KeyExtract}}(\mathbf{ID_i})$**.** If L_1 does not contain an entry for ID_i, return $\perp$. Otherwise, $\mathcal{C}$ does the following.

1. If $ID_i = ID_B$, return $\perp$.

[5] By new, we mean that the random value chosen must not have been already chosen during an earlier execution.

2. If $ID_i \neq ID_B$, recover the tuple $(ID_i, Q_i, d_i, \mathcal{S}_i)$ from L_1 and return $S_i = d_i P_{pub} = aQ_i$.

Oracle $\mathcal{O}_{\mathbf{Register}}(\mathbf{ID_{B_j}}, \mathbf{ID_i}, \mathbf{y_i Q_i}, \mathbf{y_i Q_{B_j}})$**.** If L_1 does not contain an entry for ID_i or ID_{B_j}, then abort. Otherwise, $\mathcal{C}$ checks if $\hat{e}\left(y_i Q_i, Q_{B_j}\right) = \hat{e}\left(Q_i, y_i Q_{B_j}\right)$. If not, then abort. Otherwise, it does the following.

1. Update the tuple $(ID_i, Q_i, d_i, \mathcal{S}_i)$ in L_1 by setting $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \left\{\left(x_i, ID_{B_j}\right)\right\}$, where $x_i = H_2(\alpha_i)$, where $\alpha_i = \hat{e}\left(y_i Q_{B_j}, S_i\right)$, where the subscriber's secret key is obtained from $\mathcal{O}_{\mathbf{KeyExtract}}(\mathbf{ID_i})$.
2. Retrieve the tuple $\left(ID_{B_j}, Q_{B_j}, d_{B_j}, \mathcal{S}_{B_j}\right)$ from L_1, where $\mathcal{S}_{B_j} = \{c_0, c_1, \ldots, c_t\}$. Construct the new subscriber polynomial as $f_{B_j}(x) = \left(\sum_{\ell=0}^{t} c_\ell x^\ell\right) \cdot (x - x_i)$. Let the set of new coefficients be $\mathcal{S}'_{B_j} = \left\{c'_0, c'_1, \ldots, c'_{t+1}\right\}$. Update this tuple in L_1 by replacing $\mathcal{S}_{B_j}$ with $\mathcal{S}'_{B_j}$.

Oracle $\mathcal{O}_{\mathbf{Unregister}}(\mathbf{ID_{B_j}}, \mathbf{ID_i})$**.** If L_1 does not contain an entry for ID_i or ID_{B_j}, then abort. Otherwise, $\mathcal{C}$ does the following.

1. Update the tuple $(ID_i, Q_i, d_i, \mathcal{S}_i)$ in L_1 by setting $\mathcal{S}_i \leftarrow \mathcal{S}_i - \left\{\left(x_i, ID_{B_j}\right)\right\}$. Temporarily store the value of x_i for use in the next step.
2. Retrieve the tuple $\left(ID_{B_j}, Q_{B_j}, d_{B_j}, \mathcal{S}_{B_j}\right)$ from L_1, where $\mathcal{S}_{B_j} = \{c_0, c_1, \ldots, c_t\}$. Construct the new subscriber polynomial as $f_{B_j}(x) = \left(\sum_{\ell=0}^{t} c_\ell x^\ell\right) \cdot (x - x_i)^{-1}$. Let the set of new coefficients be $\mathcal{S}'_{B_j} = \left\{c'_0, c'_1, \ldots, c'_{t-1}\right\}$. Update this tuple in L_1 by replacing $\mathcal{S}_{B_j}$ with $\mathcal{S}'_{B_j}$.

Oracle $\mathcal{O}_{\mathbf{Signcrypt}}\left(\mathbf{m}, \mathbf{ID_{B_j}}\right)$**.** On receiving this query, $\mathcal{C}$ checks if there is an entry for ID_{B_j} in L_1 and if the set $\mathcal{S}_{B_j}$ is not singleton. If one or both of these conditions are not satisfied, then $\mathcal{C}$ aborts. Otherwise, $\mathcal{C}$ retrieves the tuple $\left(ID_{B_j}, Q_{B_j}, d_{B_j}, \mathcal{S}_{B_j}\right)$ from L_1, where $\mathcal{S}_{B_j} = \{c_0, c_1, \ldots, c_t\}$, and checks if $ID_{B_j} = ID_B$. If not, then it executes ***Signcrypt***$(m, ID_{B_j}, \mathcal{S}_{B_j}, S_{B_j})$ as usual, where the secret key S_{B_j} is computed from $\mathcal{O}_{\mathbf{KeyExtract}}(\mathbf{ID_{B_j}})$ and returns what the signcryption algorithm returns. In the case when $ID_{B_j} = ID_B$, it chooses $r \in_R \mathbb{Z}_q^*$ and a new $h_3 \in_R \mathbb{Z}_q^*$ and does the following.

1. Compute $P_1 = rc_1 P$, $P_2 = rc_2 P$, …, $P_t = rc_t P$.
2. Compute $X = rP - h_3 Q_B$ and add the tuple (ID_B, P_1, X, m, h_3) to L_3.
3. Compute the following.
   (a) $Z = rP_{pub} = raP$
   (b) $\beta = \hat{e}(Z, P)$
   (c) $U = \mathcal{O}_{H_2}(\beta) P$
   (d) $P_0 = rc_0 P + U$
   (e) $y = (m \| X \| Z) \oplus \mathcal{O}_{H_4}(U)$
4. Return the signcryption $\sigma_B = (ID_B, y, P_0, P_1, \ldots, P_t)$.

Oracle $\mathcal{O}_{\mathbf{Designcrypt}}\left(\sigma_{\mathbf{B_j}}, \mathbf{ID_i}\right)$**.** On receiving the signcryption $\sigma_{B_j} = \left(ID_{B_j}, y, P_0, P_1, \ldots, P_t\right)$, $\mathcal{C}$ first checks if there are entries for ID_{B_j} and ID_i in L_1 and there is a tuple of the form $\left(x_i, ID_{B_j}\right) \in \mathcal{S}_i$. If one or more of these conditions are not satisfied, then $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ executes ***Designcrypt***$(\sigma_{B_j}, ID_i, x_i)$ in the normal way and returns what the designcryption algorithm returns.
Eventually $\mathcal{A}$ outputs a forged signcryption $\sigma_B^* = (ID_B, y^*, P_0^*, P_1^*, \ldots, P_t^*)$ on some message m^* from the broadcaster B to all his subscribers. Challenger $\mathcal{C}$ retrieves the entry corresponding to ID_B in L_1 and uses one of the tuples of $\mathcal{S}_B$, say (x_i, ID_i) to execute ***Designcrypt***(σ_B, ID_i, x_i). If σ_B^* is a valid signcryption from the broadcaster B to his subscribers, that is, a message m^* is returned by the decryption algorithm, then $\mathcal{C}$ applies the oracle replay technique to produce two valid signcryptions $\sigma_B' = (ID_B, y', P_0', P_1, \ldots, P_t)$ and $\sigma_B'' = (ID_B, y'', P_0'', P_1, \ldots, P_t)$ on some message m from the broadcaster B to all his subscribers. $\mathcal{C}$ designcrypts σ_B' and σ_B'' to obtain the 'signatures' $Z' = (r+h_3')S_B$ and $Z'' = (r+h_3'')S_B$. Now we can apply standard arguments for the outputs of the forking lemma since both Z' and Z'' are valid signatures for the same message m and same random tape of the adversary. Finally, $\mathcal{C}$ obtains the solution to the CDH instance as $(h_3' - h_3'')^{-1}(Z' - Z'')$. We have

$$\begin{aligned}(h_3' - h_3'')^{-1}(Z' - Z'') &= (h_3' - h_3'')^{-1}(h_3' - h_3'')S_B \\ &= SD_B = abP\end{aligned}$$

So, we can see that the challenger $\mathcal{C}$ has the same advantage in solving the CDH problem as the adversary $\mathcal{A}$ has in forging a valid signcryption. So, if there exists an adversary who can forge a valid signcryption with non-negligible advantage, that means there exists an algorithm to solve the CDH problem with non-negligible advantage. Since this is not possible, no adversary can forge a valid signcryption with non-negligible advantage. Hence, our proposed scheme is secure against any EUF-IBBSC-CMA2 attack. □

9 Efficiency of our IBBSC Scheme

In this section, we discuss the efficiency of our proposed IBBSC scheme. The major parameters involved are the computation costs for *signcryption* and *designcryption* operations, the communication cost and the storage at the user's end. For computational cost, we consider the number of pairing computations performed, as they are the costliest operations involved. Our IBBSC scheme performs just one pairing operation during *Signcrypt* and two pairing operations per user for *Designcrypt*. Apart from these pairing computations, we note that whenever a user subscribes to a particular broadcaster, three pairing computations are to be done (two of them for verifying the authenticity of the registering user). However, this is of little concern, as it is a one time operation. In the scheme of Mu et al. [25], we note that the polynomial has to be evaluated every time during *Signcrypt* which we totally avoid here by computing it in an incremental fashion during registration of users. Thus, many redundant computations are avoided. For the communication cost, we still have to broadcast $O(t)$

group elements if the number of subscribers is t. Coming to storage cost, we consider the storage at the broadcaster's end and storage at the user's end. The broadcaster has to store information about every subscriber and so the storage cost for him is $O(t)$. The storage cost for a user is only $O(1)$ as he does not have to store anything other than his secret key and precomputed secret. Thus, our IBBSC scheme is easily seen to be very efficient in terms of computation and storage.

10 Extension of our IBBSC Scheme

As mentioned in Section 9, we could factor out the repeated polynomial evaluation for every signcryption operation only because of the constraint that the broadcaster always broadcasts to all his subscribers. In some situations, it may be desired that the broadcaster be able to broadcast to any subset of his subscribers that he desires. For example, there may be a few subscribers who have failed to pay the regular subscription fee or have shown some malicious intent of unauthorized content redistribution, etc. In such cases, broadcasters must be able to temporarily suspend these faulty subscribers in order to punish them and impose penalty. Our scheme can be extended to work for this case too. The only modification that needs to be done is that, for the *Signcrypt* algorithm, the list of users to which the broadcaster wants to send content must be an additional input and the polynomial has to be freshly computed to include only these user's x_i values. The rest of the scheme remains the same. Of course, in this extended scheme, the efficiency that was obtained by moving the polynomial evaluation to the *Register* phase is lost.

11 Conclusion

In this paper, we have considered the problem of secure and authenticated content distribution over large networks, especially wireless networks, which, on one hand, are increasingly becoming popular choices for the modern civilization, what with the advent of mobile and portable devices such as cell phones and PDAs, and on the other hand, are much easier to eavesdrop than wired networks. Broadcast signcryption schemes provide the solution to this problem and in the context of mobile devices being the computers at the end users, the efficiency of such schemes becomes very important — there is limited memory and computational power that is available. ID-based schemes are arguably the most suited because of the unique advantage that they provide — the public key of a user can be computed from any publicly available parameter of that user that is unique to him, thereby eliminating the complex public key infrastructure that would otherwise have to be employed. We have demonstrated an existential forgery attack on Mu et al's scheme. Following this, we have proposed a robust IBBSC scheme and also proven its IND-CCA2 and EUF-CMA2 security formally in the random oracle model. These are the strongest security notions for message confidentiality and authentication respectively. By imposing a reasonable restriction that broadcasters always send content to all their subscribers (while users are still allowed to join or leave broadcasters dynamically), we were able to reduce the computation cost significantly. We have also extended our scheme to the normal IBBSC scheme, where a broadcaster can selectively send content to an arbitrary (dynamically changing) subset of his subscribers.

Future Work. The scheme that we have proposed still suffers from the fact that the size of the signcryption that is to be broadcasted is linear in the number of privileged users the broadcaster intends to send the content to. In the context of mobile networks, it would be a phenomenal improvement if this can be made constant size or even logarithmic in the number of privileged users. Another observation is that the polynomial computation seems unavoidable if the broadcaster is given the freedom to choose any subset of his subscribers to send the content. It is desirable to discover IBBSC schemes that either cleverly avoid this computation, or those that do not involve such expensive computations every time a message is to be signcrypted. When looking at the number of pairing computations that are involved in the scheme, it is worthwhile to see if the number of pairing computations can be further reduced during designcryption, though it seems unlikely to be able to do so without compromising the security of the system.

References

1. Joonsang Baek, Ron Steinfeld, and Yuliang Zheng. Formal proofs for the security of signcryption. In *PKC 2002: Proceedings of the 5th International Workshop on Practice and Theory in Public Key Cryptography*, volume 2274 of *Lecture Notes in Computer Science*, pages 80–98. Springer-Verlag, 2002.
2. Feng Bao and Robert H. Deng. A signcryption scheme with signature directly verifiable by public key. In *PKC '98: Proceedings of the 1st International Workshop on Practice and Theory in Public Key Cryptography*, volume 1431 of *Lecture Notes in Computer Science*, pages 55–59. Springer-Verlag, 1998.
3. Paulo S. L. M. Barreto, Benoît Libert, Noel McCullagh, and Jean-Jacques Quisquater. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In *Advances in Cryptology - ASIACRYPT 2005: Proceedings of the 11th International Conference on the Theory and Application of Cryptology and Information Security*, volume 3788 of *Lecture Notes in Computer Science*, pages 515–532. Springer-Verlag, 2005.
4. Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2004: Proceedings of the 23rd International Conference on the Theory and Application of Cryptographic Techniques*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer-Verlag, 2004.
5. Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology - CRYPTO 2004: Proceedings of the 24th Annual International Cryptology Conference*, volume 3152 of *Lecture Notes in Computer Science*, pages 443–459. Springer-Verlag, 2004.

6. Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In *Advances in Cryptology - EUROCRYPT 2005: Proceedings of the 24th International Conference on the Theory and Application of Cryptographic Techniques*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer-Verlag, 2005.
7. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology - CRYPTO 2001: Proceedings of the 21st Annual International Cryptology Conference*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer-Verlag, 2001.
8. Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *Advances in Cryptology - CRYPTO 2005: Proceedings of the 25th Annual International Cryptology Conference*, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275. Springer-Verlag, 2005.
9. Xavier Boyen. Multipurpose identity-based signcryption (a swiss army knife for identity-based cryptography). In *Advances in Cryptology - CRYPTO 2003: Proceedings of the 23rd Annual International Cryptology Conference*, volume 2729 of *Lecture Notes in Computer Science*, pages 383–399. Springer-Verlag, 2003.
10. Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In *Advances in Cryptology - EUROCRYPT 2004: Proceedings of the 23rd International Conference on the Theory and Application of Cryptographic Techniques*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222. Springer-Verlag, 2004.
11. Liqun Chen and John Malone-Lee. Improved identity-based signcryption. In *PKC 2005: Proceedings of the 8th International Workshop on Practice and Theory in Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 362–379. Springer-Verlag, 2005.
12. Sherman S. M. Chow, Siu-Ming Yiu, Lucas Chi Kwong Hui, and K. P. Chow. Efficient forward and provably secure id-based signcryption scheme with public verifiability and public ciphertext authenticity. In *ICISC 2003: Proceedings of the 6th International Conference on Information Security and Cryptology*, volume 2971 of *Lecture Notes in Computer Science*, pages 352–369. Springer-Verlag, 2003.
13. Clifford Cocks. An identity based encryption scheme based on quadratic residues. In *IMA 2001: Proceedings of the 8th International Conference in Cryptography and Coding*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363. Springer-Verlag, 2001.
14. Cécile Delerablée, Pascal Paillier, and David Pointcheval. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In *PAIRING 2007: Proceedings of the 1st International Conference on Pairing-Based Cryptography*, volume 4575 of *Lecture Notes in Computer Science*, pages 39–59. Springer-Verlag, 2007.
15. Amos Fiat and Moni Naor. Broadcast encryption. In *Advances in Cryptology - CRYPTO '93: Proceedings of the 13th Annual International Cryptology Conference*, volume 773 of *Leture Notes in Computer Science*, pages 480–491. Springer-Verlag, 1994.
16. Chandana Gamage, Jussipekka Leiwo, and Yuliang Zheng. Encrypted message authentication by firewalls. In *PKC 1999: Proceedings of the 2nd International Workshop on Practice and Theory in Public Key Cryptography*, volume 1560 of *Lecture Notes in Computer Science*, pages 69–81. Springer-Verlag, 1999.
17. Craig Gentry. Practical identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2006: Proceedings of the 25th International*

Conference on the Theory and Application of Cryptographic Techniques, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464. Springer-Verlag, 2006.
18. Michael T. Goodrich, Jonathan Z. Sun, and Roberto Tamassia. Efficient tree-based revocation in groups of low-state devices. In *Advances in Cryptology - CRYPTO 2004: Proceedings of the 24th Annual International Cryptology Conference*, volume 3152 of *Lecture Notes in Computer Science*, pages 511–527. Springer-Verlag, 2004.
19. Dani Halevy and Adi Shamir. The lsd broadcast encryption scheme. In *Advances in Cryptology - CRYPTO 2002: Proceedings of the 22nd Annual International Cryptology Conference*, volume 2442 of *Lecture Notes in Computer Science*, pages 47–60. Springer-Verlag, 2002.
20. H. Y. Jung, K. S. Chang, D. H. Lee, and J. I. Lim. Signcryption schemes with forward secrecy. In *WISA 2001: Proceedings of the 2nd International Workshop on Information Security Applications*, pages 403–475, 2001.
21. Fagen Li, Xiangjun Xin, and Yupu Hu. Identity-based broadcast signcryption. *Computer Standards and Interfaces*, 30(1-2):89–94, 2008.
22. Benoît Libert and Jean-Jacques Quisquater. A new identity based signcryption scheme from pairings. *Proceedings of the IEEE Information Theory Workshop*, pages 155–158, 2003.
23. John Malone-Lee. Identity-based signcryption. Cryptology ePrint Archive, Report 2002/098, 2002.
24. John Malone-Lee and Wenbo Mao. Two birds one stone: Signcryption using rsa. In *Topics in Cryptology - CT-RSA 2003: Proceedings of the Cryptographers' Track at the RSA Conference 2003*, volume 2612 of *Lecture Notes in Computer Science*, pages 211–225. Springer-Verlag, 2003.
25. Yi Mu, Willy Susilo, Yan-Xia Lin, and Chun Ruan. Identity-based authenticated broadcast encryption and distributed authenticated encryption. In *Advances in Computer Science - ASIAN 2004: Proceedings of the 9th Asian Computing Science Conference*, volume 3321 of *Lecture Notes in Computer Science*, pages 169–181. Springer-Verlag, 2004.
26. Yi Mu and Vijay Varadharajan. Distributed signcryption. In *Progress in Cryptology - INDOCRYPT 2000: Proceedings of the 1st International Conference in Cryptology in India*, volume 1977 of *Lecture Notes in Computer Science*, pages 155–164. Springer-Verlag, 2000.
27. Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In *Advances in Cryptology - CRYPTO 2001: Proceedings of the 21st Annual International Cryptology Conference*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62. Springer-Verlag, 2001.
28. Moonseog Seo and Kwangjo Kim. Electronic funds transfer protocol using domain-verifiable signcryption scheme. In *ICISC '99: Proceedings of the 2nd International Conference on Information Security and Cryptology*, volume 1787 of *Lecture Notes in Computer Science*, pages 269–277. Springer-Verlag, 1999.
29. Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology - CRYPTO '84: Proceedings of the 4th Annual International Cryptology Conference*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer-Verlag, 1984.
30. Brent Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2005: Proceedings of the 24th International Conference on the Theory and Application of Cryptographic Techniques*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127. Springer-Verlag, 2005.

31. Dae Hyun Yum and Pil Joong Lee. New signcryption schemes based on kcdsa. In *ICISC 2001: Proceedings of the 4th International Conference on Information Security and Cryptology*, volume 2285 of *Lecture Notes in Computer Science*, pages 305–317. Springer-Verlag, 2001.
32. Yuliang Zheng. Digital signcryption or how to achieve cost(signature & encryption) << cost(signature) + cost(encryption). In *Advances in Cryptology - CRYPTO '97: Proceedings of the 17th Annual International Cryptology Conference*, volume 1294 of *Lecture Notes in Computer Science*, pages 165–179. Springer-Verlag, 1997.
33. Yuliang Zheng and Hideki Imai. How to construct efficient signcryption schemes on elliptic curves. *Information Processing Letters*, 68(5):227–233, December 1998.