A Brief Introduction to Certificateless Encryption Schemes and their Infrastructures

Alexander W. Dent

Information Security Group,
Royal Holloway, University of London, U.K.
a.dent@rhul.ac.uk

Abstract. Certificateless encryption is a form of public-key encryption that is designed to eliminate the disadvantages of both traditional PKI-based public-key encryption scheme and identity-based encryption. Unlike public-key encryption, there is no requirement for digital certificates or a public-key infrastructure. Unlike identity-based encryption, the trusted third party need not be given the ability to decrypt ciphertexts intended for users. In this invited paper we will review the concept of certificateless encryption from an infrastructure point of view and show that many of the different formulations for "certificateless" encryption can be instantiated using public-key infrastructures after all.

1 Introduction

Certificateless encryption is a type of public-key encryption which combines the advantages of traditional PKI-based public-key encryption and identity-based encryption [1, 2]. All three types of cryptosystem aim to transmit a message confidentially between a sender and receiver without the aid of shared secret keys. We approach the different types of primitive by considering the infrastructures needed to support them:

– In a public-key encryption scheme, a sender encrypts a message based on a public key which has been certified by a PKI [11]. The certificate binds the receiver's digital identifier with their public key. As well as performing the encryption operation, the sender must verify (at least) one digital signature on a certificate in order to verify the authenticity of the public key. This places a computational burden on the sender.
– In an identity-based encryption scheme, a sender encrypts a message based only on the digital identifier of the receiver [17]. This eliminates the (primary) need for a digital certificate. Unfortunately, identity-based encryption schemes have a systematic weakness. In order to obtain a valid decryption key for their digital identifier, the receiver must contact a key generation centre. This key generation centre can compute decryption keys for all the users in the system; the receiver has to trust that this third party will not abuse this ability to read confidential messages.

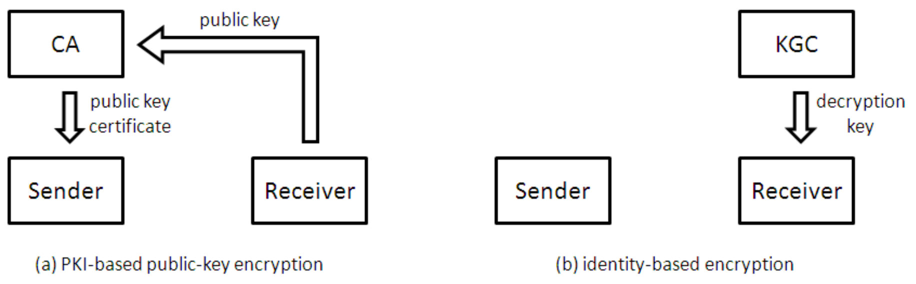

Fig. 1. The infrastructures for PKI-based public-key encryption (left) and identity-based encryption (right). In both cases it is assumed that every entity knows the other entities' digital identities.

The two architectures are shown in Figure 1.

Certificateless encryption schemes are characterised by two properties: (a) the scheme provides security without the need for a public key to be verified via a digital certificate, and (b) the scheme remains secure against attacks made by any third party (including a key generation centre or a certificate authority). This is achieved by having two public/private key pairs:

– A traditional public/private key pair generated by the receiver. The private key value is called a *secret value* to avoid confusion with the full private key of the scheme. The public key value is widely publicised but crucially is not authenticated with a digital certificate.
– An identity-based key pair consisting of the receiver's digital identifier and the associated identity-based private key supplied by a key generation centre. This private key is called a *partial private key*.

To encrypt a message, the sender uses the receiver's digital identifier and the receiver's public key. The receiver decrypts the ciphertext using the secret value generated by the receiver and the partial private key supplied by the key generation centre.

The intuition is that the sender does not require a digital certificate as the creation of a false public key for an identity will not help an attacker break the confidentiality of a transmitted message because the attacker does not know the partial private key for that identity. (This logic is similar to that of an identity-based encryption scheme.) The key generation centre cannot break the confidentiality of a transmitted message as it does not know the secret value corresponding to the receiver's public key. Of course, this makes the assumption that key generation centre will not publish a false public key for a receiver, but this attack seems unavoidable (and comparable to a CA publishing a false certificate for an identity).

The situation is complicated by a number of different infrastructures that can be put in place to support the distribution of the receiver's public key:

- **AP Formulation:** In the original Al-Riyami and Paterson (AP) formulation [1,2], the receiver can generate their public key at any time. This means that the receiver can publish their public key before receiving their partial private key from the key generation centre.
- **BSS Formulation:** In the Baek, Safavi-Naini and Susilo (BSS) formulation [4], the receiver can only generate their public key after receiving the partial private key. The partial private key is obtained via a single secure message from the key generation centre.
- **LK Formulation:** In the Lai and Kou (LK) formulation [13], the receiver can only generate their public key after completing a protocol with the key generation centre.

These three architectures are shown in Figure 2. The situation is further complicated by a series of complex and contradictory provable security models.

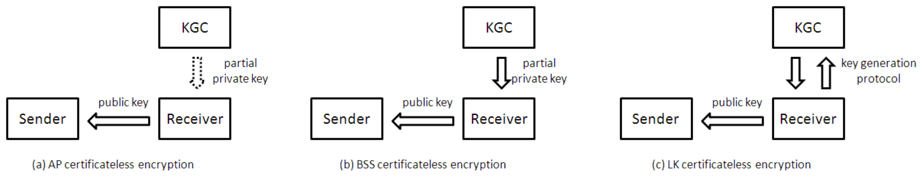

Fig. 2. The three certificateless encryption scheme architectures: (a) the AP formulation; (b) the BSS formulation; and (c) the LK formulation. The dotted arrow (in the AP formulation) denotes the fact that the public key can be published before the partial private key is obtained. In all cases, public keys are provided without a certificate. All entities are assumed to know the other entities' digital identities.

In this invited paper, we examine the relationship between certificateless encryption and other forms of public-key encryption. We show that in most cases, certificateless encryption infrastructures can be implemented using the very public-key infrastructure that they claim to eliminate. We will also briefly discuss security models.

2 Syntax and Infrastructure

The architecture for a certificateless encryption scheme involves three entities: a sender, a receiver, and a key generation centre (KGC). The syntax for a certificateless encryption scheme differs depending on the formulation. In all cases, the scheme is described by five probabilistic, polynomial-time (PPT) algorithms. We use to $\leftarrow$ to denote the assignment of the output of a deterministic algorithm (or fixed value) to a variable and $\overset{\$}{\leftarrow}$ to denote the assignment of the output of a probabilistic algorithm to a variable.

2.1 The AP Formulation

Al-Riyami and Paterson [1, 2] first defined a certificateless encryption scheme as a tuple of seven algorithms; however, a conceptually simpler five algorithm version has become widely accepted. In this version of the AP formulation, the schemes are defined by the following tuple of algorithms ($\mathtt{Setup}, \mathtt{Extract}, \mathtt{KeyGen}, \mathtt{Encrypt}, \mathtt{Decrypt}$). These form an infrastructure as follows:

- $\mathtt{Setup}(1^k)$: The setup algorithm is run by the KGC; it takes the security parameter as input and outputs a master public/private key pair $(mpk, msk) \overset{\$}{\leftarrow} \mathtt{Setup}(1^k)$. The master public key mpk is widely distributed; the master private key msk is kept secret by the KGC and used by the KGC to create partial private key values.
- $\mathtt{Extract}(msk, ID)$: The partial private key extraction algorithm is run by the KGC to create a partial private key for an identity ID. It takes as input the master private key msk and the identity ID, and outputs a partial private key $d \overset{\$}{\leftarrow} \mathtt{Extract}(msk, ID)$. This partial private key value is then sent (in a confidential manner) to the receiver.
- $\mathtt{KeyGen}(mpk, ID)$: The key generation algorithm is run by the receiver to create a key pair for that user. It takes as input the master public parameters for the scheme and the identity of the receiver, and outputs the key pair $(pk, sk) \overset{\$}{\leftarrow} \mathtt{KeyGen}(mpk, ID)$. The receiver widely publicises the public key pk, but keeps the secret value sk secret. We stress that the public key is not authenticated with a digital certificate.
- $\mathtt{Encrypt}(mpk, pk, ID, m)$: The encryption algorithm is used by the sender to send a message to the receiver. It takes as input the master public key of the system mpk, the receiver's public key pk, the receiver's digital identifier ID, and a message m drawn from some message space $\mathcal{M}$. It outputs either a ciphertext C in some ciphertext space $\mathcal{C}$ or an error symbol $\perp$ indicating that the public key was not valid for that identity.
- $\mathtt{Decrypt}(mpk, sk, d, C)$: The decryption algorithm is used by the receiver to decrypt a ciphertext. It takes as input the master public key of the system mpk, the receiver's secret value sk, the receiver's partial private key d, and a ciphertext $C \in \mathcal{C}$. It outputs either a message $m \in \mathcal{M}$ or the error symbol $\perp$ indicating that the ciphertext is invalid.

As you can see, since the receiver runs the `KeyGen` algorithm with public information as input, the receiver doesn't have to interact with the KGC before publishing their public key. Indeed, there is nothing to stop any user publishing a valid public key for any other user (see Section 3.3).

One interesting aspect of the AP formulation is that it implies the existence of both traditional PKI-based public-key encryption and identity-based encryption. To derive an identity-based encryption scheme from a certificateless encryption scheme, the receiver does not publish a public key. The sender instead generates a public/private key pair for the receiver by running the key generation algorithm `KeyGen` with a fixed random tape. The receiver can recover the associated secret value by running the key generation algorithm with the same random tape, but can only decrypt a message if it has received the partial private key value for that identity. Hence, we can immediately conclude that it is not possible to build a certificateless encryption scheme (within the AP formulation) from a trapdoor one-way permutation (in a black-box manner) [9]. The relationship between the AP formulation of certificateless encryption and public-key encryption was further investigated by Farshim and Warinschi [12].

2.2 The BSS Formulation

Unsurprisingly, due to its close relationship with identity-based cryptography, all of the certificateless encryption schemes which are designed in the AP formulation make use of elliptic curve pairings. Baek, Safavi-Naini and Susilo asked if it is possible to construct a certificateless encryption scheme without the use of elliptic curve pairings [4]. Their solution was to modify the architecture for a certificateless encryption scheme so that the receiver can not publish their public key until *after* they have obtained their partial private key value.

The BSS formulation is formally defined by five algorithms:

- $\texttt{Setup}(1^k)$: This algorithm is identical to the $\texttt{Setup}$ algorithm in the AP formulation. It is run by the KGC and produces a master key pair for the system $(mpk, msk) \overset{\$}{\leftarrow} \texttt{Setup}(1^k)$.
- $\texttt{Extract}(msk, ID)$: This algorithm is identical to the $\texttt{Extract}$ algorithm in the AP formulation. It is run by the KGC to obtain a partial private key $d \overset{\$}{\leftarrow} \texttt{Extract}(msk, ID)$ for an identity ID. This partial private key value is confidentially distributed the appropriate user.
- $\texttt{KeyGen}(mpk, ID, d)$: The key generation algorithm differs from the AP formulation in that it now takes the partial private key as input. It still outputs a key pair $(pk, sk) \overset{\$}{\leftarrow} \texttt{KeyGen}(mpk, ID, d)$ where the public key pk should be widely distributed and the private key sk should be kept secret. Notice that there is no concept of a secret value in this system; the output of the key generation algorithm is a full private key that can be used to decrypt ciphertexts. This is because the partial private key d can be included in the private key sk if necessary.
- $\texttt{Encrypt}(mpk, pk, ID, m)$: The encryption algorithm is identical to the $\texttt{Encrypt}$ algorithm in the AP formulation. It is run by the sender to create a ciphertext C which is then sent to the receiver.
- $\texttt{Decrypt}(mpk, sk, C)$: The decryption algorithm differs from the AP formulation in that it does not take the partial private key as an explicit input. The algorithm takes as input the master private key mpk, the receiver's private key sk, and a ciphertext $C \in \mathcal{C}$. It outputs either a message $m \in \mathcal{M}$ or the error symbol $\perp$.

The existing BSS certificateless encryption schemes are complex and have challenging security proofs. However, we will show that for the first time that secure BSS certificateless encryption can be derived using a PKI-based system. We will postpone a formal description of our new certificateless encryption scheme until after we introduce the certificateless security models. The basic idea, however, is very easy to understand. It is based on the concept of a certificate chain, with the key generation centre acting as the parent CA and each user acting as a subordinate CA which can only issue certificates that correspond to its own digital identity. In other words, the complete key generation process runs as follows:

1. As part of the `Setup` algorithm, the KGC generates a signature key pair. We call this the primary signature key pair. The primary public verification key is widely disseminated as part of the master public key *mpk*. The primary private signing key is kept secret as part of the master private key *msk*.
2. If the KGC wishes to produce a partial private key for the identity *ID*, then the `Extract` algorithm generates a secondary signature key pair and a digital certificate (signed using the primary signing key) which links the identity *ID* to the secondary public verification key. The partial private key contains the complete secondary key pair and the digital certificate.
3. The receiver's `KeyGen` algorithm creates a receiver public key by generating a standard (PKI-based) encryption key pair. The receiver generates a digital certificate for the public encryption key using the secondary signing key provided by the KGC. The receiver's complete public key contains the secondary verification key, the digital certificate for that key provided by the KGC, the public encryption key, and the digital certificate for that key computed by the receiver.

Now, if a sender wishes to send a message to the receiver, then the sender checks the authenticity of the public key by checking both certificates provided with the public encryption key (and only sends the message if both certificates verify correctly). The whole process is illustrated in Figure 3 and a more formal description will be given in Section 4.

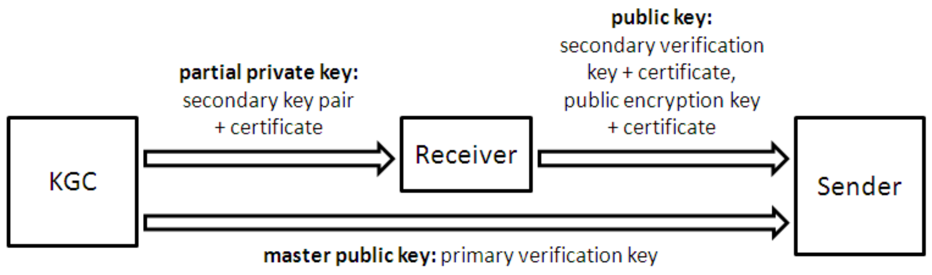
public key:
secondary verification key + certificate, public encryption key + certificate
partial private key:
secondary key pair + certificate
KGC
Receiver
Sender
master public key: primary verification key

Fig. 3. Public key distribution in the certificate-chain certificateless encryption scheme

2.3 The LK Formulation

The Lai-Kou formulation [13] can be viewed as a generalisation of the BSS formulation. Instead of a single message (the partial private key) being passed between the receiver and the KGC prior to public key publication, the receiver and the KGC must undertake a protocol before the receiver can publish its public key. Formally, it is defined by the following algorithms:

– `Setup`(1^k): This algorithm is identical to the `Setup` algorithm in the AP and BSS formulation. It is run by the KGC and produces a master key pair for the system $(mpk, msk) \overset{\$}{\leftarrow} \texttt{Setup}(1^k)$.
– `KGCKeyGen`(msk, ID) and `RecKeyGen`(mpk, ID): These two interactive algorithms define the protocol between the KGC (`KGCKeyGen`) and the receiver (`RecKeyGen`). These replace the `Extract` and `KeyGen` algorithms in the AP and BSS formulation. The KGC runs `KGCKeyGen` algorithm using the master private key *msk* and the receiver's identity *ID* as input; the receiver runs the `RecKeyGen` algorithm using the master public key and the receiver's identity as input. If the protocol is successfully completed then the receiver's algorithm (`RecKeyGen`) will output a user key pair (pk, sk). The KGC's algorithm (`KGCKeyGen`) has no output. The receiver then widely publicises the public key *pk* and keeps the private key *sk* secret.
– `Encrypt`(mpk, pk, ID, m): The encryption algorithm is identical to the `Encrypt` algorithm in the AP and BSS formulation. It is run by the sender to create a ciphertext C which is then sent to the receiver.
– `Decrypt`(mpk, sk, C): The decryption algorithm is identical to the `Decrypt` algorithm in the BSS formulation. It is run by the receiver to recover the message $m \in \mathcal{M}$ or the error symbol $\perp$.

It is easy to see that the traditional notion of PKI-based encryption can instantiate the LK formulation of certificateless encryption. The protocol interaction between the receiver and the KGC runs as follows:

1. The KGC's `Setup` algorithm generates a signature key pair and publishes the public verification key as part of the master public key *mpk*. The private signing key is kept secret as part of the master private key *msk*.
2. To generate a user key pair, the receiver generates an encryption key pair and sends the KGC the public key. (This is the first part of the `RecKeyGen` algorithm.)
3. The KGC then creates a digital certificate (signed using the KGC's private signing key) which binds the receiver's encryption key to their identity. This certificate is returned to the receiver (as part of the `KGCKeyGen` algorithm).
4. The receiver's full public key contains the public encryption key and the digital certificate for that key. This is computed as the final part of the `RecKeyGen` algorithm.

If a sender wishes to encrypt a message, then the sender first checks whether the certificate correctly authenticates the encryption key for the receiver's identity. This observation was made by Dent [10] who cited a security proof given by

Boldyreva *et al.* [7] in the context of public-key encryption schemes which incorporate the PKI into their security/efficiency models. Indeed, the LK formulation of a certificateless encryption scheme is so similar to Boldyreva *et al.* model of a public-key encryption scheme with PKI that they may effectively be considered one security model. As pointed out by Boldyreva *et al.*, the advantage of considering this enhanced security model is that it allows for the construction of public-key encryption schemes which are more efficient as a whole process (i.e. when the time spent verifying the correctness of the certificate is taken into account).

3 Security Models

One of the major problems with the development of certificateless encryption schemes has been the development of correct security models. These models should be powerful enough to demonstrate that the scheme resists all practical attacks, but not so powerful that they require overly complex and inefficient schemes in order to meet the security notions. The original models by Al-Riyami and Paterson [1, 2] made important conceptual decisions, but have been criticised for not reflecting the reality of a certificateless encryption scheme's usage scenario; the models are too strong in some aspects and too weak in others. Other security models have been suggested and a survey of these models was produced by Dent [10].

One important contribution by Al-Riyami and Paterson [1] was to split the security requirements into two separate models: the first concerns the security of the scheme against attacks made by an outsider and the second concerns the security of the scheme against attacks made by the KGC. These are traditionally called Type I and Type II attacks respectively, although efforts are being made to change the nomenclature to something more descriptive.

The reason for the dual security model is to account for a trivial attack that can be made by the KGC. The attacker is trying to break the confidentiality of a message that is sent by a sender to a receiver. Since there are no (explicit) digital certificates in the system, the sender has no guarantee that he has an authentic copy of the receiver's public key. In other words, our models have to cope with a situation in which the attacker convinces the sender to use a receiver public key generated by the attacker. Recall that the schemes should resist attacks of this form made by an outside attacker (as such an attacker would not know the partial private key for the receiver). However, a certificateless encryption scheme can never resist such an attack if it is made by the KGC. By definition, the KGC can compute all partial private keys; hence, it can always replace the public key with one for which it knows the underlying secret value and therefore decrypt all ciphertexts computed by the sender. This gives rise to two security modes:

1. The outsider (or Type I) security model. The attacker is allowed to replace the public key that the sender uses to encrypt messages.

2. The KGC (or Type II) security model. The attacker is not allowed to replace the public key that the sender uses to encrypt messages, but can compute the master public key value maliciously.

We give formal security models for the BSS formulation of a certificateless encryption scheme as these will be used to prove the security of the certificate chain scheme described in Section 2.2. We will mostly want to show that an attacker's "advantage" is negligible, where the term negligible means that the success probability falls away faster than the reciprocal of any polynomial (as a function of the security parameter). Technically, a function f is negligible if for all polynomials p there exists a constant $N(p)$ such that $f(k) \leq 1/|p(k)|$ for all $k \geq N(p)$.

3.1 Outsider Attacks

This security model is designed to show that an outside attacker cannot break the confidentiality of the scheme unless it somehow obtains a user's partial private key *and* replaces the public key with one which has been maliciously generated.

A security model is typically presented as a game played between an arbitrary (probabilistic polynomial-time) attacker and a challenger (who represents the system with which the attacker interacts). Crucially, the challenger keeps a list of users in the system, their real public/private key pairs, and the public key value that the sender associates with each user. The attacker interacts with the system via a series of oracles which force the challenger to perform certain operations and model the different ways that the attacker can interact with the system. The attacker is modelled as a pair of PPT algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and the security game is as follows:

1. The challenger generates a master key pair $(mpk, msk) \overset{\$}{\leftarrow} \texttt{Setup}(1^k)$.
2. The attacker runs $\mathcal{A}_1$ on mpk. $\mathcal{A}_1$ may query the following oracles:
   - **Request Public Key**: This oracle takes an identity ID as input and generates a full public/private key (pk, sk) for the identity ID using the `Extract` and `KeyGen` algorithms. The oracle returns the public key pk. The oracle also records (pk, sk) as ID's real public/private key and pk as the public key that the sender associated with ID.
   - **Replace Public Key**: This oracle takes an identity ID and a public key pk' as input. The oracle changes the records so that the sender now associates the public key pk' with the identity ID.
   - **Extract Partial Private Key**: This oracle takes an identity ID as input and outputs ID's partial private key $d \overset{\$}{\leftarrow} \texttt{Extract}(msk, ID)$.
   - **Decrypt**: This oracle takes as input an identity ID and a ciphertext C. It outputs $m \leftarrow \texttt{Decrypt}(mpk, sk, C)$ where sk is the private key for ID computed during the "Request Public Key" query.

   $\mathcal{A}_1$ terminates with the output of an identity ID^*, two equal-length messages (m_0, m_1), and some state information ω.

3. The challenger randomly generates a bit $b \xleftarrow{\$} \{0,1\}$ and computes the challenge ciphertext $C^* \xleftarrow{\$} \texttt{Encrypt}(mpk, pk', ID^*, m_b)$ where pk' is the public key that the sender associates with the identity ID^*.
4. The attacker runs $\mathcal{A}_2$ on the challenge ciphertext C^* and the state information ω. $\mathcal{A}_2$ may query the same oracles as in the first phase of its execution. It terminates with the output of a bit b'.

The attacker can trivially win this security game if:

- The attacker replaces the public key of ID^* in Step 2 *and* requests the partial private key of ID^* at any time (as the attacker can then compute a full decryption key for ID^*).
- The attacker does not replace the public key of ID^* in Step 2 *and* requests the decryption of the challenge ciphertext C^* by ID^* in Step 4 (as this trivially returns the message m_b).

The attacker wins the game if it outputs $b' = b$ without performing these trivial attacks. The attacker's advantage is defined to be $Adv^{\text{out}}_{\mathcal{A}}(k) = |\Pr[b = b'] - 1/2|$ and the scheme is said to be *outsider secure* if this advantage is negligible.

One interesting quirk of the original outsider (Type I) security model is that if the attacker replaced the public key of an identity and then queried the decryption oracle, then the decryption oracle would decrypt the ciphertext using the private key corresponding to the replaced public key rather than the original public key. This security model is widely believed not to reflect an attacker's real-life capabilities and we encourage the use of the simpler model.

3.2 KGC Attacks

The security model for KGC attacks is slightly simpler than the model for outsider attacks as the KGC is forbidden from replacing public keys. This means that the challenger does not have to keep track of the public keys that the sender believes are associated with each user. Originally, the security model for KGC attacks still used a "correctly generated" master public key *mpk* [1, 2]; however, Au *et al.* noted that this does not reflect the reality of a malicious key generation centre and allowed the KGC to generate their master public key in an adversarial manner [3]. This model was later refined by Dent [10].

The formal model involves a PPT attacker $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ and a hypothetical challenger. The security game runs as follows:

1. The attacker generates the master public key and some state information $(mpk, \omega) \xleftarrow{\$} \mathcal{A}_0(1^k)$.
2. The attacker runs $\mathcal{A}_1$ on the state information ω. $\mathcal{A}_1$ may query the following oracles:
   - **Request Public Key**: This oracle takes an identity *ID* and a partial private key *d* as input. It computes a public/private key pair $(pk, sk) \xleftarrow{\$} \texttt{KeyGen}(mpk, ID, d)$ and returns *pk*.

  - **Decrypt**: This oracle takes an identity ID and a ciphertext C as input, and returns $m \leftarrow \texttt{Decrypt}(mpk, sk, C)$ where sk is the private key for identity ID.

   The attacker terminates with the output of an identity ID^*, two equal-length messages (m_0, m_1), and some state information ω.
3. The challenger generates a bit $b \xleftarrow{\$} \{0,1\}$ and computes the challenge ciphertext $C^* \xleftarrow{\$} \texttt{Encrypt}(mpk, pk, ID^*, m_b)$ where pk is the public key associated with identity ID^*.
4. The attacker runs $\mathcal{A}_2$ on the challenge ciphertext C^* and the state information ω. $\mathcal{A}_2$ may query the **Request Public Key** and **Decrypt** oracles as above. $\mathcal{A}_2$ terminates with the output of a bit b'.

The attacker may trivially break the scheme if $\mathcal{A}_2$ queries the Decrypt oracle for ID^* with C^*. The attacker wins the game if it outputs $b' = b$ and does not perform this trivial attack. The attacker's advantage is defined to be $Adv_{\mathcal{A}}^{\texttt{KGC}}(k) = |\Pr[b = b'] - 1/2|$ and the scheme is said to be KGC secure if this advantage is negligible.

3.3 Denial of Decryption Attacks

Another attractive feature that we may require from a certificateless encryption is that it prevents denial of decryption attacks. Liu, Au and Susilo [14] were the first to notice that a certificateless encryption scheme didn't prevent a sender from encrypting a message using an "incorrect" public key — i.e. a public key which does not correspond to the identity ID for which the message is intended. This was termed a "denial of decryption" or "DoD" attack as an attacker that convinces a sender to use an incorrect public key denies the receiver the opportunity to decrypt the message.

At one end of the spectrum, a certificateless encryption scheme in the AP formulation can never achieve this notion security. Since the `KeyGen` algorithm does not depend on a secret information known only to the user with identity ID, anybody can run the `KeyGen` algorithm to create a valid public key for ID. On the other end of the spectrum, a traditional PKI-based public-key encryption scheme (which can be viewed as an example of the LK formulation of a certificateless encryption scheme — see Section 2.3) resists these attacks. The minimum requirement for a certificateless encryption scheme to achieve denial of decryption security is that it is expressed in the BSS or LK formulations.

The formal model of security for denial of decryption attacks is designed to capture the notion that the attacker cannot convince a sender that a false public key is correct unless it encrypts message in such a way that it can still be decrypted correctly by the legitimate receiver with the original key pair. It is formally described as a game played between a PPT attacker $\mathcal{A}$ and a challenger:

1. The challenger generates a master key pair $(mpk, msk) \xleftarrow{\$} \texttt{Setup}(1^k)$.
2. The attacker runs $\mathcal{A}$ on mpk. $\mathcal{A}$ may query **Request Public Key**, **Replace Public Key**, **Extract Partial Private Key** and **Decrypt** oracles as in

the outsider security model (see Section 3.1). $\mathcal{A}$ terminates with the output of an identity ID^* and a message m^*.

The attacker wins if $C^* \overset{\$}{\leftarrow} \texttt{Encrypt}(mpk, pk', ID^*, m^*)$ satisfies $C^* \neq \perp$ and $m^* \neq \texttt{Decrypt}(mpk, sk, C^*)$ where the encryption operation is performed with the public key pk' that the sender associates with the identity ID^* and the decryption operation is performed with the original private key sk that was generated during the Request Public Key query. The scheme is said to be DoD secure if the probability $Adv_{\mathcal{A}}^{\texttt{DoD}}(k)$ that an attacker wins is negligible.

4 BSS Certificateless Encryption Based on a PKI

In this section, we will describe the certificate-chain certificateless encryption scheme briefly discussed in Section 2.2. This scheme demonstrates that a PKI-based public-key encryption scheme can be used to instantiate a BSS certificateless encryption scheme. Since this construction combines a digital signature scheme and a traditional public-key encryption scheme, we begin by formally defining these primitives.

4.1 Digital Signature Schemes

A digital signature scheme is a triple of algorithms $(\mathcal{G}_s, \mathcal{S}, \mathcal{V})$. The key generation algorithm $\mathcal{G}_s$ takes the security parameter 1^k as input and outputs a key pair $(pk, sk) \overset{\$}{\leftarrow} G_s(1^k)$. The signing algorithm $\mathcal{S}$ takes a message m and the private key sk as input, and outputs a signature $\sigma \overset{\$}{\leftarrow} \mathcal{S}(sk, m)$. The verification algorithm $\mathcal{V}$ takes a message m, a signature σ, and the public key pk as input. It outputs either a symbol $\top$ to indicate the signature is valid or a symbol $\perp$ to indicate the signature is invalid.

We require that the digital signature scheme is sUF-CMA secure. This means that it should be infeasible for an attacker to find a new signature on any message (even if the attacker has previously obtained a signature on that message). The security model considers a PPT attacker $\mathcal{A}$ playing the following game:

1. The challenger generates a key pair $(pk, sk) \overset{\$}{\leftarrow} \mathcal{G}_s(1^k)$.
2. The attacker runs $\mathcal{A}$ on the input pk. $\mathcal{A}$ may query a signing oracle with a message m and the oracle will return $\sigma \overset{\$}{\leftarrow} \mathcal{S}(sk, m)$. $\mathcal{A}$ terminates with the output of a message m^* and a signature σ^*.

The attacker wins if $\mathcal{V}(pk, m^*, \sigma^*) = \top$ and it did not query the signing oracle with the message m^* and receive the signature σ^* in response. The scheme is sUF-CMA secure if the probability $Adv_{\mathcal{A}}^{\texttt{sig}}(k)$ of the attacker winning the game is negligible.

4.2 Public-Key Encryption Schemes

A public-key encryption scheme is a triple of PPT algorithms $(\mathcal{G}_e, \mathcal{E}, \mathcal{D})$. The key generation algorithm $\mathcal{G}_e$ takes as input a security parameter 1^k and outputs a key pair $(pk, sk) \xleftarrow{\$} \mathcal{G}_e(1^k)$. The encryption algorithm $\mathcal{E}$ takes as input a message $m \in \mathcal{M}$ and the public key pk, and outputs a ciphertext $C \in \mathcal{C}$. The decryption algorithm $\mathcal{D}$ takes as input a ciphertext $C \in \mathcal{C}$ and the private key sk, and outputs either a message $m \in \mathcal{M}$ or the error symbol $\perp$.

We require the IND-CCA2 notion of security for the encryption scheme. This captures the notion that no attacker can determine any information about a message from a ciphertext even if they can obtain the decryptions of any other ciphertext. This is formalised via the following security game played between a PPT attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and a hypothetical challenger:

1. The challenger generates a key pair $(pk, sk) \xleftarrow{\$} \mathcal{G}_e(1^k)$.
2. The attacker runs $\mathcal{A}_1$ on the public key pk. $\mathcal{A}_1$ may query a decryption oracle with any ciphertext $C \in \mathcal{C}$. The oracle returns $\mathcal{D}(sk, C)$. $\mathcal{A}_1$ terminates with the output of two equal-length messages (m_0, m_1) and some state information ω.
3. The challenger generates $b \xleftarrow{\$} \{0, 1\}$ and computes the challenge ciphertext $C^* \xleftarrow{\$} \mathcal{E}(pk, m_b)$.
4. The attacker runs $\mathcal{A}_2$ on the challenge ciphertext C^* and the state information ω. $\mathcal{A}_2$ may query the decryption oracle as before with the exception that $\mathcal{A}_2$ may not query the decryption oracle on C^*. $\mathcal{A}_2$ terminates with the output of a bit b'.

The attacker wins the game if $b = b'$. The attacker's advantage is defined to be $Adv_{\mathcal{A}}^{\mathsf{enc}}(k) = |\Pr[b = b'] - 1/2|$. The scheme is said to be IND-CCA2 secure if every PPT attacker has negligible advantage.

4.3 The Certificate-Chain BSS Certificateless Encryption Scheme

We now formally present the BSS certificateless encryption scheme based on certificate chains discussed in Section 2.2. The scheme makes use of a digital signature scheme $(\mathcal{G}_s, \mathcal{S}, \mathcal{V})$ and a public-key encryption scheme $(\mathcal{G}_e, \mathcal{E}, \mathcal{D})$. It is described in Figure 4.

The scheme provides outsider security (Section 3.1), KGC security (Section 3.2), and denial of decryption security (Section 3.3). This is summarised by the following three theorems:

Theorem 1. *Suppose there exists an attacker $\mathcal{A}$ against the certificateless encryption scheme in the outsider security model which makes at most q_{req} queries to the Request Public Key oracle. Then there exists an attacker $\mathcal{B}$ against the first instance of the signature scheme, an attacker $\mathcal{B}'$ against the second instance of the signature scheme, and an attacker $\mathcal{B}^*$ against the public-key encryption scheme such that*

$$Adv_{\mathcal{A}}^{out}(k) \leq Adv_{\mathcal{B}}^{sig}(k) + q_{req} Adv_{\mathcal{B}'}^{sig}(k) + q_{req} Adv_{\mathcal{B}^*}^{enc}(k)\,. \qquad (1)$$

```
$\mathtt{Setup}(1^k)$:
  $(mpk, msk) \xleftarrow{\$} \mathcal{G}_s(1^k)$
  Return $(mpk, msk)$

$\mathtt{Extract}(msk, ID)$:
  $(pk_s, sk_s) \xleftarrow{\$} \mathcal{G}_s(1^k)$
  $m_1 \leftarrow ID \| pk_s$
  $cert_1 \xleftarrow{\$} \mathcal{S}(msk, m_1)$
  $d \leftarrow (pk_s, sk_s, cert_1)$
  Return $d$

$\mathtt{KeyGen}(mpk, ID, d)$:
  Parse $d$ as $(pk_s, sk_s, cert_1)$
  $(pk_e, sk_e) \xleftarrow{\$} \mathcal{G}_e(1^k)$
  $m_2 \leftarrow pk_e$
  $cert_2 \xleftarrow{\$} \mathcal{S}(sk_s, m_2)$
  $pk \leftarrow (pk_s, cert_1, pk_e, cert_2)$
  $sk \leftarrow sk_e$
  Return $(pk, sk)$

$\mathtt{Encrypt}(mpk, pk, ID, m)$:
  Parse $pk$ as $(pk_s, cert_1, pk_e, cert_2)$
  $m_1 \leftarrow ID \| pk_s$
  If $\mathcal{V}(mpk, m_1, cert_1) = \perp$ then
    Return $\perp$
  $m_2 \leftarrow pk_e$
  If $\mathcal{V}(pk_s, m_2, cert_2) = \perp$ then
    Return $\perp$
  $C \xleftarrow{\$} \mathcal{E}(pk_e, m)$
  Return $C$

$\mathtt{Decrypt}(sk, C)$:
  $m \leftarrow \mathcal{D}(sk, C)$
  Return $m$
```

Fig. 4. The Certificate-Chain BSS Certificateless Encryption Scheme

Theorem 2. *Suppose there exists an attacker $\mathcal{A}$ against the certificateless encryption scheme in the KGC security model which makes at most q_{req} queries to the Request Public Key oracle. Then there exists an attacker $\mathcal{B}$ against the public-key encryption scheme such that*

$$Adv_{\mathcal{A}}^{\text{KGC}}(k) \leq q_{req} Adv_{\mathcal{B}}^{\text{enc}}(k) \,. \tag{2}$$

Theorem 3. *Suppose there exists an attacker $\mathcal{A}$ against the denial of decryption security of the certificateless encryption scheme which makes at most q_{req} queries to the Request Public Key oracle. Then there exists an attacker $\mathcal{B}$ against the first instance of the digital signature scheme and an attacker $\mathcal{B}'$ against the second instance of the digital signature scheme such that*

$$Adv_{\mathcal{A}}^{\text{DoD}}(k) \leq Adv_{\mathcal{B}}^{\text{sig}}(k) + q_{req} Adv_{\mathcal{B}'}^{\text{sig}}(k) \,. \tag{3}$$

The proofs of these theorems are given in the full version of the paper but all of the proofs essentially rely on two observations:

– If the attacker does not replace the public key of the identity ID^* then the attacker is essentially attacking the IND-CCA2 security of the public-key encryption scheme.
– In order to replace the public key of an identity then the attacker has to forge either $cert_1$ or $cert_2$. This is an attack against the sUF-CMA security of the digital signature scheme.

The proof of Theorem 1 can be adapted to show that the scheme is secure in the original outsider (Type I) security model of Al-Riyami and Paterson [1, 2]. Nonetheless, we prove the theorem in the (weaker) outsider security model described in Section 3.1 as we believe that this is the appropriate security model for all practical applications.

Since it is possible to construct both public-key encryption schemes [6, 16] and digital signature schemes [5] from trapdoor one-way permutations, this construction demonstrates that it is possible to construct BSS certificateless encryption schemes from trapdoor one-way permutations in a black-box manner. This is in contrast to AP certificateless encryption schemes which cannot be constructed from trapdoor one-way permutations in a black-box manner [9]. The state of the art in digital signature and public-key encryption schemes suggests that BSS certificateless encryption schemes can be efficiently constructed without elliptic curve pairings.

We also note that there is no requirement for the two “certificates” to be computed using the same signature scheme. Indeed, the security proof allows for the second signature scheme to be a one-time signature scheme. Furthermore, the schemes becomes more bandwidth efficient if an aggregate signature scheme [8] or sequential aggregate signature scheme [15] is used to compress $cert_1$ and $cert_2$ into a single signature. However, the use of such schemes will reduce the computational efficiency and so their use should be considered a trade-off between computational and bandwidth requirements.

Acknowledgements

I would like to thank the programme chairs (Fabio Martinelli and Bart Preneel) for the opportunity to speak at this conference and Pooya Farshim for his comments on the paper. This is an invited paper and has not been refereed by external reviewers or the programme committee. Any mistakes should be solely attributed to me. This work has been supported by the European Commission through the IST Program under Contract ICT-2007-216646 ECRYPT II.

References

1. S. Al-Riyami. *Cryptographic schemes based on elliptic curve pairings*. PhD thesis, Royal Holloway, University of London, 2004. Available from `http://www.isg.rhul.ac.uk/~kp/sattthesis.pdf`.
2. S. S. Al-Riyami and K. G. Paterson. Certificateless public key cryptography. In C. S. Laih, editor, *Advances in Cryptology – Asiacrypt 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 452–473. Springer-Verlag, 2003.
3. M. H. Au, J. Chen, J. K. Liu, Y. Mu, D. S. Wong, and G. Yang. Malicious KGC attack in certificateless cryptography. In *Proc. ACM Symposium on Information, Computer and Communications Security*. ACM Press, 2007.
4. J. Baek, R. Safavi-Naini, and W. Susilo. Certificateless public key encryption without pairing. In J. Zhou and J. Lopez, editors, *Proceedings of the 8th International Conference on Information Security (ISC 2005)*, volume 3650 of *Lecture Notes in Computer Science*, pages 134–148. Springer-Verlag, 2005.

5. M. Bellare and S. Micali. How to sign given any trapdoor function. *Journal of the ACM*, 39(1):214–233, 1992.
6. M. Bellare and M. Yung. Certifying permutations: Non-interactive zero-knowledge based on any trapdoor permutation. *Journal of Cryptology*, 9(1):149–166, 1996.
7. A. Boldyreva, M. Fischlin, A. Palacio, and B. Warinschi. A closer look at PKI: Security and efficiency. In T. Okamoto and X. Wang, editors, *Public Key Cryptography – PKC 2007*, volume 4450 of *Lecture Notes in Computer Science*, pages 458–475. Springer-Verlag, 2007.
8. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In E. Biham, editor, *Advances in Cryptology – Eurocrypt 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 416–432. Springer-Verlag, 2003.
9. D. Boneh, P. A. Papkonstantinou, C. Rackoff, Y. Vahlis, and B. Waters. On the impossibility of basing identity based encryption on trapdoor permutations. In *Proc. of the 49th Annual IEEE Symposium on Foundations of Computer Science – FOCS 2008*, pages 283–292, 2008.
10. A. W. Dent. A survey of certificateless encryption schemes and security models. *International Journal of Information Security*, 7(5):349–377, 2008.
11. W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22:644–654, 1976.
12. P. Farshim and B. Warinschi. Certified encryption revisited. In B. Preneel, editor, *Progress in Cryptology – Africacrypt 2009*, volume 5580 of *Lecture Notes in Computer Science*, pages 179–197. Springer-Verlag, 2009.
13. J. Lai and K. Kou. Self-generated-certificate public key encryption without pairing. In T. Okamoto and X. Wang, editors, *Public Key Cryptography – PKC 2007*, volume 4450 of *Lecture Notes in Computer Science*, pages 476–489. Springer-Verlag, 2007.
14. J. K. Liu, M. H. Au, and W. Susilo. Self-generated-certificate public key cryptography and certificateless signature/encryption scheme in the standard model. In *Proc. ACM Symposium on Information, Computer and Communications Security*. ACM Press, 2007.
15. A. Lysyanskaya, S. Micali, L. Reyzin, and H. Shacham. Sequential aggregate signatures from trapdoor permutations. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology – Eurocrypt 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 74–90. Springer-Verlag, 2004.
16. A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th Annual Symposium on Foundations of Computer Science, FOCS '99*, pages 543–553. IEEE Computer Society, 1999.
17. A. Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and D. Chaum, editors, *Advances in Cryptology – Crypto '84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer-Verlag, 1984.