Chapter 10
Identity-Based Signcryption

Xavier Boyen

10.1 Introduction

The notion of identity-based (IB) cryptography was proposed by Shamir [177] as a specialization of public key (PK) cryptography which dispensed with the need for cumbersome directories, certificates, and revocation lists.

We recall that in the traditional public key (or asymmetric key) cryptography model some mechanism must be used in order to bind a particular public key to its owner; often, this mechanism involves a trusted certificate authority (CA), whose role is to issue certificates, which are digital signatures that bind a user's public key to his/her real name. Such a system is called a public key infrastructure (PKI), and an apt metaphor for it is that of a phone book bearing the authentic seal of the phone authority.

By contrast, the distinguishing characteristic of IB cryptography lies in its ability to use any string as a public key, such as the real name of a person. Because of this, IB systems implement an automatic directory with implicit binding, without the need for costly certification and public key publication steps. Although public keys can be computed by anyone from public information, the corresponding private key can only be extracted by a trusted authority called the private key generator (PKG). The PKG has custody of a master secret, which allows it to compute any private key in the IB system. The PKG can be thought of as an identity-based analog to the CA at the helm of a traditional public key infrastructure.

10.1.1 Identity-Based Cryptography

In his original description, Shamir had already envisioned the use of IB cryptography for the purposes of signature and encryption. Although IB signatures could

X. Boyen (✉)
Chair of Cryptography and Information Security, Montefiore Institute, Universite de Liege, Liege, Belgium
e-mail: xb@boyen.org

A.W. Dent, Y. Zheng (eds.), *Practical Signcryption*, Information Security and Cryptography, DOI 10.1007/978-3-540-89411-7_10,

be constructed based on the techniques known at the time, it was only much later that a solution for IB encryption became known [45, 46]. In both types of scheme, individual users authenticate with the PKG in order to obtain their private key, in person or over a secure channel. The keys may then be used as follows:

IB signature (IBS): For signing, a private key can be used by its owner to create IB signatures: these signatures can be verified from the public parameters of the IB system only and are binding on the signer's name without requiring a certificate chain.

IB encryption (IBE): In the case of encryption, the private key will be used to decrypt any message encrypted under the recipient's proper name (and the IB system's public parameters): the originator need not look up the recipient's key, and indeed the recipient need not even know her private key at the time the ciphertext is created.

We note that in actual implementations, identity-based keys for signature and encryption are likely to be distinct and incompatible; however, the abstract key generation process is the same in both instances.

Many refinements to Shamir's model have been proposed in recent years. For key generation, Boneh and Franklin [45, 46] suggested that systems could take advantage of the flexibility in users' public keys by appending validity periods to the names of the individuals, in order to enforce a more frequent rotation of keys and lessen or eliminate the need for revocation lists. Another refinement is the combination of IB signature and IB encryption into a single IB signcryption operation [51], for both performance and security reasons.

IB signcryption (IBSC): Consider two parties, Alice and Bob, with unique names in some common IB system (controlled by the same PKG). Using her private key, Alice may signcrypt a message addressed to a recipient named Bob. Using his private key, Bob can decrypt the ciphertext and authenticate the sender as Alice.

On top of this basic functionality, there may be advantages to using an IB signcryption primitive that features a number of additional security properties. For instance, Bob may wish to obtain from the decryption process a cleartext signature by Alice stripped of its encryption: this requires that the process of *unsigncryption* be separable into a pair of independent decryption/verification algorithms (we shall call such an IBSC scheme a two-layer or *detachable* IBSC scheme). Additionally, it is often desirable to have some guarantee of anonymity, which is that no outsider should be able to recognize the parties involved in a signcrypted transmission.

The reader will notice that there are many similarities between the security properties that one can obtain from an identity-based signcryption scheme and certain non-ID-based signcryption schemes from pairings discussed in Chap. 5. In the next section we outline certain features specific to ID-based cryptography.

10.1.2 Advantages and Disadvantages

Identity-based cryptosystems differ substantially from their PKI counterparts in a number of respects. Before we turn our attention to IB signcryption per se, it is useful to review some of the main implications of identity-based private key generation. See also Paterson and Price [157] for further discussion on the advantages and disadvantages of identity-based cryptography.

10.1.2.1 Simplicity of Deployment

A substantial benefit of identity-based encryption over traditional public key systems is that the sender need not obtain the recipient's certified public key prior to initiating a secure communication. The recipient need not generate these keys ahead of time in the first place or even archive them on the client side since the PKG can always regenerate lost keys as needed. As a result, the number of flows of interactions between the various parties is reduced and key management tends to be greatly simplified, especially on the users' side.

10.1.2.2 Expiration vs. Revocation

As a side effect of the simplified key management that IB cryptography has to offer, the issue of compromise and revocation can be dealt with differently and more simply. Rather than deal with the long-lived keys and revocation lists typical of PKI, it is common in IB systems to eschew explicit revocation altogether and instead make the keys sufficiently short-lived that they will expire naturally shortly after any compromise. Boneh and Franklin [45, 46] propose appending a time-dependent common component, such as the number of weeks since a predetermined time in the past, to all static identities. To revoke a user, the PKG will simply stop issuing her new keys. This approach of using medium-lived keys is practical with IB systems, but not in traditional PKI, due to the higher complexity of the PKI key generation and certification process. Generally, medium-sized keys all but eliminate the need for revocation lists, unless revocation must occur with a shorter latency than any practical lifespan of the identity-based keys would allow.

Revocation lists are orthogonal to the IB model and can be used in conjunction with IB cryptography if necessary. However, it is generally an advantage not to push revocation lists to the edges of the networks (the users) and instead deal with revocation centrally, at the PKG level. Another advantage of this is that the list of revoked users need not be made public.

10.1.2.3 Compactness of Signatures

For signatures, the advantages of IB cryptography are less obvious, since IB signatures are functionally equivalent to regular PKI signatures with full certificate chains to the root CA; the main difference is that certificate chains are likely to occupy much more space than an IB signature. The benefit of identity-based signatures is

thus one of compactness. This benefit will carry over to identity-based signcryption, provided that the ciphertext can be stripped of the encryption layer to expose a plaintext with a regular IB signature.

IB signatures are also useful in cases where an IB encryption system is already in use, and the keys and infrastructure can be shared. In the context of IB signcryption, it is natural to seek to reuse the same infrastructure and keys for the signature and the encryption functionalities.

10.1.2.4 Concentration of Trust

The main criticism facing IB cryptography stems from the high level of trust that is bestowed upon the PKG, which has to be trusted at least to the same extent as a CA is trusted in the traditional model. Indeed, an untrustworthy PKG will have the power to forge signatures in the name of any user of the system, as well as the ability to decrypt all of their private communications. One difference is that an abuse of trust by a CA in a PKI is detectable by the afflicted party, whereas in an ID-based infrastructure there is more potential for a malicious PKG to remain undetected.

The single point of failure that constitutes the PKG can be partially alleviated by splitting the master secret among several PKGs under the jurisdiction of several independent authorities, using threshold techniques as explained in [45, 46]. Additionally, one can reduce the window of vulnerability from compromise of the PKG by instituting a policy whereby the public parameters are periodically changed, and all expired master secrets beyond a certain age are permanently purged from the system, which would effectively limit the interval during which any IB private key can be issued.

10.1.2.5 Proof of Possession

A small potential benefit of identity-based cryptography, over public key infrastructures, is that public key certification in the latter requires the registrant to submit a *proof of possession* of the corresponding private key, in addition to proper authentication credentials, or else security can be doubted. There is no analogous step in identity-based key extraction, which may result in one fewer point of failure.

10.1.2.6 Mandatory Key Escrow

A direct consequence of PKG-issued private keys is that the PKG acts as a mandatory key escrow. In certain circumstances this is not desirable, such as when the users of the system are individuals acting on their own behalf. In other settings the existence of a mandatory key escrow is indeed very desirable, as in corporate environments or in any case where the private key holders are members of a larger organization: the PKG then acts as an easy-to-administer and hard-to-circumvent central key escrow system, which ensures continuity of decryption by the company in the event that employees part with the organization without surrendering their keys. In general, this is a greater concern for encryption than for signatures.

10.1.3 From IBE to Signcryption

Although the idea of IB cryptography dates from 1984 [177], only an IB signature scheme was actually constructed at the time, based on conventional algebraic methods in composite-order RSA groups. One had to wait until 2000 and 2001 to see the apparition of practical IB encryption (IBE) schemes. One such IBE construction, due to Cocks [64], is based on the quadratic residuosity problem in traditional composite-order RSA groups. A more efficient approach was independently proposed in Sakai et al. [171] and Boneh and Franklin [45, 46], based on the mathematical notion of a bilinear pairing constructed on certain types of elliptic curves (see Chap. 5). Among these, Boneh and Franklin [45, 46] were the first to define a rigorous security model for IBE and prove the security of their construction in that model. The work of Sakai et al. [171] can be more appropriately described as an IB key exchange protocol that uses IB public and private keys similar to the Boneh–Franklin system. The difference is that key agreement requires secret keys on both sides and thus requires both parties to be enrolled in the system.

Pairings had made their appearances earlier in cryptology, first in the cryptanalysis of certain elliptic-curve systems with the MOV attack [138] and later in constructive cryptography with the creation of a tripartite key exchange protocol [109]. Also, and although IB signatures had been known long ago, it was soon realized that pairings opened the door to simpler and more efficient constructions than those already known. Among the first and most influential pairing-based IBS schemes, we mention Paterson [156], Hess [96], and Cha and Cheon [57].

10.1.3.1 Combining IBE with IBS

A natural question therefore is how to combine IB signatures or authentication with IB encryption. A direct approach would be to invoke such black-box combination techniques as discussed in Chap. 2, starting from any IBE and IBS primitive. This is based on the observation that the identity-based character of the primitives being combined does not interfere with the security of the combination, provided that their respective keys are independent.

The first real strides toward efficient IBSC constructions were, however, nongeneric. They included an authenticated key agreement scheme [59], an authenticated IBE system [127], and two IB signcryption schemes from [129, 122] with differing security properties. Such combined systems were typically more efficient than what could be achieved by using black-box combination techniques.

10.1.3.2 Alternative IB Paradigms

In parallel, over the years one has seen the development of several alternative approaches to IBE from pairings, with sometimes quite different characteristics and applications. We follow the nomenclature introduced in [52].

The *full-domain hash* IB family is that of the celebrated original Boneh–Franklin scheme and its many derivatives. It has many advantages, such as simplicity of

principle and implementation. Its main drawback is its unavoidable reliance on the random oracle model for all security reductions. The *commutative blinding* family is by far the largest. It originates in Boneh and Boyen's first IBE scheme (called BB-1) from [41]. It is more complex, but more efficient and empirically much more flexible, having been extended in many ways, e.g., to support parallel hierarchies [43], attributes [93], or wildcards [3]. The *exponent inversion* family is also quite well known. Its earliest instance in the random-oracle model is the Sakai–Kasahara [170] scheme and in the standard model the Boneh–Boyen [41] second scheme (called BB-2). Members of this family are often very efficient, but tend to require stronger complexity assumptions, and there are only few known extensions [52]. Gentry's [88] tight IBE scheme arguably belongs to this family.

Whereas in the past 5 years much of the activity in IB cryptography happened in the commutative blinding paradigm, most known IBSC constructions still follow the original full-domain hash framework (perhaps because of the simple and very convenient IBS primitives it supports). For this reason, the concrete IBSC schemes we discuss here are all based on the Boneh–Franklin full-domain hash paradigm.

10.1.4 Specifying an IBSC System

In this chapter, we study the question of combining IBE and IBS in a practical and secure way into a unified IBSC system with good security properties. Indeed, it is of great practical interest to be able to use the same IB infrastructure for signing and encrypting, while reaping efficiency gains over generic approaches in the process.

To this end, we aim to exploit similarities between IBE and IBS and elaborate a dual-purpose IBSC scheme based on a shared infrastructure. On the one hand, a unified system built on a shared infrastructure should bring us efficiency rewards. On the other hand, care must be taken to ensure that no hidden weakness arises from the combination, which is always a risk if the same parameters and keys are used. The questions we must address can be summarized as follows:

- Can IBE and IBS be practiced in conjunction, sharing infrastructure, parameters, and keys, with greater efficiency than black-box constructions?
- How can such a combination be done in a secure manner?
- What emerging security properties can be gained from the combination?

We will address these questions in a two-prong approach, first by defining a stringent security model for IBSC and then by studying an actual construction that fulfills the security model. Both the model and the construction are borrowed from Boyen [51].

10.1.4.1 Security Models for IBSC

Following [51], we define a five-prong security model that any unified IBSC system should satisfy. At the core, our model must capture the strong notions of security commonly accepted in public key cryptography, adapted for IBSC: indistinguisha-

bility of the ciphertext under adaptive chosen-ciphertext attacks and existential unforgeability of the signature under chosen-message attacks. In both cases, we specifically consider "insider" adversaries (see Chaps. 2 and 3).

Additionally, we propose three new security notions for IBSC: ciphertext authentication, anonymity, and unlinkability. Although less conventional, these security notions are highly desirable in practice: they serve to convince the legitimate recipient that the ciphertext itself is authentic and hide its origin and destination to any eavesdropper or man-in-the-middle impersonator (see also Chap. 5 for the related notions for non-identity-based signcryption).

10.1.4.2 Two-layer Detachable IBSC Design

After establishing the model, we construct a compliant IBSC scheme following a two-layer design. It consists of an inner randomized IBS component, on top of which is grafted a simplified deterministic IBE which "reuses" the randomness of the inner layer. This results in more compact ciphertexts than a generic composition of IBE and IBS. The two-layer design also allows the ciphertext to be stripped of its encryption in order to expose a signature on the decrypted message that anyone can verify. Here, the two-layer design is furthermore well suited for multi-recipient encryption of the same message, because the recipient-specific encryption header can be detached in such a way to allow the signature layer and the bulk message encryption to be shared across all recipients.

We remark that an efficient and generic approach for constructing "hybrid" signcryption schemes was recently proposed in [37], based on an underlying tag-KEM, a.k.a. key encapsulation mechanism with labels (see Chap. 7).

10.1.5 Concrete IBSC from Pairings

For concreteness, at the end of this chapter we shall study the IBSC construction of Boyen [51], which uses the properties of bilinear pairings to achieve a detachable sign-then-encrypt combination. In the nomenclature of [52], it is based on the full-domain hash IB paradigm of Boneh and Franklin and has proofs of security under the bilinear Diffie–Hellman (BDH) assumption [45, 46] in the random oracle model [29]. This scheme was selected because it satisfies the strongest and most useful notions of security for IBSC. Its construction borrows elements from the Boneh–Franklin IBE [45, 46] and the Cha–Cheon IBS [57], but achieves better performance than their generic combination.

We mention that a variation of the Boyen scheme [51] has been subsequently proposed by Chen and Malone-Lee [60]. The latter version is slightly more efficient but eschews some of the security properties of the original scheme, which is why we will focus on the original construction. We also note that several other IBSC systems have been proposed over the years [18, 122, 129, 136, 145, 170]; some of these are even more efficient than the two schemes we just mentioned, but at the expense of one or another important security property.

10.2 The Identity-Based Signcryption Primitive

An *identity-based signcryption* scheme, or IBSC, comprises four algorithms: `Setup`, `Extract`, `Signcrypt`, and `Unsigncrypt`. In a (two-layer) IBSC with *detachable signature*, the signcryption/unsigncryption algorithms are the composition of explicit subroutines: `Signcrypt` $=$ `Encrypt` $\circ$ `Sign` and `Unsigncrypt` $=$ `Verify` $\circ$ `Decrypt`.

In summary, `Setup` generates random instances of the common public parameters and master secret; `Extract` computes the private key corresponding to a given public identity string; `Signcrypt` produces a signature for a given message and private key, and then encrypts the signed plaintext for a given identity (note that the encryption routine may specifically require the signature as input); `Decrypt` decrypts a ciphertext using a given private key; `Verify` checks the validity of a given signature for a given message and identity. Messages are arbitrary strings in $\{0, 1\}^*$.

It is useful to decompose `Signcrypt` into `Sign` and `Encrypt`, even if the latter can only be applied on the output of the former. We shall need this finer level of granularity when discussing efficient multi-recipient signcryption in particular. With this convention, the functions that compose a generic IBSC scheme are as follows:

- `Setup`(1^k): On input 1^k, produces a pair (msk, mpk) (where msk is a randomly generated master secret and mpk the corresponding common public parameters, for the security parameter k).
- `Extract`(mpk, msk, ID): On input ID, computes a private key sk (corresponding to the identity ID under (msk, mpk)).
- `Signcrypt`$(mpk, ID_S, ID_R, sk_S, m)$: The sequential application of
  - `Sign`(mpk, ID_S, sk_S, m): On input (ID_S, sk_S, m), outputs a signature s (for sk_S, under mpk), and some ephemeral state data r.
  - `Encrypt`$(mpk, ID_R, sk_S, m, s, r)$: On input (ID_R, sk_S, m, s, r), outputs an anonymous ciphertext C (containing the signed message (m, s) encrypted for the identity ID_R under mpk).
- `Unsigncrypt`$(mpk, sk_R, \hat{C})$: The sequential application of
  - `Decrypt`$(mpk, sk_R, \hat{C})$: On input $(sk_R, \hat{C})$, outputs a triple $(\hat{ID}_S, \hat{m}, \hat{s})$ (containing the purported sender identity and signed message obtained by decrypting $\hat{C}$ by the private key sk_R under mpk).
  - `Verify`$(mpk, ID_S, \hat{m}, \hat{s})$: On input $(\hat{ID}_S, \hat{m}, \hat{s})$, outputs $\top$ "true" or $\bot$ "false" (indicating whether $\hat{s}$ is a valid signature for the message $\hat{m}$ by the identity $\hat{ID}_S$, under mpk).

As mentioned, we shall often view the sequential application of `Sign` and `Encrypt` as a single function, called `Signcrypt`, which for all purposes may be monolithic. However, we insist on keeping a formal separation between the `Decrypt` and `Verify` algorithms that constitute the function `Unsigncrypt`. The separation is necessary in order to allow the authenticity of the plaintext message to be verifiable by third parties, without requiring the recipient's decryption key. The two-step

unsigncryption process produces a decrypted message–signature pair as an intermediate output that is no longer bound to the recipient and is thus verifiable by anyone. Of course, if both operations are performed in lockstep, we may refer to them as a single `Unsigncrypt` function.

We have the following consistency constraints.

Definition 10.1 For master secret and common parameters $(msk, mpk) \xleftarrow{R} \texttt{Setup}(1^k)$, any identities ID_S and ID_R, and matching private keys $sk_S \xleftarrow{R} \texttt{Extract}(mpk, msk, ID_S)$ and $sk_R \xleftarrow{R} \texttt{Extract}(mpk, msk, ID_R)$, we require for consistency that, $\forall m \in \{0,1\}^*$:

$$\left.\begin{array}{c}(s,r) \xleftarrow{R} \texttt{Sign}(mpk, ID_S, sk_S, m) \\ C \xleftarrow{R} \texttt{Encrypt}(mpk, ID_R, sk_S, m, s, r) \\ (\hat{ID}_S, \hat{m}, \hat{s}) \leftarrow \texttt{Decrypt}(mpk, sk_R, \hat{C})\end{array}\right\} \Longrightarrow \begin{array}{c}\hat{ID}_S = ID_S \\ \hat{m} = m \\ \texttt{Verify}(mpk, ID_S, \hat{m}, \hat{s}) = \top\end{array}$$

We omit the parameters *mpk* and *msk* when understood from context.

Identity Roles for Signature and Encryption

To reduce the number of keys that need to be handed out by the PKG, it is desirable to allow the same user private key, extracted from a given identity, to be used alternatively as a signing key in a sender role and as a decryption key in a recipient role. This corresponds to the notion of one-key signcryption (see Chap. 3). The drawback of this approach, compared to two-key signcryption, is that it may complicate the security reduction. Furthermore, it may also be necessary for technical reasons to disallow the same identity from assuming both the sender and the recipient roles at once in the same ciphertext: This is the *irreflexivity requirement*.

If for some reason a "signcrypt-to-self" functionality is desired in a one-key system subject to the irreflexivity requirement, it can be emulated by making available to every user an additional "self"-identity for the sole purpose of signcrypting to oneself. A less economical option is to duplicate each identity into a signing identity and a decryption identity, in essence downgrading the one-key system to a two-key system, at the cost of doubling the number of private keys to be extracted and stored.

Notational Convention

In the sequel we consider one-key signcryption by default. For clarity, we adopt the convention of using the subscripts "S" for the sender and "R" for the recipient.

10.3 Security Definitions

We define a number of notions of security for identity-based signcryption.

Fundamental Properties

Our first two notions are the usual security notions for confidentiality and non-repudiation/origin authentication, adapted to the context of IBSC. Following the taxonomy of Chap. 2, in both cases we consider the strongest type of attacker, the "insider," which has access to all private keys except that of the party being attacked.

More precisely, when defining *message confidentiality*, we assume that the adversary may obtain any private key other than that of the targeted recipient and has an oracle that decrypts any valid ciphertext other than the challenge: this is an insider chosen-ciphertext attack in the terminology of Chap. 2.

When defining *signature non-repudiation*, we correspondingly assume that the forger has access to any private key other than that of the signer and can query an oracle that signs and encrypts any message but the challenge: This adversary therefore mounts an insider chosen-message existential forgery attack in the terminology of Chap. 2.

Peripheral Properties

We also define the complementary notions of *ciphertext authentication* and *ciphertext unlinkability*, which allow the legitimate recipient to privately authenticate that he was indeed the intended recipient of a particular ciphertext, but not prove this to a third party. This is important because the message (and its universally verifiable signature) does not necessarily specify who the intended recipient is; only the ciphertext does so unequivocally by virtue of being encrypted under a particular identity. Ciphertext authentication and unlinkability are not trivial to combine with non-repudiation and confidentiality, and we note, for example, that most IBSC schemes proposed in the literature do not achieve all four properties at once. Ciphertext authentication was introduced by Lynn [127] in the context of authenticated IBE and ciphertext unlinkability was defined by Boyen [51].

Another natural property to demand is *ciphertext anonymity* [51], which is the requirement that no third party should be able to discover whom a ciphertext originates from or is addressed to, without the recipient's private key. As for confidentiality, it is possible to define anonymity against insider attacks, where the adversary has access to the sender's signing key: this is the notion we shall consider. We note that the anonymity requirement only guarantees security against attacks that focus on the cryptographic aspect of IBSC; in practice, it will be equally important that the ciphertext conveyance mechanism from sender to recipient does not betray their identities, e.g., from a traffic analysis attack on the communication network. Ciphertext anonymity has recently become an active subject of inquiry in other areas of IBE; see, for example, [1, 44, 53].

Omitted Properties

A couple of additional properties of IBSC schemes have also been put forward in the literature; these properties are redundant or conflict with the above, so we will not define them explicitly.

One redundant property is that of *forward secrecy*, suggested in the context of IBSC first by Libert and Quisquater [122] and also by Nalla and Reddy [145], and later formalized by McCullagh and Barreto [136]. All these papers define forward secrecy as the infeasibility of recovering the message from an IBSC ciphertext, even under exposure of the private key of the sender. Since it is essentially the notion of semantic security under insider attacks defined in Chap. 2, forward secrecy is implied by our model and we will not need to consider it explicitly.

One incompatible property that has been put forward is that of *transferable verification*; see, for example, Libert and Quisquater [122] and McCullagh and Barreto [136]. Transferable verification requires that the ciphertext itself, and not just the decrypted message, be publicly verifiable under a weakened notion of authenticity that excludes knowledge of the message: transferable verification ensures that anyone, including third parties, can ascertain the true originator of a ciphertext (but not its content or the intended recipient).[1]

The main objection against transferable verification is that it violates intuitive expectations of secrecy, because the sender is compelled to broadcast her identity to everyone, in the clear and without repudiation. Transferable verification thus conflicts with ciphertext unlinkability. For these reasons, transferable verification is not necessarily needed or desirable for security; rather, it should be accepted only after due consideration of its ramifications.

Summary of the IBSC Security Notions

The five distinct IBSC security properties that we seek are thus the following:

1. *Insider message confidentiality* (Sect. 10.3.1): Guarantees the secrecy, or semantic security, of the message among the communicating parties, against any attacker, even if the sender's private key is subsequently exposed. This implies *forward secrecy*.
2. *Insider signature non-repudiation* (Sect. 10.3.2): Provides universal verifiability that a decrypted message was written by the signer. The signature remains binding even if the correct recipient's private key is exposed. As usual, non-repudiation implies message authentication and integrity.
3. *Ciphertext unlinkability* (Sect. 10.3.3): Allows the sender to disavow creating a ciphertext for any given recipient, even though he/she remains bound to any validly signed message it contains. In other words, it allows a sender to claim that her signed message was re-encrypted for another recipient.
4. *Ciphertext authentication* (Sect. 10.3.4): Guarantees to the legitimate recipient, alone, that the ciphertext and the signed message it contains were crafted by the same entity. This property also implies ciphertext integrity and, in particular,

[1] We remark that, among the three generic signcryption methods studied by Zheng [203, 204], "encrypt-then-sign" ($\mathcal{E}t\mathcal{S}$) entails transferable verification, "sign-then-encrypt" ($\mathcal{S}t\mathcal{E}$) forbids it, and "encrypt-and-sign" ($\mathcal{E}\&\mathcal{S}$) can go either way.

reassures the recipient that the communication was secured end to end and was not re-encrypted along the way.
5. *Insider ciphertext anonymity* (Sect. 10.3.5): Makes the ciphertext appear anonymous (hiding both the sender and the recipient identities) to anyone who does not possess the recipient decryption key. This remains true even if the sender's signing key is exposed.

These properties (including the redundant forward secrecy) were first achieved together in the IBSC construction of Boyen [51]. Subsequently, Chen and Malone-Lee [60] made the scheme computationally more efficient by sacrificing ciphertext unlinkability.

10.3.1 Message Confidentiality

Message confidentiality against adaptive chosen-ciphertext attacks is defined in terms of the following game, played between a challenger and an adversary. We combine signature and encryption into a dual-purpose oracle, to allow `Encrypt` to access the ephemeral random state data r from `Sign`. We give the adversary access to a decryption oracle which differs from an unsigncryption oracle in that it returns messages and signatures for correctly formed ciphertexts, rather than just messages.

1. *Start*: The challenger runs the `Setup` procedure for a given value of the security parameter k and provides the common public parameters mpk to the adversary, keeping the secret msk for itself.
2. *Phase 1*: The adversary makes a number of queries to the challenger, in an adaptive fashion (i.e., one at a time, with knowledge of the previous replies). The following queries are allowed:
   - *Signcryption queries* in which the adversary submits a message and two distinct identities, and obtains a ciphertext containing the message signed in the name of the first identity and encrypted for the second identity.
   - *Decryption queries* in which the adversary submits a ciphertext and an identity, and obtains the identity of the sender, the decrypted message, and a valid signature, provided that (1) the decrypted identity of the sender differs from that of the specified recipient and (2) the signature verification condition $\texttt{Verify} = \top$ is satisfied; otherwise, the oracle only indicates that the ciphertext is invalid for the specified recipient.
   - *Private key extraction queries* in which the adversary submits any identity of its choice and obtains the corresponding private key.
3. *Selection*: At some point, the adversary returns two distinct messages m_0 and m_1 (assumed to be of equal length), a signer identity ID_S, and a recipient identity ID_R, on which it wishes to be challenged. The adversary must have made no private key extraction query on ID_R.

4. *Challenge*: The challenger flips $b \xleftarrow{R} \{0, 1\}$, computes $sk_S \xleftarrow{R} \texttt{Extract}(ID_S)$, $(s, r) \xleftarrow{R} \texttt{Sign}(ID_S, sk_S, m_b)$, $C^* \xleftarrow{R} \texttt{Encrypt}(ID_R, sk_S, m_b, s, r)$, and returns the ciphertext C as challenge to the adversary.
5. *Phase 2*: The adversary adaptively issues a number of additional signcryption, decryption, and extraction queries, under the additional constraint that it not ask for the private key of ID_R or the decryption of C^* under ID_R.
6. *Response*: The adversary returns a guess $\hat{b} \in \{0, 1\}$ and wins if $\hat{b} = b$.

It is emphasized that the adversary is allowed to know the private key sk_S corresponding to the signing identity. The resulting notion is that of *insider security* for confidentiality, also called forward secrecy.

This game is very similar to the IND-ID-CCA attack defined in [45, 46]; we call it an IND-IBSC-CCA attack.

Definition 10.2 An identity-based signcryption (IBSC) scheme is said to be semantically secure against adaptive chosen-ciphertext insider attacks, or *IND-IBSC-CCA secure*, if no randomized polynomial-time adversary has a non-negligible advantage in the above game. In other words, the advantage $Adv_{\mathcal{A}}(k) = |\Pr[\hat{b} = b] - \frac{1}{2}|$ of every randomized polynomial-time IND-IBSC-CCA adversary $\mathcal{A}$ is a negligible function of the security parameter k.

We remark that the model requires the decryption oracle to perform a validity check before returning a decryption result, even though `Decrypt` does not specify it. This requirement does not weaken the model since the verification function is public and allows for stronger security results. We similarly ask that the oracles enforce the irreflexivity requirement, e.g., by refusing to produce or decrypt ciphertexts addressed to their sender.

10.3.2 Signature Non-repudiation

Signature non-repudiation is formally defined in terms of the following game, played between a challenger and an adversary.

1. *Start*: The challenger runs the `Setup` procedure for a given value of the security parameter k and provides the common public parameters *mpk* to the adversary, keeping the secret *msk* for itself.
2. *Query*: The adversary makes a number of queries to the challenger. The attack may be conducted adaptively and allows the same queries as in the confidentiality game of Sect. 10.3.1, namely signcryption queries, decryption queries, and private key extraction queries.
3. *Forgery*: The adversary returns a recipient identity ID_R and a ciphertext C.
4. *Outcome*: The adversary wins the game if the ciphertext C decrypts, under the private key of ID_R, to a signed message $(ID_S, \hat{m}, \hat{s})$ that satisfies $ID_S \neq ID_R$ and $\texttt{Verify}(ID_S, \hat{m}, \hat{s}) = \top$, where we also require that (1) no private key extraction query was made on ID_S and (2) no signcryption query was made that involved $\hat{m}$,

ID_S, and some recipient $ID_{R'}$, and resulted in a ciphertext C' whose decryption under the private key of $ID_{R'}$ is the claimed forgery $(ID_S, \hat{m}, \hat{s})$.

Such a model is very similar to the usual notion of existential unforgeability against chosen-message attacks [163]; we call it an sUF-IBSC-CMA attack.

Definition 10.3 An IBSC scheme is said to be existentially signature-unforgeable against chosen-message insider attacks, or *sUF-IBSC-CMA secure*, if no probabilistic, polynomial-time adversary has a non-negligible advantage in the forgery game above. That is, the advantage $Adv_{\mathcal{A}}(k) = \Pr[\texttt{Verify}(mpk, ID_S, \hat{m}, \hat{s}) = \top]$ of every randomized polynomial-time sUF-IBSC-CMA adversary $\mathcal{A}$ is a negligible function of the security parameter k.

In the above experiment, the adversary is allowed to obtain the private key sk_R for the forged message recipient ID_R, which corresponds to the stringent requirements of *insider security* for authentication (see Chaps. 2 and 3). There is one important difference: in Chaps. 2 and 3, unforgeability and non-repudiation apply to the ciphertext itself, which is the only sensible choice in the context of a signcryption model with a monolithic "unsigncryption" function. Here, given our two-step `Decrypt/Verify` specification, we define sUF-IBSC-CMA with a notion of non-repudiation that concentrates on the decrypted message and its signature, which is more intuitively desirable and does not preclude ciphertext unlinkability (see Sect. 10.3.3).

10.3.3 Ciphertext Unlinkability

Ciphertext unlinkability is the property that makes it possible for Alice to deny having sent a given ciphertext to Bob, even if the ciphertext decrypts (under Bob's private key) to a message bearing Alice's signature. In other words, the signature should only be a proof of authorship of the plaintext message; not that the ciphertext was addressed to a particular recipient.

Ciphertext unlinkability allows Alice, e.g., as a news correspondent in a hostile area, to stand behind the content of her reporting, but conceals any detail regarding the particular channel, method, place, or time of communication, lest subsequent forensic investigations be damaging to her sources. When used in conjunction with the multi-recipient technique of Sect. 10.4.4, ciphertext unlinkability also allows her to make exact copies of her writings to additional recipients without anyone being able to prove that she made those copies.

We do not present a formal experiment for this property. A sufficient condition for this property is that, given a plaintext message signed by Alice, anyone should be able to create from it a valid ciphertext addressed to himself with an identical distribution as the corresponding signcryption from Alice.

Definition 10.4 An IBSC scheme is said to be ciphertext unlinkable if there exists a polynomial-time algorithm `EncryptToSelf` that, given an identified signed message (ID_S, m, s) such that $\texttt{Verify}(ID_S, m, s) = \top$, and a private key $d_R \xleftarrow{R}$

$\texttt{Extract}(ID_R)$, assembles a ciphertext C that is computationally indistinguishable from a genuine encryption of (m, s) by ID_S for ID_R.

As mentioned earlier, ciphertext unlinkability is the reason why we considered the notion of (plaintext) signature unforgeability in Sect. 10.3.2, instead of the usual notion of ciphertext unforgeability as studied in the signcryption model discussed in Chaps. 2 and 3. Indeed, if the ciphertext itself were unforgeable it would not be unlinkable.

Note also that ciphertext unlinkability only makes sense in a detachable signcryption model as in this chapter, as opposed to the monolithic model of Zheng [203, 204] used by Malone-Lee [129] and by Libert and Quisquater [122]. Indeed, if part of the ciphertext itself is needed to verify the authenticity of the plaintext, ciphertext indistinguishability is lost as soon as the recipient is compelled to expose the validity of the signature. Ciphertext unlinkability is thus a property that is unattainable in the monolithic signcryption model.

10.3.4 Ciphertext Authentication

Ciphertext authentication is, in a sense, complementary to ciphertext unlinkability. Whereas unlinkability required that the recipient be unable to prove the origin of a given ciphertext to a third party, authentication allows the recipient to positively authenticate the same ciphertext as originating from Alice: it just cannot prove it to anyone else. Technically, we define ciphertext authentication as the requirement that the legitimate recipient be able to match the origin of a ciphertext with that of the signed message it contains.

A useful application is to convince the recipient that the ciphertext remained encrypted throughout the entire transmission (because it would not pass the test if it had been re-encrypted in transit). In particular, a ciphertext properly authenticated in this model cannot have been the target of a (successful, active) man-in-the-middle interception. We define ciphertext authentication in terms of the following game:

1. *Start*: The challenger runs the $\texttt{Setup}$ procedure for a given value of the security parameter k and provides the common public parameters mpk to the adversary, keeping the secret msk for itself.
2. *Query*: The adversary makes a number of queries to the challenger as in the confidentiality game of Sect. 10.3.1, namely signcryption queries, decryption queries, and private key extraction queries.
3. *Forgery*: The adversary returns a recipient identity ID_R and a ciphertext C.
4. *Outcome*: The adversary wins the game if C decrypts, under the private key of ID_R, to a signed message $(ID_S, \hat{m}, \hat{s})$ such that $ID_S \neq ID_R$ and that satisfies $\texttt{Verify}(ID_S, \hat{m}, \hat{s}) = \top$, provided that (1) no private key extraction query was made on either ID_S or ID_R and (2) C did not result from a signcryption query with sender and recipient identities ID_S and ID_R.

We contrast the above experiment, which is a case of "outsider" security for authentication on the whole ciphertext, with the scenario for signature non-repudiation, which required insider security on the signed plaintext only. We call the above experiment an AUTH-IBSC-CMA attack.

Definition 10.5 An IBSC scheme is said to be existentially ciphertext-unforgeable against chosen-message outsider attacks, or *AUTH-IBSC-CMA secure*, if no randomized polynomial-time adversary has a non-negligible advantage in the preceding game. That is, the advantage $Adv_{\mathcal{A}}(k) = \Pr[\texttt{Verify}(ID_S, \hat{m}, \hat{s}) = \top]$ of every randomized polynomial-time sUF-IBSC-CMA adversary $\mathcal{A}$ is a negligible function of the security parameter k.

10.3.5 Ciphertext Anonymity

Ciphertext anonymity is the last property we define. It requires that the ciphertext leak no knowledge about its originator or its intended recipient to a polynomially bounded adversary. (Naturally, the ciphertext must be decipherable by the intended recipient without that information.)

Ciphertext anonymity against adaptive chosen-ciphertext attacks is defined as follows:

1. *Start*: The challenger runs the `Setup` procedure for a given value of the security parameter k and provides the common public parameters *mpk* to the adversary, keeping the secret *msk* for itself.
2. *Phase 1*: The adversary is allowed to make adaptive queries of the same types as in the confidentiality game of Sect. 10.3.1, i.e., signcryption queries, decryption queries, and private key extraction queries.
3. *Selection*: At some point, the adversary returns a message m, two sender identities ID_{S_0} and ID_{S_1}, and two recipient identities ID_{R_0} and ID_{R_1}, on which it wishes to be challenged. The adversary must have made no private key extraction query on either ID_{R_0} or ID_{R_1}.
4. *Challenge*: The challenger flips two random coins $b', b'' \overset{R}{\leftarrow} \{0, 1\}$, computes $sk \overset{R}{\leftarrow} \texttt{Extract}(ID_{S_{b'}})$, $(s, r) \overset{R}{\leftarrow} \texttt{Sign}(ID_{S_{b'}}, sk, m)$, $C \overset{R}{\leftarrow} \texttt{Encrypt}(ID_{R_{b''}}, sk_S, m, s, r)$, and gives the ciphertext C to the adversary.
5. *Phase 2*: The adversary adaptively issues a number of additional signcryption, decryption, and extraction queries, under the additional constraint that it not ask for the private key of either ID_{R_0} or ID_{R_1} or the decryption of C under ID_{R_0} or ID_{R_1}.
6. *Response*: The adversary returns two guesses $\hat{b}', \hat{b}'' \in \{0, 1\}$ and wins the game if $(\hat{b}', \hat{b}'') = (b', b'')$.

This game is the same as for confidentiality, except that the adversary is challenged on the identities instead of the message; it is an insider attack. We call it an ANON-IBSC-CCA attack.

Definition 10.6 An IBSC scheme is said to be ciphertext-anonymous against adaptive chosen-ciphertext insider attacks, or *ANON-IBSC-CCA secure*, if no randomized polynomial-time adversary has a non-negligible advantage in the above game. In other words, the advantage $Adv_{\mathcal{A}}(k) = |\Pr[\hat{b} = b] - \frac{1}{4}|$ of every randomized polynomial-time ANON-IBSC-CCA adversary $\mathcal{A}$ is a negligible function of the security parameter k, where $b = (b', b'')$ and $\hat{b} = (\hat{b}', \hat{b}'')$.

We emphasize that anonymity only pertains to the ciphertext, against non-recipients and is thus consistent with both non-repudiation (Sect. 10.3.2) and authentication (Sect. 10.3.4). To illustrate the difference between unlinkability and anonymity, we note that the authenticated IBE scheme of Lynn [127] is unlinkable, since any ciphertext can be created by its recipient rather than its sender, but not anonymous, since the sender identity must be known prior to decryption in order to decrypt.

An analogous notion of ciphertext anonymity exists for traditional public key cryptography (see the discussion in Chap. 5).

10.4 A Concrete IBSC Scheme

In this section we construct two efficient identity-based signcryption schemes; both are based on the two-layer detachable design and satisfy the full complement of security properties presented in Sect. 10.3. Both constructions make use of the Boneh–Franklin setup, which we recall next.

10.4.1 The Boneh–Franklin Framework

We give a brief summary of the Boneh–Franklin system for identity-based cryptography based on bilinear pairings on elliptic curves. Its setup and private key generation algorithms will be used in the IBSC construction.

We recall the notion of a bilinear map group from Sect. 5.2. In this chapter, we treat the bilinear pairing and the algebraic group over which it is defined as abstract mathematical objects satisfying the properties summarized in a few definitions to follow.

Let $\mathbb{G}_1$ and $\mathbb{G}_T$ be two cyclic groups of prime order p written in multiplicative notation (and using 1 to denote their respective neutral elements).

Definition 10.7 A *bilinear pairing* is an efficiently computable, non-degenerate map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ such that, for all $x, y \in \mathbb{G}_1$ and all $a, b \in \mathbb{Z}$, we have $e(x^a, y^b) = e(x, y)^{a\,b}$. The group $\mathbb{G}_1$ is called a bilinear map group; the group $\mathbb{G}_T$ is the target group.

Definition 10.8 The (computational) *bilinear Diffie–Hellman (BDH) problem* in a bilinear map group as above is described as follows: given $g, g^a, g^b, g^c \in \mathbb{G}_1$, where g is a generator and $a, b, c \xleftarrow{R} \mathbb{Z}_p$ are chosen at random, compute $e(g, g)^{a\,b\,c}$. The

advantage of an algorithm $\mathcal{B}$ at solving the BDH problem is defined as $Adv_{\mathcal{B}}(k) = \Pr[\mathcal{B}(g, g^a, g^b, g^c) = e(g,g)^{a\,b\,c}]$.

Definition 10.9 Let $\mathcal{G}$ be a polynomial-time randomized function that, on input 1^k, returns the description of a bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ between two groups $\mathbb{G}_1$ and $\mathbb{G}_T$ of prime order p. A BDH parameter generator $\mathcal{G}$ satisfies the *bilinear Diffie–Hellman assumption* if there is no probabilistic, polynomial-time algorithm $\mathcal{B}$ that solves the BDH problem in time at most polynomial in k and with advantage at least inverse polynomial in k. The probability space is that of the randomly generated parameters $(\mathbb{G}_1, \mathbb{G}_T, p, e)$, the BDH instances (g, g^a, g^b, g^c), and the randomized executions of $\mathcal{B}$.

Using a BDH parameter generator, the Boneh–Franklin IBE scheme defines four operations: two operations used by the PKG (for setup and key extraction) and two used by the individual users (for encryption and decryption). We will make use of the two PKG algorithms (as defined below):

`bfSetup`: On input a security parameter $k \in \mathbb{N}$: obtain $(\mathbb{G}_1, \mathbb{G}_T, p, e) \xleftarrow{R} \mathcal{G}(1^k)$ from the BDH parameter generator; pick a random generator $g \xleftarrow{R} \mathbb{G}_1$ and a random exponent $msk \xleftarrow{R} \mathbb{Z}_p$, set $g^{msk} \in \mathbb{G}_1$; and specify a hash function $H_0 : \{0,1\}^* \to \mathbb{G}_1$. Output the common public parameters $mpk = (\mathbb{G}_1, \mathbb{G}_T, p, e, g, g^{msk}, H_0)$ and the master secret msk.

`bfExtract`: On input $ID \in \{0,1\}^*$: hash the given identity into a public element $i_{ID} \leftarrow H_0(ID) \in \mathbb{G}_1$ and output $d_{ID} \leftarrow (i_{ID})^{msk} \in \mathbb{G}_1$ as the private key sk_{ID}.

10.4.2 Fully Secure IBSC Construction

Table 10.1 details the algorithms of the scheme.

Although `Sign` and `Encrypt` are described separately, the latter can only be run on the output of the former; together they constitute the atomic identity-based `Signcrypt` operation.

Recall also that `Decrypt` and `Verify` together define the `Unsigncrypt` operation, but those can be used separately.

10.4.2.1 Principle of Operation

The `Setup` and `Extract` functions are based on the original Boneh–Franklin IBE system [45, 46]. `Sign` and `Verify` implement the IBS of Cha and Cheon [57]. `Encrypt` and `Decrypt` are specially crafted to interface with the IBS layer and reuse its randomness.

In brief, `Sign` implements a randomized IBS whose signatures comprise a commitment j to some random r chosen by the sender and a closing v that depends on r and the message m. `Encrypt` superimposes two layers of (expansionless)

Table 10.1 The identity-based signcryption (IBSC) scheme introduced by Boyen [51]

`Setup`

Input: security parameter $k \in \mathbb{N}$

Method:

Create Boneh–Franklin parameters $\mathbb{G}_1, \mathbb{G}_T, p, e, g, g^{msk}$ and secret msk as in `bfSetup`

Specify five independent cryptographic hash functions (H_0 as in `bfSetup`):

$H_0 : \{0,1\}^* \to \mathbb{G}_1$

$H_1 : \mathbb{G}_1 \times \{0,1\}^* \to \mathbb{Z}_p$

$H_2 : \mathbb{G}_T \to \{0,1\}^{\lceil \log p \rceil}$

$H_3 : \mathbb{G}_T \to \mathbb{Z}_p$

$H_4 : \mathbb{G}_1 \to \{0,1\}^k$

Output: the public system parameters $(\mathbb{G}_1, \mathbb{G}_T, p, e, g, g^{msk}, k, H_0, H_1, H_2, H_3, H_4)$ and corresponding master secret $msk \in \mathbb{Z}_p$

`Extract`

Input: master secret msk and identity string $ID \in \{0,1\}^*$

Output: private key $d_{ID} \leftarrow H_0(ID)^{msk} \in \mathbb{G}_1$, as in `bfExtract`

`Signcrypt` = `Sign` + `Encrypt`

`Sign`

Input: private key d_S of some ID_S, plaintext message m

Method:

$i_S \leftarrow H_0(ID_S)$ (so $d_S = (i_S)^{msk}$)

Randomly sample $r \xleftarrow{R} \mathbb{Z}_p$

$j \leftarrow (i_S)^r \in \mathbb{G}_1$

$h \leftarrow H_1(j, m) \in \mathbb{Z}_p$

$v \leftarrow (d_S)^{r+h} \in \mathbb{G}_1$

Output: signature (j, v) and auxiliary data (m, r, ID_S, i_S, d_S)

`Encrypt`

Input: recipient ID_R, signature data $(ID_S, i_S, d_S, j, v, m, r)$ as above

Method:

$i_R \leftarrow H_0(ID_R)$

$u \leftarrow e(d_S, i_R) \in \mathbb{G}_T$

$k \leftarrow H_3(u) \in \mathbb{Z}_p$

$x \leftarrow j^k \in \mathbb{G}_1$

$w \leftarrow u^{k\,r} \in \mathbb{G}_T$

$y \leftarrow H_2(w) \oplus v$

$z \leftarrow \mathtt{Enc}_{H_4(v)}((ID_S, m))$

Output: ciphertext (x, y, z)

`Unsigncrypt` = `Decrypt` + `Verify`

`Decrypt`

Input: pvt. key d_R of recipient ID_R, anonymous ciphertext $(\hat{x}, \hat{y}, \hat{z})$

Method:

$i_R \leftarrow H_0(ID_R)$

$\hat{w} \leftarrow e(\hat{x}, d_R)$

$\hat{v} \leftarrow H_2(\hat{w}) \oplus \hat{y}$

$(\hat{ID}_S, \hat{m}) \leftarrow \mathtt{Dec}_{H_4(\hat{v})}(\hat{z})$

$\hat{i}_S \leftarrow H_0(\hat{ID}_S)$

$\hat{u} \leftarrow e(\hat{i}_S, d_R)$

$\hat{k} \leftarrow H_3(\hat{u})$

$\hat{j} \leftarrow \hat{x}^{\hat{k}^{-1}}$

Output: purported plaintext $\hat{m}$, signature $(\hat{j}, \hat{v})$, and sender $\hat{ID}_S$

`Verify`

Input: message $\hat{m}$, signature $(\hat{j}, \hat{v})$, and sender $\hat{ID}_S$ to verify

Method:

$\hat{i}_S \leftarrow H_0(\hat{ID}_S)$

$\hat{h} \leftarrow H_1(\hat{j}, \hat{m})$

Test $e(g, \hat{v}) \stackrel{?}{=} e(g^{msk}, (\hat{i}_S)^{\hat{h}}\, \hat{j})$

Output: $\top$ if equality holds; else $\bot$

Here, $\mathtt{Enc}_{key}(data)$ and $\mathtt{Dec}_{key}(data)$ are the encryption and decryption functions of a deterministic symmetric cipher assumed semantically secure under passive attacks (for one-time keys), e.g., the "XOR" operation with the key used as a one-time pad. All hash functions are modeled as random oracles

deterministic encryption. The inner layer encrypts j into x using a minimalist authenticated IBE built from an implicit identity-based key agreement. The outer layer concurrently determines the value w that encrypts to the same x under a kind of anonymous IBE, derandomized to rely on the entropy already present in x. Bulk encryption uses a deterministic symmetric cipher with a one-time key.

It is helpful to observe that the exponentiations $\star^r$ and $\star^k$ used in `Sign` for commitment and in `Encrypt` for authenticated encryption, as well as the key extraction $\star^{msk}$, and the bilinear pairing $e(\star, i_R)$ that intervenes in the determination of w, all commute. The legitimate recipient derives its ability to decrypt x from the capacity to perform all of the above operations (either explicitly or implicitly), in a specific order, which is different from the order in which the sender performed the corresponding operations, but gives the same result.

10.4.2.2 Consistency and Security

The next theorem establishes that the scheme behaves as expected when operated by honest parties.

Theorem 10.1 *The IBSC scheme of Table 10.1 is consistent.*

Proof First, we show that the decryption of a honest ciphertext is correct. Observe that if $(\hat{x}, \hat{y}, \hat{z}) = (x, y, z)$, it follows that $\hat{w} = e({i_S}^{r\,k}, {i_R}^{msk}) = e({i_S}^{msk}, {i_R})^{r\,k} = w$ (in $\mathbb{G}_T$), and thus $\hat{v} = v$ and $(\hat{ID}_S, \hat{m}) = (ID_S, m)$; we also have $\hat{u} = e(\hat{i_S}, i_R)^{msk} = u$ (in $\mathbb{G}_T$), hence $\hat{k} = k$ (in $\mathbb{Z}_p$), and thus $\hat{j} = (j^k)^{\hat{k}^{-1}} = j$ (in $\mathbb{G}_1$).

Next, we show that the decrypted message/signature pair will pass the verification test. Indeed, if $(\hat{m}, \hat{ID}_S, \hat{j}, \hat{v}) = (m, ID_S, j, v)$, we have $e(g, \hat{v}) = e(g, i_S)^{msk\,(r+h)} = e(g^{msk}, (\hat{i_S})^h\,(\hat{i_S})^r) = e(g^{msk}, (\hat{i_S})^h\,\hat{j})$ (in $\mathbb{G}_T$), as required.

We now state without proof the security theorems corresponding to the five security properties given in Sect. 10.3. We refer the reader to the full version of [51] for the proofs.

Theorem 10.2 *Let $\mathcal{A}$ be a polynomial-time IND-IBSC-CCA attacker that has advantage at least ε and makes at most q_i queries to the random oracles H_i, $i = 0, 1, 2, 3, 4$. Then, there exists a polynomial-time algorithm $\mathcal{B}$ that solves the bilinear Diffie–Hellman problem with advantage at least $\varepsilon/(q_0\,q_2)$.*

Theorem 10.3 *Let $\mathcal{A}$ be an sUF-IBSC-CMA attacker that makes at most q_i queries to the random oracles H_i, $i = 0, 1, 2, 3, 4$, and at most q_{sc} queries to the signcryption oracle. Assume that, within a time span at most t, $\mathcal{A}$ produces a successful forgery with probability at least $\varepsilon = 10\,(q_{sc} + 1)\,(q_{sc} + q_1)/2^k$, for a security parameter k. Then, there exists an algorithm $\mathcal{B}$ that solves the bilinear Diffie–Hellman problem in expected time at most $120686\,q_0\,q_1\,t/\varepsilon$.*

Theorem 10.4 *There exists a deterministic polynomial-time `EncryptToSelf` algorithm that, given an identifier ID_S, a signed plaintext (m, j, v) issued by ID_S, and a private key d_R for an identity ID_R, creates a ciphertext (x, y, z) identical to the ciphertext that `Encrypt` would produce from (m, j, v) for identity ID_R. In particular, (x, y, z) decrypts to (m, j, v) under d_R with probability 1.*

Theorem 10.5 *Let $\mathcal{A}$ be a polynomial-time AUTH-IBSC-CMA attacker with advantage at least ε that makes at most q_i queries to the random oracles H_i, $i = 0, 1, 2, 3, 4$. Then, there exists a polynomial-time algorithm $\mathcal{B}$ that solves the BDH problem with advantage at least $2\,\varepsilon/(q_0\,(q_0-1)\,(q_1\,q_2+q_3))$.*

Theorem 10.6 *Let $\mathcal{A}$ be a polynomial-time ANON-IBSC-CCA attacker that has advantage at least ε and makes at most q_i queries to the random oracles H_i, $i = 0, 1, 2, 3, 4$. Then, there exists a polynomial-time algorithm $\mathcal{B}$ that solves the bilinear Diffie–Hellman problem with advantage at least $3\,\varepsilon/(q_0\,(q_0-1)\,(q_1\,q_2+2\,q_2+q_3))$.*

10.4.3 A Performance/Security Trade-Off

It is possible to optimize the previous scheme in various ways if one accepts to relax certain of its security properties.

For example, Chen and Malone-Lee [60] show how to achieve a 30% speed-up by removing some of the blinding and unblinding from the encryption and decryption functions, at the cost of dropping the unlinkability requirement.

We briefly describe the changes as follows:

- `Sign` is unchanged.
- `Encrypt` is simplified by dropping the computation of $x = j^k$ and outputting j instead, and using a hash of u^r instead of u^{kr} to blind (v, ID_S, m) in the output.
- `Decrypt` is likewise simplified by computing $\hat{u}^k = e(\hat{j}, d_R)$ instead of $\hat{w}$ and using it to unblind $(\hat{v}, \hat{ID}_S, \hat{m})$. The second pairing previously used to recover $\hat{j}$ is no longer necessary since $\hat{j}$ is now given in the ciphertext.
- `Verify` is unchanged. It takes the decrypted quadruple $(\hat{m}, \hat{j}, \hat{v}, \hat{ID}_S)$ as input.

With this modification, the resulting scheme is no longer unlinkable because the "decrypted" signature component $\hat{j}$, required by `Verify`, can be matched with the "encrypted" ciphertext component j, exposed by `Encrypt`.

10.4.4 Signcrypting for Multiple Recipients

It is often desirable to sign and encrypt the same message for multiple recipients. In this case, and especially if the message is a large data file, it is natural to ask whether the bulk of the signcryption can be performed once, with each recipient receiving identical ciphertexts except for some small recipient-specific header file.

In the scheme (as well as in the relaxed version), signcrypting the same message m for a set of n recipients $ID_{R_1}, \ldots, ID_{R_n}$ is easily achieved by carrying out the `Sign` operation once (which establishes the randomization parameter r), followed by an application of the `Encrypt` operation for each recipient identity, based on the same intermediate values.

Since the message m and the randomization parameter r are invariant for all the `Encrypt` instances, it is easy to see that the z component of the ciphertext also remains the same. Thus, the multi-recipient composite ciphertext is easily assembled from one instance of $(x_i, y_i) \in \mathbb{G}_1 \times \mathbb{G}_1$ for each recipient R_i, plus a single instance of $z \in \{0, 1\}^*$ to be shared by all. Thus, a multi-recipient ciphertext is compactly encoded in the form $C \leftarrow ((x_1, y_1), \ldots, (x_n, y_n), z)$. Since z is the only ciphertext component whose length depends on the message, this encoding results in a substantial economy of space.