International Journal of Information Security manuscript No.
(will be inserted by the editor)

Alexander W. Dent

A Survey of Certificateless Encryption Schemes and Security Models

December 14, 2007

Abstract This paper surveys the literature on certificateless encryption schemes. In particular, we examine the large number of security models that have been proposed to prove the security of certificateless encryption schemes and propose a new nomenclature for these models. This allows us to "rank" the notions of security for a certificateless encryption scheme against an outside attacker and a passive key generation centre, and we suggest which of these notions should be regarded as the "correct" model for a secure certificateless encryption scheme.

We also examine the security models that aim to provide security against an actively malicious key generation centre and against an outside attacker who attempts to deceive a legitimate sender into using an incorrect public key (with the intention to deny the legitimate receiver that ability to decrypt the ciphertext). We note that the existing malicious key generation centre model fails to capture realistic attacks that a malicious key generation centre might make and propose a new model.

Lastly, we survey the existing certificateless encryption schemes and compare their security proofs. We show that few schemes provide the "correct" notion of security without appealing to the random oracle model. The few schemes that do provide sufficient security guarantees are comparatively inefficient. Hence, we conclude that more research is needed before certificateless encryption schemes can be thought to be a practical technology.

1 Introduction

In 1978, Rivest, Shamir and Adleman [26] proposed the first public-key encryption scheme. This scheme was a concrete realisation of a seemingly paradoxical conjecture of Diffie and Hellman [17]: that it was possible for an entity (the sender) to securely send another entity (the receiver) a message without these two entities having a pre-existing shared secret key, without any online contact between them, and even without the receiver knowing that they were about to receive a message. This functionality is achieved by generating a pair of keys instead of just one: a public key that is widely distributed for encryption, and a related private key that is kept secret and used for decryption.

Information Security Group, Royal Holloway,
Egham Hill, Egham, Surrey, U.K.
E-mail: a.dent@rhul.ac.uk

History, though, has shown that public key encryption has significant practical problems. In particular, the sender has to be sure that the public key that they have is the correct public key for the receiver. Hence, we require a public key infrastructure — a series of trusted third parties that can be relied upon to check a receiver's identity and vouch for the connection between that identity and a particular public key. Public key management is the most costly, cumbersome and inefficient part of any framework that makes use of public key cryptography.

One way of avoiding the tiresome need for a public key infrastructure is to use an identity-based encryption scheme [27]. The most a public key infrastructure can do is to confirm a link between a digital identity and a public key, where a digital identity is some bitstring that uniquely identifies the user in some context. An identity-based encryption scheme removes the need for a public key infrastructure by setting an entity's public key to be equal to their digital identity. Of course, in such a situation, an entity cannot be expected to compute their own private key; hence, there must exist a trusted third party who initially sets up the system and uses their secret knowledge of the system to compute private keys for other entities.

Identity-based encryption seems to remove the need for a public key infrastructure, replacing it with the need for a key generation centre that computes a user's private key for them. This is more efficient, but has a significant disadvantage too. The fact that the trusted third party computes the private decryption keys for the users means that that trusted third party can read the messages of every user in the system. There are also significant practical problems associated with identity-based encryption, including the problem of handling key revocation.

In 2003, Al-Riyami and Paterson proposed a new type of encryption scheme that avoids the drawbacks of both traditional public-key encryption and identity-based encryption [3]. They termed this new type of encryption *certificateless public-key encryption* (CL-PKE) because their encryption scheme did not require a public key infrastructure. Roughly speaking, their idea was to combine the functionality of a public key scheme with that of an identity based scheme. Hence, to encrypt a message, a sender requires both the receiver's identity and a public key value produced by the receiver. Similarly, to decrypt a ciphertext, a receiver requires the partial private key corresponding to their identity (which is given to them by a key generation centre) and the private key corresponding to the distributed public key.

This paper surveys and extends the known results about certificateless encryption schemes. In particular, we survey the existing security models, propose slight changes to these models so that they may better reflect reality (where applicable), and propose a new nomenclature that can be used to clarify the contradictory definitions that are currently prominent in the area. We also survey the known certificateless encryption schemes, particularly with respect to the models in which they claim security. Lastly we propose a new certificateless encryption scheme based on the Dodis-Katz [18] multiple encryption scheme and prove the security of this scheme in an appropriate model. This resolves the folklore question of whether a Dodis–Katz-based certificateless encryption scheme can be constructed.

2 Security Models For CL-PKE

This section will examine the various security models proposed for a certificateless public-key encryption scheme. This has been an area of some controversy. As we shall see, the original security definitions proposed by Al-Riyami and Paterson [2,3] are very strong, and have been criticised for not realistically reflecting an attacker's capabilities. Furthermore, we suggest that Al-Riyami and Paterson's security definitions are not consistent in their strength, i.e. that a certificateless scheme is held to a higher standard of security with regard to Type I attackers, than the standard to which it is held with regard to Type II attackers. We then survey the most common relaxations of Al-Riyami and Paterson's security notions and propose a new nomenclature. We also examine the recent proposals to strengthen the existing security models to cope with malicious KGCs and to solve the problems associated with distributing public keys reliably.

2.1 CL-PKE Schemes

2.1.1 The Al-Riyami and Paterson Formulation

The notion of a certificateless public-key encryption scheme was introduced by Al-Riyami and Paterson [2,3]. A certificateless public-key encryption scheme is defined by seven probabilistic, polynomial-time algorithms:

- `Setup`: This algorithm takes as input a security parameter 1^k, and returns the master private key *msk* and the master public key *mpk*. This algorithm is run by a KGC in order to initially set up a certificateless system.
- `Extract-Partial-Private-Key`: This algorithm takes as input the master public key *mpk*, the master private key *msk*, and an identity $ID \in \{0,1\}^*$. It outputs a partial private key d_{ID}. This algorithm is run by a KGC once for each user, and the corresponding partial private key is distributed to that user in a suitably secure manner.
- `Set-Secret-Value`: This algorithm takes as input the master public key *mpk* and an entity's identity *ID*, and outputs a secret value $x_{ID} \in \mathscr{S}$ for that identity. This algorithm is run by the user. Note that the secret value space $\mathscr{S}$ is somehow defined by the master public key *mpk* and an entity's identity *ID*.
- `Set-Private-Key`: This algorithm takes the master public key *mpk*, an entity's partial private key d_{ID} and an entity's secret value $x_{ID} \in \mathscr{S}$ as input. It outputs the full private key sk_{ID} for that user. This algorithm is run once by the user.
- `Set-Public-Key`: This algorithm takes as input the master public key *mpk* and an entity's secret value $x_{ID} \in \mathscr{S}$. It output a public key $pk_{ID} \in \mathscr{PK}$ for that user. This algorithm is run once by the user and the resulting public key is widely and freely distributed. The public-key space $\mathscr{PK}$ for a particular user is defined by the master public key *mpk* and the user's identity *ID*.
- `Encrypt`: This algorithm takes as input the master public key *mpk*, a user's identity *ID*, a user's public key $pk_{ID} \in \mathscr{PK}$ and a message $m \in \mathscr{M}$. It outputs either a ciphertext $C \in \mathscr{C}$ or the error symbol $\perp$. Note that the message space $\mathscr{M}$ and the ciphertext space $\mathscr{C}$ are somehow defined by a combination of the master public key *mpk*, the user's public key pk_{ID} and the user's identity *ID*.
- `Decrypt`: This algorithm takes as input the master public key *mpk*, a user's private key sk_{ID} and a ciphertext $C \in \mathscr{C}$. It returns either a message $m \in \mathscr{M}$ or the error symbol $\perp$.

A certificateless public-key encryption scheme allows anybody to encrypt a message for a particular receiver using publicly available information (in exactly the same way as a traditional public-key encryption scheme or an identity-based encryption scheme). However, unlike a traditional public-key encryption scheme, no certificates are needed. This is because an attacker who publishes a false public key pk_{ID} for an identity will still not be able to decrypt messages encrypted under that public key, because the key generation centre will not release the partial private key d_{ID} to the attacker and so the attacker will not be able to compute the full private key.

This functionality is also given by identity-based cryptography, but identity-based encryption schemes have the disadvantage that the key generation centre can always decrypt messages. A certificateless public-key encryption scheme does not have this disadvantage. An honest-but-curious KGC can always compute an identity's partial private key d_{ID}, but since they will not know the secret value x_{ID} associated with that entity's public key pk_{ID}, they will not be able to form the full private key either. Of course, a malicious KGC that masquerades as a user by publishing a false public key can still break the security of the system; however, it is unclear how to prevent this threat and we will not consider it any further.

2.1.2 An Equivalent Method For Constructing Public Keys

The `Set-Secret-Value` and `Set-Public-Key` algorithms may be replaced with a single `Set-User-Keys` algorithm that works as follows:

– `Set-User-Keys`: This algorithm takes as input the master public key *mpk* and an entity's identity *ID*, and outputs a secret value x_{ID} and a public key pk_{ID} for that identity. This algorithm is run once by the user.

This is functionally equivalent to the formulation given by Al-Riyami and Paterson. It is easy to see that a scheme presented in the original way can also be presented in this new form: the new `Set-User-Keys` algorithm is defined to be the algorithm that executes the original `Set-Secret-Value` algorithm and `Set-Public-Key` algorithm.

A little bit of thought is required to show that a certificateless scheme presented in this new formulation can also be presented in the old formulation. Suppose that the `Set-User-Keys` algorithm takes as input $p(k)$ random bits. We define the `Set-Secret-Value` algorithm to be the algorithm that outputs a set of $p(k)$ random bits, x'_{ID}. The `Set-Public-Key` algorithm is defined to be the algorithm that takes the random bits x'_{ID} as input, runs `Set-User-Keys` to determine a public/private key pair, and outputs the public key. Similarly, `Set-Private-Key` is defined to be the algorithm that runs `Set-User-Keys` using the random bits x'_{ID} to determine the 'proper' secret value x_{ID} and then uses this to create a private key in the normal way.

Several authors have suggested that the `Set-User-Keys` algorithm removes the need to have private keys at all. In such a situation, the decryption algorithm is defined as taking both the partial private key d_{ID} and the secret value x_{ID} as input. This is functionally the same as the normal formulation of certificateless encryption; however, this can have a significant impact on the security models, which typically assume that an attacker can obtain partial private keys and private keys, but not secret values. This new formulation implicitly assumes that one uses security models that have Extract Secret Value oracles (see Section 2.5).

2.1.3 The Baek, Safavi-Naini and Susilo Formulation

The only significant departure from the Al-Riyami and Paterson formulation of a certficateless encryption scheme was proposed by Baek, Safavi-Naini and Susilo [7]. In this model, a public key can only be computed after a partial private key has been obtained[1]. In other words, they make a slight change to the `Set-Public-Key` algorithm:

– `Set-Public-Key`: This algorithm takes as input the master public key *mpk*, an entity's partial private key d_{ID} and secret value x_{ID}. It outputs a public key pk_{ID} for that user. This algorithm is run once by the user and the resulting public key is widely distributed.

This slight change allows Baek, Safavi-Naini and Susilo to propose a certificateless encryption scheme based on the CDH problem alone (i.e. without requiring elliptic curve pairings)[2]. The only slight drawback of this formulation is that it does not allow messages to be encrypted "into the future". Under the Al-Riyami and Paterson formulation, an entity may publish a public key pk_{ID} without necessarily knowing the partial private key, and therefore they may receive messages that they cannot decrypt until the KGC releases the partial private key to them. However, the Baek, Safavi-Naini and Susilo formulation requires that the entity obtains their partial private key before releasing their full public key; hence, an entity that releases a public key must necessarily have enough information to compute the full private key. On the other hand, it should be noted that only certificateless encryption schemes that use the Baek, Safavi-Naini and Susilo formulation can resist "denial of decryption" attacks (see Section 2.7).

It should also be noted that, using this formulation, we can combine the `Set-Secret-Value`, `Set-Public-Key` and `Set-Private-Key` algorithms into a single `Set-User-Keys` algorithm in a manner similar to that discussed in the previous section. The new algorithm behaves as follows:

– `Set-User-Keys`: This algorithm takes as input the master public key *mpk*, an entities identity *ID* and a partial private key d_{ID} as input, and outputs a public/private key pair (pk_{ID}, sk_{ID}) or the error symbol $\perp$.

This leads to a very efficient formulation of certificateless encryption, in which a scheme consists of only five algorithms:

– `Setup`,
– `Extract-Partial-Private-Key`,
– `Set-User-Keys`,
– `Encrypt`, and
– `Decrypt`.

This formulation has no concept of a secret value, which makes some security models (including the Weak Type Ia model) inappropriate.

[1] Technically, the Baek *et al.* model proposed that the KGC return a partial public key (used to help compute the full public key) and a partial private key (used to help compute the full private key). However, the security model they provided is equivalent to the one given here.

[2] This scheme does not currently have a security proof, but there are no known attacks against the scheme either. For more details, see Section 3.3

2.2 The General Security Model

The security of a certificateless encryption scheme is expressed by two (very similar) games. In this section we will describe a basic framework. In both cases, an attacker $\mathscr{A} = (\mathscr{A}_1, \mathscr{A}_2)$ is trying to break the IND-CCA2 security of the scheme, the formal model describing confidentiality. The game runs as follows:

1. The challenger generates a master key pair $(mpk, msk) = \mathtt{Setup}(1^k)$.
2. The attacker executes $\mathscr{A}_1$ on *mpk* and (possibly) some extra information *aux*. During its execution $\mathscr{A}_1$ may have access to certain oracles (described subsequently). $\mathscr{A}_1$ terminates by outputting an identity ID^*, two messages of equal length (m_0, m_1), and some state information *state*.
3. The challenger randomly chooses a bit $b \in \{0,1\}$ and computes the challenge ciphertext

   $$C^* = \mathtt{Encrypt}(mpk, ID^*, pk_{ID^*}, m_b) \tag{1}$$

   using the value of pk_{ID^*} currently associated with the identity ID^*. If the public key pk_{ID^*} does not exist, then the challenger computes a public key pk_{ID^*} for ID^* by running the `Set-Secret-Value` and `Set-Public-Key` algorithms.
4. The attacker executes $\mathscr{A}_2$ on the input $(C^*, state)$. During its execution $\mathscr{A}_2$ may have access to certain oracles (described subsequently). $\mathscr{A}_2$ terminates by outputting a guess b' for b.

The attacker wins the game if $b = b'$ and its advantage is defined to be:

$$Adv_{\mathscr{A}} = |Pr[b = b'] - 1/2| \tag{2}$$

It now remains to define the oracles that the attacker may have access to:

– **Request Public Key**: The attacker supplies an identity *ID* and the challenger responds with the public key pk_{ID} for *ID*. If the identity *ID* has no associated public key, then the challenger generates a public key for *ID* by running `Set-Public-Key` and `Set-Secret-Value` (as necessary).
– **Replace Public Key**: This oracle models the attacker's ability to convince a legitimate sender to use an invalid public key. This can happen because public keys are no longer verified by a trusted third party, and a user may be given a false public key by an attacker and believe it to be correct. The attacker supplies an identity *ID* and a valid public key value $pk_{ID} \in \mathscr{PK}$, and the challenger replaces the current public key value with the value pk_{ID}.
– **Extract Partial Private Key**: The attacker supplies an identity *ID* and the challenger responds with the partial private key d_{ID}. If the identity has no partial private key, then the challenger generates a partial private key by running `Extract-Partial-Private-Key` on *ID* using *msk*.
– **Extract Private Key**: The attacker supplies an identity *ID* and the challenger responds with the private key sk_{ID}. If the identity has no associated private key, then the challenger generates one using `Set-Private-Key` (after running the `Set-Secret-Value` algorithm and the `Extract-Partial-Private-Key` algorithm as necessary).

An attacker may also have access to one or more different types of decryption oracle:

– **Strong Decrypt**: The attacker supplies an identity *ID* and a ciphertext *C*, and the challenger responds with the decryption of $C \in \mathscr{C}$ under the private key sk_{ID}. Note that if the attacker has replaced the public key for *ID*, then this oracle should return the correct decryption of *C* using the private key that inverts the public key pk_{ID} currently associated with the identity *ID* (or $\perp$ if no such private key exists).
– **Weak SV Decrypt**: The attacker supplies an identity *ID*, a secret value $x_{ID} \in \mathscr{S}$, and a ciphertext $C \in \mathscr{C}$. The challenger computes the full private key sk_{ID} for the identity from the (correct) partial private key d_{ID} and the supplied secret value x_{ID}, then returns the decryption of *C* under this private key sk_{ID}. If either process fails, then the oracle returns $\perp$. Note that this functionality can be achieved by a strong decryption oracle.
– **Decrypt**: The attacker supplies an identity *ID* and a ciphertext $C \in \mathscr{C}$, and the challenger responds with the decryption of *C* under the original private key sk_{ID} for *ID*. Note that this functionality can be achieved by a strong decryption oracle.

The Weak SV Decrypt oracle is so named as the attacker chooses the secret value which is to be combined with the correct partial private key to give the full private key to be used for decryption. We make this explicit to differentiate the Weak SV Decrypt oracle from a second weak decryption oracle which will be introduced later.

It should be noted that the Weak SV Decrypt oracle can only be simulated by a strong decrypt oracle when the Al-Riyami and Paterson formulation for certificateless encryption schemes is used (see Section 2.1.1 and 2.1.2). If the Baek, Safavi-Naini and Susilo formulation is used then a Strong Decrypt oracle cannot necessarily simulate the response of a Weak SV Decrypt oracle as it is necessary to know both the partial private key and secret value for an identity before computing an identity's full public key. This may result in certain "strong" security models needing to be being altered to allow attackers explicit access to a Weak SV Decrypt oracle when proving the security of certificateless encryption schemes presented using the Baek, Safavi-Naini and Susilo formulation.

A certificateless scheme is proven secure by showing that any attacker attempting to break the scheme only has a negligible chance of success.

Definition 1 (Negligible Function) A function $f : \mathbb{N} \to \mathbb{R}$ is said to be negligible if, for all polynomial p, there exists an integer $N(p)$ such that $|f(x)| \leq 1/|p(x)|$ for all $x \geq N(p)$.

2.3 Type I Attackers

The Type I security model is designed to protect against a third party attacker (i.e. anyone except the legitimate receiver or the KGC) who is trying to gain some information about a message from its encryption. There has been some debate about how to precisely formulate this notion and we survey the main attempts in this section. We also comment on their correctness, and provide a new and consistent nomenclature for the different notions.

2.3.1 Strong Type I Security

The original definition proposed by Al-Riyami and Paterson [3] is as follows.

Definition 2 A certificateless encryption scheme is Strong Type I secure if every probabilistic, polynomial-time attacker $\mathscr{A}^I = (\mathscr{A}_1^I, \mathscr{A}_2^I)$ has negligible advantage in winning the IND-CCA2 game subject to the following oracle constraints:

- $\mathscr{A}^I$ cannot extract the private key for the challenge identity ID^* at any time,
- $\mathscr{A}^I$ cannot extract the private key of any identity for which it has replaced the public key,
- $\mathscr{A}^I$ cannot extract the partial private key of ID^* if $\mathscr{A}^I$ replaced the public key pk_{ID^*} before the challenge was issued,
- $\mathscr{A}_2^I$ cannot query the Strong Decrypt oracle on the challenge ciphertext C^* for the identity ID^* unless the public key pk_{ID^*} used to create the challenge ciphertext has been replaced,
- $\mathscr{A}^I$ cannot query the Weak SV Decrypt or Decrypt oracles (although this functionality can be given by the Strong Decrypt oracle).

In this model, the attacker is given no extra information, i.e. *aux* is the empty bit-string.

This model gives as much power as possible to the attacker. Its only restrictions are those necessary to prevent attacks that would trivially allow the attacker to win, and to prevent the attacker from asking for the private key of a user with a replaced public key. This latter restriction was only made because it was considered too difficult to achieve a notion of security that doesn't have this restriction.

It should be noted that the model expects the challenger to be able to correctly respond to decryption queries made on identities for which the attacker has replaced the public key. This is a very strong notion of security and it is unclear whether it represents a realistic attack scenario. In general, decryption oracles are provided to an attacker to model the fact that the attacker may be able to gain some information from a legitimate receiver about the decryptions of some ciphertexts (for example, by bribing the legitimate receiver to give up the message or by deducing whether a ciphertext is a valid encryption of a particular message by observing the legitimate receiver's behaviour after receiving the ciphertext). This situation cannot happen if we replace a public key: when we replace a public key, we are duping a *sender* into encrypting a message using a false public key that the receiver has not published. Under no circumstances will the receiver then attempt to decrypt that ciphertext using the private key corresponding to that replaced public key. Hence, providing a decryption oracle that will accurately decrypt ciphertexts encrypted under the replaced public key gives the attacker more power than it would have in practice.

This represents an interesting philosophical question in the construction of security models: do we give the attacker as much power as is possible (perhaps subject to the restriction that we must still be able to construct secure certificateless encryption schemes)? Or should the model only try to reflect a realistic attacker's abilities? The former approach leads to strong security models, and potentially more complex schemes. The latter approach may lead to more efficient schemes, but a scheme's security can only be guaranteed if an attacker's abilities have been correctly modelled. We shall return to this issue in Section 4.1.

2.3.2 Weak Type Ia Security

Several authors have judged Al-Riyami and Paterson's Type I security model to be too strong and proposed weaker versions. We will consider each of the major alternatives in turn. The strongest of these definitions, which we will term Weak Type Ia security, has been put forward by Bentahar *et al.* [10].

Definition 3 A certificateless encryption scheme is Weak Type Ia secure if every probabilistic, polynomial-time attacker $\mathscr{A}^I$ has negligible advantage in winning the IND-CCA2 game subject to the following oracle constraints:

- $\mathscr{A}^I$ cannot extract the private key for the challenge identity ID^* at any time,
- $\mathscr{A}^I$ cannot extract the private key of any identity for which it has replaced the public key,
- $\mathscr{A}^I$ cannot extract the partial private key of ID^* if $\mathscr{A}^I$ replaced the public key pk_{ID^*} before the challenge was issued,
- $\mathscr{A}^I$ cannot query the Strong Decrypt oracle at any time,
- $\mathscr{A}_2^I$ cannot query the Weak SV Decrypt oracle on the challenge ciphertext C^* for the identity ID^* if the attacker replaced the public key pk_{ID^*} before the challenge was issued.
- $\mathscr{A}_2^I$ cannot query the Decrypt oracle on the challenge ciphertext C^* for the identity ID^* unless the attacker replaced the public key pk_{ID^*} before the challenge was issued.

In this model, the attacker is given no extra information, i.e. *aux* is the empty bit-string.

It should be noted that the original notion of Weak Type Ia security [10] did not give the attacker the ability to request decryptions using the original private key value *after* the public key had been replaced. We make this small change to make the model more realistic. It does not affect the results presented by Bentahar *et al.*

The Weak Type Ia model seems to most realistically reflect the potential abilities of an attacker. The attacker can replace public keys with arbitrary values of its choice, thus allowing for senders to be duped, but the attacker can still ask a legitimate receiver to decrypt any ciphertext with his original private key value (using the Decrypt oracle). Furthermore, the attacker may be able to dupe a legitimate receiver into changing his public key and secret value to that of the attacker's choice (using a combination of the Replace Public Key oracle and the Weak SV Decrypt oracle), and so obtain encryptions and decryptions using keys formed from arbitrary secret values.

2.3.3 Weak Type Ib Security

A weakening of this model gives Weak Type Ib security [31]:

Definition 4 A certificateless encryption scheme is Weak Type Ib secure if every probabilistic, polynomial-time attacker $\mathscr{A}^I$ has negligible advantage in winning the IND-CCA2 game subject to the following oracle constraints:

- $\mathscr{A}^I$ cannot extract the private key for the challenge identity ID^* at any time,
- $\mathscr{A}^I$ cannot extract the private key of any identity for which it has replaced the public key,
- $\mathscr{A}^I$ cannot extract the partial private key of ID^* if $\mathscr{A}^I$ replaced the public key pk_{ID^*} before the challenge was issued,
- $\mathscr{A}^I$ cannot query the Strong Decrypt or Weak SV Decrypt oracles,
- $\mathscr{A}_2^I$ cannot query the Decrypt oracle on the challenge ciphertext C^* and the identity ID^* unless the attacker replaced the public key pk_{ID^*} before the challenge was issued.

In this model, the attacker is given no extra information, i.e. *aux* is the empty bit-string.

In this model, the attacker can replace public keys (i.e. dupe senders) and can ask for decryptions of ciphertexts using the original private key values, but cannot dupe a recipient into decrypting messages using a secret value chosen by the attacker. This reflects security in a situation where users generate their public key values correctly (i.e. by using the `Set-Secret-Value` and `Set-Public-Key` algorithms) and never change their public key values once they are set.

Another interpretation of the difference between Weak Type Ia and Weak Type Ib security is based on the implementation of a certificateless scheme in a black-box device. The Weak Type Ib security model guarantees security in the case when the black box is tamper-proof in its generation of the secret value; the Weak Type Ia security model guarantees that the scheme is secure when the black-box can be forced to re-generate its secret key value and may possibly be influenced in the way that this occurs (for example, by side-channel attacks).

2.3.4 Weak Type Ic Security

Lastly, mostly for comparison with Type II attackers, we present a final weak notion of security. This model of security was briefly considered in an early version of a paper by Baek and Wang [8]. This notion of security can be achieved by a public-key encryption scheme alone.

Definition 5 A certificateless encryption scheme is Weak Type Ic secure if every probabilistic, polynomial-time attacker $\mathscr{A}^I$ has negligible advantage in winning the IND-CCA2 game subject to the following oracle constraints:

- $\mathscr{A}^I$ cannot replace any public keys at any time,
- $\mathscr{A}^I$ cannot extract the private key for the challenge identity ID^* at any time,
- $\mathscr{A}^I$ cannot query the Strong Decrypt or Weak SV Decrypt oracles,
- $\mathscr{A}_2^I$ cannot decrypt the challenge ciphertext C^* for the identity ID^*.

In this model, the attacker is given no extra information, i.e. *aux* is the empty bit-string.

The different types of oracle access that these models give to an attacker is summarised in Table 1. It is simple to deduce the following relationships between the different notions of Type I security:

$$\begin{aligned}\text{Strong Type I} &\Rightarrow \text{Weak Type Ia}\\ &\Rightarrow \text{Weak Type Ib} \Rightarrow \text{Weak Type Ic}\end{aligned}$$

where $A \Rightarrow B$ if any scheme that is A secure must necessarily be B secure.

2.3.5 The partial private key of the challenge identity

There are further variations on these security models. Several schemes are proven secure in a weakened model in which a Type I attacker is not allowed to query the partial private key extraction oracle on the challenge identity ID^*. We denote these models with an asterisk; for example, the Weak Type Ib* model is exactly the same as the Weak Type Ib security model except that the attacker is not allowed to query the partial private key extraction oracle on the challenge identity.

This security model also has a natural interpretation. It assumes that the attacker is unable to get hold of an identity's partial private keys except in specialised cases (for example, for the attacker's own identity or where an entity has been completely corrupted). This can be achieved by a system in which the KGC delivers the partial private key through some confidential channel. Since, in practice, we would expect partial private keys to be delivered confidentially, this is not an unreasonable assumption. However, it does mean that the confidentiality of the partial private keys becomes paramount. Of course, an attacker that can obtain a partial private key for an individual can always masquerade as that individual by publishing a false public key, but normally that attacker would be unable to compromise messages that were

Table 1 A Summary of the Oracle Access Provided to a Type I Attacker

	Request Public Key	Replace Public Key	Extract Private Key[1]	Extract Partial Private Key	Strong Decrypt	Weak SV Decrypt	Decrypt
Strong Type I	✓	✓	✓	✓	✓		
Weak Type Ia	✓	✓	✓	✓		✓	✓
Weak Type Ib	✓	✓	✓	✓			✓
Weak Type Ic	✓		✓	✓			✓

[1] This oracle is optional if using asymptotic security models – see Section 2.5.2

sent under the correct public key (including old messages). A model which expressly forbids querying the extract partial private key oracle on the challenge identity presents no such guarantees. Hence, wherever possible, this model should not be used.

2.4 Type II Attackers

The second security definition is designed to capture the notion that an honest-but-curious key generation centre should not be able to break the confidentiality of the scheme. Here we allow the attacker to have access to master private key by setting $aux = msk$. This means that we do not have to give the attacker explicit access to an Extract Partial Private Key oracle, as they are able to compute these value for themselves. The most important point about Type II security is that the KGC can trivially break the scheme if it is allowed to replace the public key for the challenge identity *before* the challenge is issued.

2.4.1 Weak Type II Security

Al-Riyami and Paterson [3] chose to prevent the trivial key-replacement attack from occurring by forbidding the KGC from replacing any public keys at all, proposing the following model:

Definition 6 A certificateless encryption scheme is Weak Type II secure if every probabilistic, polynomial-time attacker $\mathscr{A}^{II} = (\mathscr{A}_1^{II}, \mathscr{A}_2^{II})$, which is given the auxiliary information $aux = msk$, has negligible advantage in winning the IND-CCA2 game subject to the following oracle constraints:

- $\mathscr{A}^{II}$ cannot extract the private key for the challenge identity ID^* at any time,
- $\mathscr{A}^{II}$ cannot query the Extract Partial Private Key oracle at any time,
- $\mathscr{A}^{II}$ cannot replace public keys at any time,
- $\mathscr{A}^{II}$ cannot query the Strong Decrypt or Weak SV Decrypt oracles at any time,
- $\mathscr{A}_2^{II}$ cannot query the Decrypt oracle on the challenge ciphertext C^* and the identity ID^*.

This roughly corresponds to the weakest notion of security proposed for Type I attackers, and it is easy to see that any scheme that is Weak Type II secure is necessarily Weak Type Ic secure. Furthermore, this notion of security can be achieved by a public key encryption scheme alone, i.e. a scheme which contains no identity-based component and in which the user simply publishes a public key. In such a situation, it is easy to see that the above definition of Weak Type II security corresponds exactly to the "multi-user" definition of IND-CCA2 security.

Al-Riyami and Paterson justify the Weak Type II security model by noting that the KGC has no more power than a certificate authority in a traditional PKI: if a malicious certificate authority is allowed to replace public keys, by producing certificates on false public key values, then that certificate authority can easily break the security of any public key scheme using a man-in-the-middle attack. Thus, any model of security against a malicious CA must necessarily forbid that CA from replacing public keys. Al-Riyami and Paterson simply extend this restriction to the KGC in a certificateless scheme.

There is one important difference between the CA architecture and the certificateless architecture: in the certificateless architecture it may not be possible to detect the actions of a KGC that is publishing false public keys. If a CA replaces a public key, then the CA necessarily leaves evidence of its crime – it must publish a certificate for the false public key that is signed with the CA's private key. For a certificateless scheme, the KGC may simply publish a new public key for a user without leaving any evidence to link that public key to the KGC's fraud. We note that in the Baek, Safavi-Naini and Susilo formulation, it may be possible to prove that a KGC committed a fraudulent act in a similar manner to the CA case, as it might be the case that only an entity in possession of the partial private key can compute a valid public key. Since only the KGC and legitimate users are assumed to have knowledge of user's partial private key, any valid but fraudulent public key value must have been published by the KGC. The topic of whether these attacks are possible or not is related to the subject of denial of decryption (see Section 2.7).

2.4.2 Strong Type II Security

The Weak Type II model prevents the attacker from replacing public keys. However, by denying the KGC the ability to replace public keys or query more powerful decryption oracles, we might be denying it the ability to perform certain attacks that might occur in practice, and we are certainly not providing it with the huge level of power provided to a Strong Type I attacker. Hence, we should consider whether the KGC gains any advantages if we allow it to replace public keys (subject to the restriction that it cannot replace the public key of the challenge identity until after the challenge has been issued) or allow it access to more powerful decryption oracles.

Clearly, unless we permit the attacker to access a specialised decryption oracle, the ability to replace is public keys is useless to an attacker. This is because the attacker cannot replace the challenge public key before the challenge ciphertext is issued; hence, the challenger never gives the attacker any information based on a replaced public key value. The weak decryption oracle is of no use to an attacker because the attacker can always compute the full private key of a user given their identity ID and their secret value x_{ID} themselves. Hence, all of the Weak Type II security models that we might propose (based on the Weak Type I security models) are equivalent.

However, if we follow the principle that we should give the attacker as much power as possible, then there is some merit in considering a Strong Type II security model. This gives an equivalent security level for the scheme against Type II attackers as is demanded for Type I attackers. It is unreasonable to require a scheme to meet the Strong Type I security level, without also requiring to meet the Strong Type II security level.

Definition 7 A certificateless encryption scheme is Strong Type II secure if every probabilistic, polynomial-time attacker $\mathscr{A}^{II} = (\mathscr{A}_1^{II}, \mathscr{A}_2^{II})$, which is given the auxiliary information $aux = msk$, has negligible advantage in winning the IND-CCA2 game subject to the following oracle constraints:

– $\mathscr{A}^{II}$ cannot extract the private key for the challenge identity ID^* at any time,
– $\mathscr{A}^{II}$ cannot extract the private key of any identity for which it has replaced the public key,
– $\mathscr{A}^{II}$ cannot query the Extract Partial Private Key oracle at any time,
– $\mathscr{A}_1^{II}$ cannot output a challenge identity ID^* for which it has replaced the public key,
– $\mathscr{A}_2^{II}$ cannot query the Strong Decrypt oracle on the challenge ciphertext C^* for the identity ID^* unless the public key pk_{ID^*} used to create the challenge ciphertext has been replaced,
– $\mathscr{A}^{II}$ cannot query the Weak SV Decrypt or Decrypt oracles (although this functionality can be given by the Strong Decrypt oracle).

The different types of oracle access that these models give to an attacker is summarised in Table 2.

2.5 Secret Value Oracles

2.5.1 Type I Attackers

Cheng and Comley [14] have proposed a variation on the above security models. In the Cheng and Comley variation, the attacker is not given access to an Extract Private Key oracle, but is instead given access to an Extract Secret Value oracle:

– **Extract Secret Value**: The attacker supplies an identity ID and the challenger responds with the secret value x_{ID} associated with that entity. If the identity has no associated secret value, then the challenger generates one by running `Set-Secret-Value`.

Of course, there are certain trivial attacks that can be performed by an attacker with access to an Extract Secret Value oracle which must be excluded. Therefore, we add the following oracle conditions when using a Type I security model with an Extract Secret Value oracle:

– $\mathscr{A}^{I}$ cannot extract the secret value of any identity for which it has replaced the public key,
– $\mathscr{A}^{I}$ cannot query both the Extract Partial Private Key oracle and the Extract Secret Value oracle on the challenge identity,
– $\mathscr{A}_2^{I}$ cannot query the Weak SV Decrypt oracle on the challenge ciphertext C^* for the identity ID^* and using the secret value x_{ID^*} if the attacker has queried the Extract Secret Value oracle for the public key used to create the challenge ciphertext and receive the response x_{ID^*}.

The last condition only applies to the Weak Type Ia security model (see Definition 3).

We denote a security model in which the attacker has access to an Extract Secret Value oracle using a dagger; for example, the Weak Type Ib† model is exactly the same as the Weak Type Ib security model except that the attacker has access to an Extract Secret Value oracle instead of an Extract Private Key oracle.

This change gives rise to slightly more powerful security models. The attacker can simulate an Extract Private Key oracle by making queries to both the Extract Partial Private Key and Extract Secret Value oracles, and then assembling the full private key itself. However, the attacker now has the ability to find out the secret value associated with a public key (and, in particular, the secret value associated with the challenge identity). This is not an ability that the attacker is guaranteed to have in the normal security models.

It is interesting to note that only schemes with deterministic `Set-Public-Key` algorithms can achieve security against attackers with access to a Extract Secret Value oracle. If the `Set-Public-Key` algorithm is probabilistic, or if it is possible to find two public keys that a sender will recognise as valid for a single secret value, then the scheme must be weak. The attacker simply requests pk_{ID} for an identity of their choice, then recovers x_{ID} using the Extract Secret Value oracle, computes a distinct public key $pk'_{ID} \neq pk_{ID}$ using the `Set-Public-Key` algorithm, and replaces the public key for ID with pk'_{ID}. The attacker can now legitimately decrypt the challenge ciphertext C^* using the Decrypt oracle and recover the underlying message.

Table 2 A Summary of the Oracle Access Provided to a Type II Attacker

	Request Public Key	Replace Public Key	Extract Private Key[1]	Extract Partial Private Key[2]	Strong Decrypt	Weak SV Decrypt[2]	Decrypt
Strong Type II	✓	✓	✓		✓		
Weak Type II	✓		✓				✓

[1] This oracle is optional if using asymptotic security models – see Section 2.5.2
[2] This oracle is unnecessary as the attacker may compute its function without querying the oracle

2.5.2 Type II Attackers

It is possible to allow the use of Extract Secret Value oracles for Type II attackers too. In this case, we must apply the following oracle conditions to exclude trivial attacks:

– $\mathscr{A}^{II}$ cannot query the Extract Secret Value oracle on the challenge identity ID^* at any time,
– $\mathscr{A}^{II}$ cannot extract the secret value of any identity for which it has replaced the public key.

However, it can be shown an Extract Secret Value oracle will not significantly help a Type II attacker. Consider an attacker that has access to an Extract Secret Value oracle. We may simulate that oracle for that attacker if we correctly guess the Request Public Key oracle query that defines the value of the public key used during the challenge phase. We respond to all other Request Public Key queries (i.e. all queries that are not about the challenge public key) by generating the public/private key pair ourselves. We respond to the attacker's oracle queries as follows:

– The Replace Public Key oracle and the Strong Decrypt oracle can be used as normal as they ignore the original public key values.
– The Extract Secret Value oracle can be simulated by returning the secret value used during the original key generation. Note that by definition, the attacker cannot query the Extract Secret Value oracle on the challenge identity.
– For all identities except for the challenge identity, the Decrypt oracle can be simulated by using the original private key to decrypt ciphertexts. The original Decrypt oracle can be used to handle Decrypt oracle queries for the challenge identity.

Note that here we have completely simulated the Extract Secret Value oracle. Therefore the oracle may be removed from the security model at the cost of guessing which Request Public Key query defines the value of the challenge public key. This is similar to the argument that states that the "multi-user" IND-CCA2 security for public key encryption is equivalent to the "single-user" IND-CCA2 security model. A similar argument can also be used to show that the Extract Private Key oracle does not significantly help a Type I or Type II attacker. We choose to leave these oracles in the security models in order not to disguise the loss of concrete security given by the necessity of guessing the challenge public key.

2.5.3 Practical Considerations

It can be argued that allowing the attacker access to an Extract Secret Value oracle does not reflect reality. In a normal mode of use, a secret value will be used to generate a public/private key pair for an entity and then deleted. In such a scenario, it does not seem likely that an attacker will be able to extract the secret value at any point. However, it might be possible for an attacker to persuade a receiver to give up some information about his secret value or to obtain some information about a secret value as it is generated (for example, using some form of side-channel analysis). Hence, whenever possible, it is prudent to prove the security of a CL-PKE in a model that allows access to an Extract Secret Value oracle, simply to provide a 'margin of error' for the security model.

Extract Secret Value oracles should definitely be included in any model that attempts to model the situation where certificateless encryption is used to encrypt messages 'into the future'. In such a situation, we have to protect against a curious receiver (who knows his own secret value) who wishes to read a message before being issued his partial private key.

Obviously, in the Baek, Safavi-Naini and Susilo formulation, in which the concept of secret values can be eliminated, it does not make much sense to give the attacker access to an Extract Secret Value oracle.

2.6 Security Against Malicious KGCs

The original Type II security model only ever considered an honest-but-curious KGC. The models assume that the KGC generates its keys in complete accordance with the `Setup` algorithm, including deleting any data used during the setup procedure but not explicitly contained within the master keys. This does not necessarily reflect reality when we consider a KGC that is attempting to break user confidentiality. A new model was proposed by Au *et al.* [5] to overcome this deficiency.

The model of Au *et al.* allows the attacker to choose the master public key, subject to the restriction that it comes from some recognisable set $\mathscr{MPK}$ defined by the certificateless scheme and containing at least all possible master public keys that could be output by the `Setup` algorithm. This presents a choice as to which oracles the attacker should be allowed access. Since the master public key is produced by the attacker, and so only the attacker would know the underlying master private key, or even whether such a key exists, it would be difficult to provide the attacker with access to any oracle which would usually require knowledge of an entity's partial private key to function. Hence, the attacker is not given access to Extract Partial Private Key, Extract Private Key, Weak SV Decrypt or Decrypt oracles. (It is possible to define an attack game in which the oracles are expected to respond to oracle queries even without knowing the underlying partial private key. This is consistent with the definition of a strong decryption oracle, which forces the decryption oracle to compute decryptions of ciphertexts without necessarily knowing the private key. However, Au *et al.* chose not to follow this approach. This is consistent with the decision of Al-Riyami and Paterson not to allow an attacker to be able to query the extract private key/extract secret value oracle on a replaced public key.)

However, there are still problems that need to be solved. The first of which is that by denying the attacker the ability to query either the Extract Partial Private Key and Extract Private Key oracles, the attacker has no way of corrupting a user and learning their long term private key. One way to compensate for this is to insist that the attack model allow the attacker to query an Extract Secret Value oracle (see Section 2.5). However, this solution only seems valid in models which make use of secret values. It may be less valid in the Baek, Safavi-Naini and Susilo model, in which the concept of a secret value can be eliminated. In such a situation, one has to consider consider extracting a private key belonging to a user that has been given a specific partial private key value. It is possible that a malicious KGC might send a user a specific partial private key, which the user will use to form a public/private key pair, before the malicious KGC in some way corrupts the user and obtains the private key value. In other words, it may be considered necessary to allow the attacker access to the following oracle:

– **Extract Private Key From PPK**: The attacker supplies and identity ID and a partial private key d_{ID}, and the oracle returns both the public and private key values computed for that identity using the given partial private key.

We note, however, that the attacker should not be able to use a public key value derived in this way as the challenge public key, or the attacker would be to trivially win the game. This means that the Extract Private Key From PPK oracle can be simulated by an attacker that simply generates a secret value, and computes these public and private keys itself. Hence, it is not necessary to include this oracle explicitly in the security model.

There is another type of attack that must be considered. In the current security model, the attacker is unable to obtain decryptions of ciphertexts, as the implementation of any decryption oracle requires knowledge of a partial private key. This seems to run contrary to the spirit of an IND-CCA2 definition of security. Furthermore, we may apply the same logic as in the previous paragraphs to this situation and imagine situation in which a malicious KGC sends a user a specific partial private key, which, when combined with their secret value, might leak information that aids an attacker via a decryption oracle. In other words, we need to propose a new weak decryption oracle:

– **Weak PPK Decrypt**: The attacker supplies an identity ID, a partial private key d_{ID}, and a ciphertext C. The challenger computes the full private key sk_{ID} for the identity from the (correct) secret value x_{ID} and the supplied partial private key d_{ID}; then returns the decryption of C under this private key. If either process fails, then the oracles returns $\perp$.

It is reasonable to assume that the functionality given by this oracle should not be available either to a Type I attacker (i.e. an attacker who is not the KGC) or a Type II attacker (i.e. an honest-but-curious KGC who computes of all its algorithms correctly). However, it cannot be ignored in the malicious KGC setting. Unfortunately, this oracle is not considered in the Au *et al.* paper and therefore their model must necessarily be considered incomplete. Note that the functionality of this oracle *cannot* be simulated by a strong decryption oracle.

The allowable use of this new oracle is also an interesting question. It is clear that one can trivially break the confidentiality of a scheme if one can submit the challenge ciphertext C^* to the Weak PPK Decrypt oracle along with a correct partial private key d_{ID^*} for ID^*. It would be nice if we could allow the attacker to query the Weak PPK Oracle on the ciphertext C^* with partial private key values d which are "incorrect" for the identity ID^*; however, this pre-supposes that the users and the challenger in the security model can tell the difference between correct and incorrect keys (using only knowledge of the master public key mpk). Since we cannot guarantee this to be the case, we choose instead to stipulate that the challenge ciphertext C^* cannot be submitted to the Weak PPK Decrypt oracle at all and note that the potential for a stricter security model exists.

We therefore arrive at a new model for malicious KGCs (which is heavily based on the existing Au *et al.* model). An attacker is defined as a triple of algorithms $(\mathscr{A}_0^{II}, \mathscr{A}_1^{II}, \mathscr{A}_2^{II})$ and the attack game is altered as follows:

1. The attacker executes $\mathscr{A}_0^{II}$ on the input 1^k. $\mathscr{A}_0$ terminates by outputting a master public key value $mpk \in \mathscr{MPK}$ and some state information $state_0$. $\mathscr{A}_0^{II}$ may not query any oracles during this phase of the attack game.
2. The attacker executes $\mathscr{A}_1^{II}$ on the input $state_0$. During its execution $\mathscr{A}_1^{II}$ may query its oracles as usual. $\mathscr{A}_1^{II}$ terminates by outputting an identity ID^*, two equal-length messages (m_0, m_1) and some state information $state_1$.
3. The challenger randomly chooses a bit $b \in \{0,1\}$ and computes the challenge ciphertext as before, i.e. $C^* = \mathtt{Encrypt}(mpk, ID^*, pk_{ID^*}, m_b)$ using the value of pk_{ID^*} currently associated with the identity ID^*.
4. The attacker executes $\mathscr{A}_2^{II}$ on the input $(C^*, state_1)$. During its execution $\mathscr{A}_2^{II}$ may query its oracles as usual. $\mathscr{A}_2^{II}$ terminates by outputting a guess b' for b.

The attacker wins the game if $b = b'$ and its advantage is defined in the usual way.

Definition 8 A certificateless encryption scheme is Malicious Strong Type II secure if every probabilistic polynomial-time attacker $\mathscr{A}^{II} = (\mathscr{A}_0^{II}, \mathscr{A}_1^{II}, \mathscr{A}_2^{II})$ has negligible advantage in winning the above attack game subject to the following oracle constraints:

- $\mathscr{A}^{II}$ may not query the Extract Partial Private Key, Extract Private Key, Weak SV Decrypt or Decrypt oracles at any time,
- $\mathscr{A}_0^{II}$ may not query any oracle,
- $\mathscr{A}_1^{II}$ may not output a challenge identity ID^* for which it has replaced the public key,
- $\mathscr{A}_2^{II}$ may not query the Weak PPK Decrypt oracle with the ciphertext C^* on the identity ID^* (with any partial private key value),
- $\mathscr{A}_2^{II}$ may not query the Strong Decrypt oracle on the challenge ciphertext C^* for the identity ID^* unless the public key pk_{ID^*} used to create that ciphertext has been replaced.

Definition 9 A certificateless encryption scheme is Malicious Weak Type II secure if every probabilistic polynomial-time attacker $\mathscr{A}^{II} = (\mathscr{A}_0^{II}, \mathscr{A}_1^{II}, \mathscr{A}_2^{II})$ has negligible advantage in winning the above attack game subject to the following oracle constraints:

- $\mathscr{A}^{II}$ may not query the Replace Public Key, Extract Partial Private Key, Extract Private Key, Strong Decrypt, Weak SV Decrypt or Decrypt oracles at any time,
- $\mathscr{A}_0^{II}$ may not query any oracle,
- $\mathscr{A}_2^{II}$ may not query the Weak PPK Decrypt oracle with the ciphertext C^* on the identity ID^* (with any partial private key value),

Note that a Malicious Strong Type II† attacker may simulate a Strong Type II† attacker by setting $\mathscr{A}_0$ to run the `Setup` algorithm and storing the complete master key pair (mpk, msk). The attacker's knowledge of the master private key allows the attacker to simulate the Extract Partial Private Key oracle. A Malicious Weak Type II† attacker may simulate a Weak Type II† attacker in the same way. Again, the attacker's knowledge of the master private key allows the attacker to simulate the Extract Partial Private Key and Weak SV Decrypt oracle. However, the Weak PPK Decrypt oracle is necessary to simulate the Decrypt oracle. The different type of oracles given to an attacker is summarised in Table 3. The relationship between the different types of security models for Type II attackers is shown Figure 1.

2.7 Denial of Decryption Attacks

2.7.1 Type I Attackers

One potential problem with certificateless encryption is that a sender may be presented with a choice of public keys which claim to belong to a given individual. Since none these public keys are verified by a trusted authority, the sender may be unable to determine which public key to use when encrypting a message. In preprints of this paper, we have termed this the public key distribution problem. In a paper by Liu, Au and Susilo [25] this is termed a "denial of decryption" attack as a sender that uses a false public key denies the receiver the opportunity to decrypt the ciphertext.

A certificateless encryption scheme is said to resist denial of decryption attacks if it is impossible for a polynomial-time attacker to observe a genuine public key pk and compute a new public key pk' which is valid but acts in a substantially different way to pk. In particular, it should be impossible to find a message m which can be encrypted under pk' to give a ciphertext that a genuine receiver will decrypt to give a value other than m.

Obviously, public keys that are generated using the original formulation of Al-Riyami and Paterson (see Section 2.1.1) can never resist these denial of decryption attacks. In the Al-Riyami and Paterson formulation, an attacker can always generate a secret value (using `Set-Secret-Value`) and a valid public key (using `Set-Public-Key`) and claim that this key belongs to another user. Hence, if we are to have systems that resist denial of decryption attacks, then we are forced to use the Baek, Safavi-Naini and Susilo formulation (see Section 2.1.3). In this formulation, public keys can only be created after receiving partial private keys.

In order to model denial-of-decryption attacks, Liu, Au and Susilo [25] proposed the following attacker game for an attacker $\mathscr{A}$:

1. The challenger generates a master key pair $(mpk, msk) = \texttt{Setup}(1^k)$.
2. The attacker executes $\mathscr{A}$ on mpk. During its execution $\mathscr{A}_1$ may have access to certain oracles (described subsequently). $\mathscr{A}$ terminates by outputting an identity ID^* and a message m^*.

The attacker wins the game if

- $C^* = \texttt{Encrypt}(mpk, ID^*, pk_{ID^*}, m^*) \neq \perp$ where pk_{ID^*} is the public key currently associated with the identity ID^*, and
- $\texttt{Decrypt}(mpk, sk_{ID^*}, C^*) \neq m^*$ where sk_{ID^*} is the (original) private key for the identity ID^*.

We wish to give the attacker access to as many oracles as possible, while still preventing the trivial attacks that occur when the attacker is given access to d_{ID^*}. We give the attacker access to Request Public Key, Replace Public Key, Extract Partial Private Key, Extract Private Key, Strong Decrypt, and Decrypt oracles. (Since we are forced to use the Baek, Safavi-Naini and Susilo formulation, there is no concept of a secret value and so we cannot give the attacker access to a Extract Secret Value or Weak SV Decrypt oracle.)

Definition 10 A certificateless encryption scheme is Strong DoD-Free if every probabilistic polynomial-time attacker $\mathscr{A}$ has negligible advantages in winning the DoD-Free game subject to the following oracle constraints:

Table 3 A Summary of the Oracle Access Provided to a Malicious Type II Attacker

	Request Public Key	Replace Public Key	Extract Private Key	Extract Partial Private Key	Strong Decrypt	Weak PPK Decrypt	Weak SV Secret	Decrypt
Mal. Strong Type II	✓	✓			✓	✓		
Mal. Weak Type II	✓					✓		

Fig. 1 The relationship between security notions for Type II attackers

$$\begin{array}{ccccc} \text{Malicious Strong Type II}^{\dagger} & \Rightarrow & \text{Malicious Weak Type II}^{\dagger} & & \\ \Downarrow & & \Downarrow & & \\ \text{Strong Type II}^{\dagger} & \Rightarrow & \text{Weak Type II}^{\dagger} & \Rightarrow & \text{Weak Type Ic}^{\dagger} \\ \Updownarrow & & \Updownarrow & & \Downarrow \\ \text{Strong Type II} & \Rightarrow & \text{Weak Type II} & \Rightarrow & \text{Weak Type Ic} \end{array}$$

– $\mathscr{A}$ cannot output an identity ID^* of which it has extracted the partial private key.

Definition 11 A certificateless encryption scheme is Weak DoD-Free if every probabilistic polynomial-time attacker $\mathscr{A}$ has negligible advantages in winning the DoD-Free game subject to the following oracle constraints:

– $\mathscr{A}$ cannot output an identity ID^* of which it has extracted the partial private key,
– $\mathscr{A}$ cannot query the Strong Decrypt oracle at any time.

2.7.2 Type II Attackers

The aforementioned definition of "denial of decryption" attacks only consider Type I attackers. A KGC can always perform denial of decryption attacks as the KGC can always compute a correct partial private key d_{ID} and a correct secret value x_{ID}, and form a valid public key. Therefore, the best situation that one might be able to hope for is that the KGC cannot create a new public key for an identity without somehow leaving some evidence as to its illegal action. This is the case for a CA, which, if it certifies a false public key, must produce a certificate signed using the CA's private key. Since this signature cannot be produced by any entity except the CA, any certificate produced on a false public key indicates that the CA must have performed an illegal operation.

Unfortunately, it is not clear how to phrase a reasonable "denial of decryption" requirement for a Type II attacker, even for a scheme using the Baek, Safavi-Naini and Susilo formulation. The problem lies in that it is impossible to differentiate between a public key produced and distributed by a legitimate user and a public key produced and distributed by a cheating KGC. In the case of CA, the CA will keep records of the protocol undertaken with the user to publish a key; if these records cannot be produced then the CA must have fraudulently created a certificate. Since, in a certificateless system, the user does not have to interact with the KGC (beyond initially receiving a partial private key) in order to produce a new public key, it is impossible to tell whether a new public key was produced by the user or not.

We note that if the process of creating a public key was extended to include a protocol interaction between the user and the KGC, then a sensible security notion can be achieved. An extended system of this form would share many similarities with a standard PKI system (see Section 3.6).

3 Surveying Certificateless Encryption Schemes

We shall now attempt to survey the existing literature and constructions for certificateless encryption schemes. We will do this by breaking the literature down into a series of topics. The different constructions of certificateless encryption scheme are summarised in Table 4. This table differentiates between concrete constructions (which give a full description of a specific certificateless encryption scheme) and generic constructions (which describe a method for constructing a certificateless encryption scheme from other primitives).

3.1 Concrete Certificateless Encryption Schemes

The notion of certificateless encryption was introduced by Al-Riyami and Paterson [2,3]. This paper also introduced a concrete scheme (Al-Riyami–Paterson 1) which was proven secure in the random oracle model for the original security models. A second paper by Al-Riyami and Paterson [2, 4] was published two years later. This paper introduced a new concrete certificateless encryption scheme (Al-Riyami–Paterson 2) which claimed to be more efficient than the original scheme. Both of these papers began by introducing a weakly secure scheme and applied techniques similar to the Fujisaki–Okamoto transform [19] to the weaker scheme to achieve full security. However, both papers proved the security of the full scheme directly.

Unfortunately, the second Al-Riyami and Paterson scheme was broken (independently) by Libert and Quisquater [24] and Zhang and Feng [34]. Each of these papers proposed a 'fix' for the scheme. Zhang and Feng proposed a 'tweaked' version of the scheme without attempting to prove its security. Despite the lack of proof, the scheme appears to be secure, and is very similar to the Cheng and Comley [14] scheme (which is proven secure in the random oracle model). Libert and Quisquater suggested that the Al-Riyami and Paterson scheme could be secured by applying the certificate-

Table 4 A Survey of Certificateless Encryption Schemes

Scheme	Reference	Style	Model	Type I Model	Type II Model	Broken?
Al-Riyami 1 / Yum–Lee 1	[2,31]	Generic		Weak Type Ib*	Weak Type II	[20,24]
Al-Riyami 2	[2]	Generic	- - -	No proof given	No proof given	Sec. 3.2
Al-Riyami 3	[2]	Generic	- - -	No proof given	No proof given	[24]
Al-Riyami–Paterson 1	[2,3]	Concrete	ROM	Strong Type I	Weak Type II	
Al-Riyami–Paterson 2	[2,4]	Concrete	ROM	Strong Type I	Weak Type II	[24,33]
Au *et al.*	[5]	Concrete	ROM	Strong Type I*†	Mal. Strong Type II†♯	?
Baek–Safavi-Naini–Susilo	[7]	Concrete	ROM	Strong Type I*	Weak Type II	?
Bentahar *et al.*	[10]	Generic	ROM	Weak Type Ia	Weak Type II	
Cheng–Comley	[14]	Concrete	ROM	Weak Type Ib*†	Weak Type II	
Dent–Libert–Paterson 1	[16]	Generic		Strong Type I	Strong Type II	
Dent–Libert–Paterson 2	[16]	Concrete		Strong Type I	Strong Type II	
Dodis–Katz	Sec 3.5	Generic		Weak Type Ia†	Mal. Weak Type II†	
Lai–Kou 1	[23]	Concrete	ROM	Strong Type I	Weak Type II	?
Lai–Kou 2	[23]	Concrete	ROM	Strong Type I	Weak Type II	?
Libert–Quisquater 1	[24]	Generic	ROM	Strong Type I	Weak Type II	
Libert–Quisquater 2	[24]	Generic	ROM	Strong Type I	Weak Type II	
Libert–Quisquater 3	[24]	Generic	ROM	Strong Type I	Weak Type II	
Libert–Quisquater 4	[24]	Concrete	ROM	Strong Type I	Weak Type II	
Liu–Au–Susilo	[25]	Concrete		Weak Type Ia	Weak Type II	
Huang–Wong 1	[21]	Generic		Weak Type Ia*†	Mal. Weak Type II†♯	
Huang–Wong 2	[22]	Generic		Weak Type Ia*†	Mal. Weak Type II†♯	
Huang–Wong 3	[22]	Generic		Weak Type Ia*†	Mal. Weak Type II†♯	
Shi–Li	[28]	Concrete	ROM	Strong Type I	Weak Type II	
Yum–Lee 2	[32]	Generic		Weak Type Ia*	Weak Type II	[20]
Zhang–Feng	[33]	Concrete	- - -	No proof given	No proof given	?

* = The attacker is not allowed to query the partial private key extraction oracle on the challenge identity
† = The attacker has access to a secret value extraction oracle instead of a private key extraction oracle
♯ = The attacker does not have access to a Weak PPK Decrypt oracle

less Fujisaki–Okamoto transform to the underlying weak certificateless encryption scheme (see Section 3.2).

Libert and Quisquater also proposed an efficient concrete certificateless encryption scheme (Libert–Quisquater 4) that is proven secure in the random oracle model. This scheme is similar to, but more efficient than, the scheme by Shi and Li [28].

However, these schemes are all proven secure in the random oracle model or are only proven secure in weak security models. Hence, there was a question as to whether it was possible to prove the security of a certificateless encryption scheme in a strong security model without the use of the random oracle model. This question was answered by Dent, Libert and Paterson, who proved the security of two schemes [16] in the Strong Type I and Strong Type II models without using random oracles, including one concrete scheme based on the Waters identity-based encryption scheme [30] and the Boyen–Mei–Waters hierarchical identity-based encryption scheme [13].

3.2 Generic Constructions

Soon after the introduction of the notion of certificateless encryption, researchers turned their attention to the question of designing a certificateless encryption scheme using a public-key encryption scheme and an identity-based encryption scheme. The intuition was simple: independent instances of an identity-based encryption scheme and a public-key encryption scheme can be combined to give security. The user's public key would be the public key of the public-key encryption scheme. The user's private key would be the combination of the user's identity-based private key (supplied by the KGC as the partial private key) and the private key for the public-key encryption scheme.

The first attempt to construct certificateless encryption schemes in this manner were given by Al-Riyami [2] and Yum and Lee [31]. The following constructions were obtained:

- Al-Riyami 1/Yum–Lee 1 sequentially compose the public-key encryption scheme and the identity-based encryption scheme. The ciphertext is formed by first encrypting the message using the public-key scheme and then the resulting ciphertext is encrypted using the identity-based encryption scheme.
- Al-Riyami 2 sequentially composes the identity-based encryption scheme and the public-key encryption scheme. The ciphertext is formed by first encrypting the message using the identity-based encryption scheme and then the resulting ciphertext is encrypted using the public-key encryption scheme.
- Al-Riyami 3 composes the identity-based and public-key encryption schemes in parallel. The ciphertext is formed by splitting the ciphertext into two shares s_1 and s_2 such that $s_1 \oplus s_2 = m$, encrypting s_1 under the identity-based

encryption scheme and encrypting s_2 under the public-key encryption scheme.

A second scheme was also proposed by Yum and Lee [32]. The Yum–Lee 2 scheme is similar to the Al-Riyami 1/Yum–Lee 1 scheme except that the public-key encryption scheme is replaced with a second instance of the identity-based encryption scheme.

Unfortunately, none of these naive schemes are secure. Libert and Quisquater [24] note that Al-Riyami 1/Yum–Lee 1 can be easily broken if the attacker can obtain the partial private key (identity-based decryption key) of the challenge identity. (It should be noted that Yum and Lee specifically exclude this possibility in their security proof.) Galindo, Morillo and Ráfols [20] use a similar technique to break the scheme in the Weak Type II model. This attack applies to the second Yum and Lee scheme too.

Libert and Quisquater [24] prove that Al-Riyami 3 can be easily broken using two decryption oracle queries: one that recovers the share s_1 and one that recovers the share s_2.

Libert and Quisquater [24] also note that Al-Riyami 2 is not secure in the Strong Type I model. However, we can demonstrate that the scheme has more serious problems. The scheme fails to achieve Weak Type Ib security. Consider an attacker who replaces the public key of the challenge identity with a public key *pk* for which the attacker knows the corresponding private key *sk*. The attacker may now "strip off" the public key component of the challenge ciphertext using *sk* to leave the ciphertext for the identity-based encryption scheme. The attacker may now re-encrypt the identity-based ciphertext using the original public key value. This is likely to result in a new ciphertext which can be submitted to the Decrypt oracle to retrieve the value of the challenge message.

Bentahar *et al.* [10] approach the problem of constructing a certificateless encryption scheme in a different manner. These authors show that a certificateless encryption scheme may be constructed by composing a certificateless KEM with a standard DEM in a manner similar to Cramer and Shoup [15]. This construction involves separating the encryption scheme into an asymmetric KEM part (which randomly generates a symmetric key K and an "encapsulation" of that key) and a symmetric DEM part (which encrypts a message under the symmetric key K). Bentahar *et al.* were able to propose a notion of security for a certificateless KEM that enabled them to prove that the combination of a secure KEM and a secure DEM is secure in the Weak Type Ia and Weak Type II models. The authors also propose a generic certificateless KEM based on the parallel composition paradigm of Al-Riyami 3 and prove that this scheme is secure in the random oracle model.

Huang and Wong [22] extended the concept of a certificateless KEM to a certificateless Tag-KEM, mirroring the work of Abe *et al.* [1] in the public key setting. The advantage of Tag-KEMs is that they can be combined with passively secure DEMs and still produce schemes which are fully secure against active attackers. Huang and Wong proposed a generic secure KEM and Tag-KEM (Huang–Wong 2 and Huang–Wong 3) that can be proven secure without the use of the random oracle methodology and are only slightly less efficient that the Bentahar *et al.* construction. The construction is similar to Huang–Wong 1 (see Section 3.4).

Libert and Quisquater [24] propose a different method for constructing secure certificateless encryption schemes, via a certificateless version of the Fujisaki–Okamoto transform. This transform works in the random oracle model and produces a strongly secure certificateless encryption scheme from a suitably random certificateless encryption scheme that is secure against attackers who do not make decryption oracle queries. (The "suitably random" condition is a technical necessity known as γ-uniformity.) Libert and Quisquater go on to note that the three naive generic constructions (Al-Riyami 1/Yum–Lee 1, Al-Riyami 2, Al-Riyami 3) are secure against attackers who do not make decryption oracle queries. Thus, applying the certificateless Fujisaki-Okamoto transform, we can obtain three certificateless encryption schemes that are secure in the random oracle model against strong attackers (Libert–Quisquater 1–3).

Since the certificateless Fujisaki–Okamoto transform is used in the construction of several schemes, we briefly review it here. Let `Setup`, `Extract-PPK`, `Set-Secret-Value`, `Set-Public-Key`, `Set-Private-Key`, `Encrypt`, `Decrypt` be a certificateless encryption scheme that is secure against attackers that do not make decryption oracle queries. Let *Hash* be a hash function. We will assume that $\mathtt{Alg}(x;r)$ denotes the execution of the algorithm `Alg` on the input x using the random coins r. The fully secure certificateless encryption scheme, denoted using primes, is shown in Figure 2. Note that only the `Encrypt` and `Decrypt` algorithms are changed.

Lastly, Dent, Libert and Paterson [16] have presented a scheme that is secure against Strong Type I and Strong Type II attackers without requiring the use of the random oracle model. Their construction combines a certificateless encryption which is secure against attackers that make no decryption oracle queries, a public-key encryption scheme and a non-interactive zero-knowledge proof system. This demonstrates that strongly secure certificateless encryption schemes exist if trapdoor one-way permutations and identity-based encryption schemes that are secure against attackers who make no decryption oracle queries exist.

3.3 The Baek, Safavi-Naini and Susilo Formulation

Baek, Safavi-Naini and Susilo presented an alternative formulation for certificateless encryption (see Section 2.1.3) in which a user can only produce their public key after receiving the partial private key. This allowed them to present a certificateless encryption scheme [7] which does not require a pairing. Unfortunately, the authors have recently revealed that a flaw in their proof has been found [6]. Thus, the security of the scheme has to be viewed as unproven.

Certificateless encryption schemes that are presented in the Baek *et al.* formulation have the opportunity to resist denial of decryption attacks (see Section 2.7).

$\texttt{Setup} \equiv \texttt{Setup}'$	$\texttt{Encrypt}'(mpk, pk, ID, m)$	$\texttt{Decrypt}'(mpk, sk, C)$
$\texttt{Extract-PPK} \equiv \texttt{Extract-PPK}'$	1. Randomly chose $\sigma \in \{0,1\}^k$	1. Compute $m\|\|\sigma = \texttt{Decrypt}(mpk, sk, C)$
$\texttt{Set-Secret-Value} \equiv \texttt{Set-Secret-Value}'$	2. Set $r = Hash(m, \sigma, pk, ID)$	2. Set $r' = Hash(m, \sigma, pk, ID)$
$\texttt{Set-Public-Key} \equiv \texttt{Set-Public-Key}'$	3. Set $C = \texttt{Encrypt}(mpk, pk, ID, m\|\|\sigma; r)$	3. Set $C' = \texttt{Encrypt}(mpk, pk, ID, m\|\|\sigma; r')$
$\texttt{Set-Private-Key} \equiv \texttt{Set-Private-Key}'$	4. Output C	4. If $C = C'$, output m. Otherwise, output $\perp$

Fig. 2 The Certificateless Fujisaki-Okamoto Transform

Liu, Au and Susilo [25] produced the original model for security against denial of decryption attacks and proposed a generic construction that combined a certificateless encryption scheme and a certificateless signature scheme to form a certificateless encryption scheme that is secure against denial of decryption attacks. Essentially, the scheme involves a user signing the public key for their certificateless encryption scheme using the certificateless signature scheme. Any attempt to create a new public key for an identity will fail because it is impossible to forge a signature on the public encryption key value

The first attempt to produce a concrete scheme that is secure against denial of decryption attacks is also given by Liu, Au and Susilo [25]. Their paper proposes a normal certificateless encryption scheme based on the Waters identity-based encryption scheme [30] and the Boneh–Katz conversion for CCA security [12]. They apply their generic transform to this scheme to give a DoD-Free certificateless encryption scheme. Huang and Wong [21] claim to break this scheme, but only manage to break it in the malicious KGC model.

Lai and Kou have presented two variants of the Baek *et al.* scheme [23] in which the user engages in a protocol with the KGC when computing their full public and private keys. This is a weakening of the Baek, Safavi-Naini and Susilo formulation, but seems reasonable. The authors claim their schemes are secure against strong attackers and also resist denial of decryption attacks; however, the proofs of these statements do not seem to appear in the published literature and it is unclear if the proofs suffer from the same flaws as the proof for the original Baek *et al.* scheme.

We note that this new relaxed formulation of a certificateless encryption scheme, in which a user is allowed to engage in a protocol with the KGC when computing their public/private key pair, allows for the normal combination of a public-key infrastructure and public-key encryption scheme to be viewed as a certificateless encryption scheme. We discuss this approach in Section 3.6.

3.4 Schemes Secure Against Malicious KGCs

Full security for a certificateless encryption scheme would mean that confidentiality was preserved against actively malicious key generation centres (see Section 2.6).

The Au *et al.* scheme [5] was the first scheme to claim security in the malicious KGC setting. The scheme itself is identical to the Libert–Quisquater 1 scheme [24]. The authors claim that one can construct a certificateless encryption scheme that is secure against Malicious Strong Type $\mathrm{II}^{*\dagger\sharp}$ attackers by applying the certificateless Fujisaki–Okamoto transform of Libert and Quisquater (see Section 3.2) to a suitably random scheme that is secure against Malicious Type $\mathrm{II}^{*\dagger\sharp}$ attackers that do not make any decryption oracle queries. It is important to note that Au *et al.* prove the security of their scheme in a security model that does not allow Weak PPK Decrypt oracle queries.

Unfortunately, we can show this transform is not secure against Malicious Type II attackers in general. In particular, we use the Weak PPK Decrypt oracle to show that there exists a scheme secure against Malicious Type $\mathrm{II}^{\dagger}$ attackers that make no decryption oracle queries but for which the Fujisaki-Okamoto transform fails to give a scheme that is secure in the full Malicious Type $\mathrm{II}^{\dagger}$ model. Since, we believe that the correct model for security against allows the attacker access to an Weak PPK Decrypt oracle, we conclude that the certificateless Fujisaki–Okamoto transform is not sufficient to provide security against malicious KGCs.

Suppose $(\texttt{Setup}_1, \texttt{Extract-PPK}_1, \texttt{Set-Sec}_1, \texttt{Set-Pub}_1, \texttt{Set-Priv}_1, \texttt{Encrypt}_1, \texttt{Decrypt}_1)$ is a certificateless encryption scheme that is secure against Malicious Type $\mathrm{II}^{\dagger}$ attackers that make no decryption oracle queries. We construct a second certificateless encryption scheme that is also secure against Malicious Type $\mathrm{II}^{\dagger}$ attackers that make no decryption oracle queries. The following algorithms remain the same:

$\texttt{Setup}_2 = \texttt{Setup}_1$
$\texttt{Set-Sec}_2 = \texttt{Set-Sec}_1$
$\texttt{Set-Pub}_2 = \texttt{Set-Pub}_1$
$\texttt{Encrypt}_2 = \texttt{Encrypt}_1$

We propose a major change to the `Extract-PPK` algorithm. The algorithm now runs as follows:

$\texttt{Extract-PPK}_2(mpk, msk, ID)$
1. Set $d = \texttt{Extract-PPK}_1(mpk, msk, ID)$
2. Output $d_{ID} = (0, 0, 0, d)$

The `Set-Priv` algorithm is changed to accommodate these changes:

$\texttt{Set-Priv}_2(mpk, d_{ID}, x_{ID})$
1. Parse d_{ID} as (a, b, c, d)
2. Set $sk' = \texttt{Set-Priv}_1(mpk, d, x_{ID})$
3. Output $sk_{ID} = (a, b, c, sk')$

Lastly, we change the `Decrypt` algorithm so that it will leak information to an attacker making Weak PPK Decrypt oracle queries.

$\texttt{Decrypt}_2(mpk, sk_{ID}, C)$
1. Parse sk_{ID} as (a, b, c, sk')
2. If $a = 0$, then output $\texttt{Decrypt}_1(mpk, sk', C)$
3. Otherwise, compute $m = \texttt{Decrypt}_1(mpk, sk', c)$
4. If the b-th bit of m is 0, output $\perp$
5. Otherwise, output $\texttt{Decrypt}_1(mpk, sk', C)$

This scheme is secure in the Malicious Type II† model provided the attacker makes no decryption oracle queries. Therefore, we may apply the certificateless Fujisaki-Okamoto transform to the scheme. The resulting scheme is secure in the Malicious Type II$^{\dagger\sharp}$ model.

However, we can show that the above scheme is not secure in the Malicious Type II† model. Interestingly, the weakness comes not from the attacker's ability to choose the master public key, but from the attacker's ability to access the Weak PPK Decrypt oracle. The attack works as follows:

– $\mathscr{A}_0$ generates a master public/private key pair according to the setup algorithm $(mpk, msk) = \mathtt{Setup}_2(1^k)$, and returns the master public key value mpk.
– $\mathscr{A}_1$ chooses two messages (m_0, m_1) that differ in the first bit. For simplicity we assume that the first bit of message m_b is b. The attacker also picks a valid identity ID^*, and outputs the messages (m_0, m_1) and the identity ID^*. The challenger then chooses a bit b at random and computes the challenge ciphertext C^* as the encryption of m_b under the identity ID^*.
– $\mathscr{A}_2$ picks a random message m distinct from m_0 and m_1, and computes the ciphertext C as the encryption of m under the identity ID^*. The attacker also computes $d = \mathtt{Extract\text{-}PPK}_1(mpk, msk, ID^*)$ and forms the false partial private key value $d_{ID^*} = (1, 1, C^*, d)$. The attacker submits the ciphertext C and the partial private key d_{ID^*} to the Weak PPK Decrypt oracle. If the attacker receives back the response $\perp$ then it outputs 0; otherwise it outputs 1.

Let us examine this attack in detail. The attacker submits the partial private key $d_{ID^*} = (1, 1, C^*, d)$. Hence, if the challenger chose message m_0 as the challenge message, then the decryption $\mathtt{Decrypt}_2$ algorithm will output $\perp$ and so the Weak PPK Decrypt oracle will return $\perp$. If the challenger chose message m_1 as the challenge message, then the $\mathtt{Decrypt}_2$ algorithm will return the decryption of the ciphertext C. Since C is a valid and perfectly formed encryption, this will pass the Fujisaki-Okamoto "re-encryption check" and the Weak PPK Decrypt oracle will return m.

We are therefore forced to conclude that the certificateless Fujisaki-Okamoto transform does not give security in the Malicious Type II† model. Since the Au *et al.* scheme relies on this transform we are forced to conclude that its security is unproven, despite their being no known attacks against the scheme.

A different approach to constructing a scheme secure against malicious key generation centres was given by Huang and Wong [21]. The Huang–Wong 1 scheme is a generic scheme that sequentially composes an IBE scheme and a public-key encryption scheme in manner similar to the Al-Riyami 2 scheme (see Section 3.2). The schemes differ in that the Huang–Wong scheme forces the sender to sign the ciphertext with a one-time signature scheme (whose verification key is included in the plaintext). The scheme is proven secure in the Weak Type Ia*† and Malicious Weak Type II$^{\dagger\sharp}$ models. We believe that the use of the Malicious Weak Type II$^{\dagger\sharp}$ model is a mistake as this does not imply security in the Weak Type II† model. However, it seems likely that this scheme is actually secure in the Malicious Type II† oracle, which does imply security in the Weak Type II† model. This approach is similar to the multiple encryption schemes proposed by Dodis and Katz [18].

Huang and Wong [22] have also proposed generic KEM and Tag-KEM schemes that are proven secure in the Weak Type Ia*† and Malicious Weak Type II$^{\dagger\sharp}$ models. These constructions (Huang–Wong 2 and Huang–Wong 3) are similar to the Huang–Wong 1 scheme in that they compose an identity-based component with a public-key encryption component, but the signature scheme is replaced with a secure MAC. This leads to a significant performance advantage and the resulting schemes are only slightly less efficient than the Bentahar *et al.* KEM (which is only proven secure in the random oracle model).

Lastly, it has often been suggested that a secure certificateless encryption scheme can be constructed by combining an identity-based encryption scheme and a public-key encryption scheme in parallel using the techniques of Dodis and Katz [18]. We show that this does in fact give a secure in certificateless encryption scheme in Section 3.5.

3.5 A Construction Based on Dodis-Katz Encryption

In this section we will discuss the possibility of constructing a certificateless encryption scheme using the multiple encryption techniques of Dodis and Katz [18]. The idea that a secure certificateless encryption scheme can be built in this way has been widely discussed in the cryptographic community. The resulting encryption scheme shares many similarities with Huang-Wong 1 scheme, except that where the Huang-Wong 1 scheme extends the sequential Al-Riyami 2 scheme, the Dodis-Katz scheme extends the parallel Al-Riyami 3 scheme.

The scheme uses an adaptively secure public-key encryption scheme:

$$PK = (PK.Gen, PK.Encrypt, PK.Decrypt)\,,$$

an identity-based encryption scheme:

$$ID = (ID.Gen, ID.Extract, ID.Encrypt, ID.Decrypt)\,,$$

and a one-time signature scheme:

$$Sig = (Sig.Gen, Sig.Sign, Sig.Verify)\,.$$

We assume that the public key encryption scheme and the identity-based encryption scheme support the use of labels[3]. The certificateless encryption scheme is defined as follows:

– $\mathtt{Setup}$: This algorithm computes $(mpk, msk) = ID.Gen(1^k)$ and returns the master public key mpk and the master private key msk.

[3] An encryption scheme supports the use of labels if it is possible to non-malleably bind information to the ciphertext. A ciphertext can only be correctly decrypted if the label used to encrypt the ciphertext is also provided on decryption. Obtaining the decryption of a ciphertext using one label should give no information about the decryption of a ciphertext under another label.

– `Extract-Partial-Private-Key`: This algorithm computes the partial private key $d_{ID} = ID.Extract(ID, msk)$.
– `Set-Secret-Value`: The secret value is defined to be $x_{ID} = (pk_{ID}, sk_{ID}) = PK.Gen(1^k)$.
– `Set-Public-Key`: The public key is defined to be pk_{ID}.
– `Set-Private-Key`: The private key is defined to be the pair (d_{ID}, sk_{ID}).
– `Encrypt`: To encrypt a message m, the sender randomly chooses a string r of length $|m|$ and forms two shares $s_1 = r$ and $s_2 = m \oplus r$. The sender also generates a one-time signature key pair $(vk, sk) = Sig.Gen(1^k)$. The sender computes the two ciphertext shares

$$C_1 = ID.Encrypt^{vk}(s_1, mpk, ID)$$

and

$$C_2 = PK.Encrypt^{vk}(s_2, pk_{ID})$$

using the label vk. Lastly, the sender computes the signature

$$\sigma = Sig.Sign((C_1, C_2), sk)$$

and outputs the ciphertext (C_1, C_2, vk, σ).
– `Decrypt`: To decrypt a ciphertext (C_1, C_2, vk, σ), the receiver firsts checks that the signature is valid, i.e. that $Sig.Verify(\sigma, (C_1, C_2), vk) = \top$. If so, the receiver recovers the shares

$$s_1 = ID.Decrypt^{vk}(C_1, d_{ID})$$

and

$$s_2 = PK.Decrypt^{vk}(C_2, sk_{ID}) ,$$

and outputs $m = s_1 \oplus s_2$.

We claim this scheme gives Weak Type Ia† and Malicious Type II† security. The proofs are given in Appendix A.

3.6 The Relationship With Public-Key Infrastructures

The original formulation of certificateless encryption by Al-Riyami and Paterson (see Section 2.1.1) had complete independence between the partial private key and the public key. The formulation by Baek, Safavi-Naini and Susilo (see Section 2.1.3) insisted that public keys could only be produced by a user after that user has received a valid partial private key. It is therefore tempting to consider an even more relaxed model, where the partial private key for an identity can only be produced after the key generation centre has received some information from the user. In other words, the user must generate a partial public key and send that to the key generation centre, before receiving their partial private key and producing their full private and public keys.

We note here that if such a formulation were allowed, then the normal use of a public-key encryption scheme and a public key infrastructure would form a certificateless encryption scheme. The scheme would involve the receiver generating a normal public key and having that key certified by the CA. The full public key would consist of the original public key and the certificate for that key, and any sender would have to check the signature produced by the CA before encrypting a message using the public key. Boldyreva, Fischlin, Palacio and Warinschi [11] have (inadvertently) proven that this scheme would be secure in the Weak Type Ia† and Malicious Weak Type II† models (assuming the logical extensions to the existing models to handle the new formulation). The scheme would also protect against denial of decryption attacks (see Section 2.7). This results in a very secure certificateless encryption scheme!

There are two disadvantages to this scheme. First, the formulation means that a receiver cannot publish a public key until after it has received its partial private key. This means that messages cannot be ‘encrypted into the future’; however, the Baek, Safavi-Naini and Susilo formulation suffers from this problem too and we do not consider this to be a major problem for most applications (see Section 2.1.3). Second, the scheme is computationally expensive for the sender, who has to verify the signature on the certificate before encrypting the message. This gives a baseline for certificateless encryption schemes. Any certificateless encryption scheme that is to be deemed practical must either be more efficient than the standard combination of a public key encryption scheme and a public key infrastructure or must be explicitly aimed at applications which require “encryption into the future”.

4 Conclusion

4.1 Strong vs. Weak Models

Perhaps the biggest debate in the field of certificateless cryptography is on whether the security models should use strong or weak decryption oracles. The strong models provide a “margin of error,” but do not appear to directly model the abilities of an attacker. We believe that weak decryption oracles are sufficient for all applications of certificateless cryptography and all practical certificateless cryptosystems should be judged against a weak security model. We believe that a scheme proven secure in a strong model should only be considered marginally better than a scheme proven secure in a weak model.

The strong security models can be considered of academic interest as proofs of security in this model seem to require proof techniques which are more complex than for many other models. The question of whether it is possible to construct a certificateless encryption scheme that is secure in the Strong Type I and Strong Type II models without the use of random oracles has now been answered by Dent, Libert and Quisquater [16]. Dent, Libert and Quisquater proposed two schemes that satisfy these conditions; however, neither scheme can be regarded as truly practical. One scheme requires the use of an arbitrary NIZK proof (which is highly inefficient) and the other scheme requires very large public keys. However, due to the theoretetical nature of this problem, we do not consider the construction of more efficient certificateless encryption schemes in this model to be a priority.

One open problem that may still be of interest is the construction of a certificateless encryption scheme that can be proven secure in the Strong Type I and Malicious Strong Type II models without the use of the random oracle methodology. The nature of the Malicious Strong Type II model seems to mean that most standard proof techniques that can be used to prove the Strong Type I security of the scheme may also be used to construct an attacker in the Malicious Strong Type II model. It would therefore be interesting to produce a scheme that can be proven to meet both of these security requirements without the use of random oracles.

4.2 The Al-Riyami and Paterson formulation

The Al-Riyami and Paterson formulation of a certificateless encryption scheme has the advantage of being able to "encrypt messages into the future". In other words, a sender is able to send a message to receiver despite the fact that the receiver may not yet have been given their partial private key. The KGC may only release the partial private key at some point in the future, after the receiver has demonstrated that certain conditions have been met. This is potentially a useful feature in certain applications, including cryptographic workflows and access control. However, many applications do not need this feature, and it therefore seems unnecessary to insist that all certificateless encryption schemes be presented in this formulation. We therefore suggest that the Al-Riyami and Paterson formulation should only be used by applications in which this "encrypt into the future" feature is required; in other circumstances, the Baek, Safavi-Naini and Susilo formulation should be preferred.

As we have already discussed, we believe that the use of strong models should be deprecated. Therefore, we believe that the "correct" notions of security for a certificateless encryption scheme are the Weak Type Ia† and the Malicious Weak Type II† models. In particular, we agree with the view of Cheng and Comley that an attacker should be given access to a secret value oracle, rather than an extract private key oracle, and we agree with the view of Au *et al.* that a secure scheme should resist attacks made by a malicious KGC. If security against a malicious KGC cannot be efficiently achieved, or if the application does not require security against a malicious KGC, then the Weak Type II† model should be adopted as the correct model for security against a passive KGC. We tentatively suggest that these models be re-named *outsider security*, *malicious KGC security*, and *passive KGC security* respectively. This nomenclature corresponds to the nomenclature for security models of signcryption schemes.

Since we are advocating the use of extract secret value oracles instead of extract private key oracles, there is no reason not to adopt the five-algorithm version of the Al-Riyami and Paterson formulation. This formulation has no concept of a private key. Hence, we believe that a certificateless encryption scheme (in the Al-Riyami and Paterson formulation) should be described as a quintuple of algorithms:

- `Setup`: This algorithm takes as input a security parameter 1^k and returns the master public key *mpk* and the master private key *msk*.
- `Extract-Partial-Private-Key`: This algorithm takes as input the master public and private keys (mpk, msk) and an identity $ID \in \{0,1\}^*$. It outputs a partial private key d_{ID}.
- `Set-User-Keys`: This algorithm takes as input the master public key *mpk* and an identity *mpk*, and outputs a secret value x_{ID} and a public key pk_{ID}.
- `Encrypt`: This algorithm takes as input the master public key *mpk*, a user's identity *ID*, a user's public key pk_{ID}, and a message m. It outputs either a ciphertext C or the error symbol $\perp$.
- `Decrypt`: This algorithm takes as input the master public key *mpk*, a user's partial private key d_{ID}, a user's secret value x_{ID}, and a ciphertext C. It outputs either a message m or the error symbol $\perp$.

If we examine Table 4, then it is easy to see which schemes meet the "correct" level of security. Only the Huang–Wong schemes [21,22] and the Dodis–Katz scheme (see Section 3.5) meet the notions of security against malicious key generation centres and are proevn secure in the standard model. Both schemes are highly inefficient (requiring separate public-key encryption, identity-based encryption, and integrity protection operations). Thus, it is clear that the construction of a secure and efficient certificateless encryption scheme should still be considered an open problem.

In situations where security against malicious key generation centres is not required, then there are several schemes proven secure in the random oracle model that seem to meet the security requirements. The most efficient of these is probably the Libert–Quisquater 4 scheme [24]. However, there are no schemes that can be considered simultaneously efficient and secure without the use of random oracles. Hence, the construction of such a scheme should still be considered an open problem.

4.3 The Baek, Safavi-Naini and Susilo formulation

The Baek, Safavi-Naini and Susilo formulation of a certificateless encryption scheme, which only allows a user to generate a public key after receiving their partial private key from the key generation centre, seems to be a very practical and general formulation for a certificateless encryption scheme. The Al-Riyami and Paterson formulation can be viewed as a special case of this formulation.

In the Baek, Safavi-Naini and Susilo formulation, since a user cannot generate a public or private key value until after they have received the partial private key, the user is unlikely to generate a secret value until after receiving the partial private key and is unlikely to keep the secret value after generating the public and private keys. The secret value can therefore be considered as an internal variable which is part of the public/private key generation process and unavailable to an attacker. We therefore consider it *inappropriate* to give the attacker access to a secret value oracle. This is a major difference from security models we proposed for the Al-Riyami and Paterson formulation!

We consider the "correct" notions of security for a certificateless encryption scheme in the Baek, Safavi-Naini and Susilo formulation to be security in the Weak Type Ib and Malicious Weak Type II models. For applications where security against malicious key generation centres is not required, we suggest that the Weak Type II security model be used. Again, we tentatively suggest that these models be re-named *outsider security*, *passive KGC security* and *malicious KGC security*.

These choices allow us to formulate a certificateless encryption scheme as a quintuple of algorithms:

- `Setup`: This algorithm takes as input a security parameter 1^k and returns the master public key *mpk* and the master private key *msk*.
- `Extract-Partial-Private-Key`: This algorithm takes as input the master public and private keys (mpk, msk) and an identity $ID \in \{0,1\}^*$. It outputs a partial private key d_{ID}.
- `Set-User-Keys`: This algorithm takes as input the master public key *mpk* and a partial private key d_{ID}, and outputs a public key pk_{ID} and a private key sk_{ID}.
- `Encrypt`: This algorithm takes as input the master public key *mpk*, a user's identity *ID*, a user's public key pk_{ID}, and a message *m*. It outputs either a ciphertext *C* or the error symbol $\perp$.
- `Decrypt`: This algorithm takes as input the master public key *mpk*, a user's private key sk_{ID}, and a ciphertext *C*. It outputs either a message *m* or the error symbol $\perp$.

We note that a scheme that is proven secure in the security models for the Al-Riyami and Paterson formulation is necessarily secure in the security models for the Baek, Safavi-Naini and Susilo formulation.

However, we believe a secure certificateless encryption scheme that is secure in the DoD-Free security model (see Section 2.7) should be given considerably more credit than a similar scheme which is not secure in the DoD-Free security model. The DoD-Free security model is concerned with preventing an attacker from tricking a sender into using an incorrect public key value for a particular receiver. In other words, it prevents a situation where a sender believes that a message has been successfully encrypted, but the receiver is unable to recover the message. Schemes presented in the Al-Riyami and Paterson formulation are unable to meet the DoD-Free notion of security.

It is also worth noting that a standard public-key encryption scheme, when combined with a public-key infrastructure, can be viewed as a certificateless encryption scheme in a formulation which is only slightly more general than the Baek, Safavi-Naini and Susilo formulation. This more general formulation allows a user to engage in a protocol with the key generation centre when creating their public and private keys. While the security model for such a formulation has never been formally stated, and we note that there may be some subtleties with the security model as the key generation centre may be able to derive some information about a private key from its protocol interaction with a user, we note that Boldyreva *et al.* have proven the security of a public-key encryption scheme in a similar security model. Hence, we believe that any certificateless encryption scheme in Baek, Safavi-Naini and Susilo formulation should be more efficient than the standard public-key encryption with public-key infrastructure scheme if it is to be considered useful.

Since there are no certificateless encryption schemes that are secure in outsider and malicious KGC security models, secure in the DoD-Free model and significantly more efficient than public-key encryption, we conclude that creating an efficient and secure certificateless encryption scheme for general purposes is still an open problem.

4.4 Practical considerations

Lastly, we note that certificateless cryptography has not provided a reasonable general-purpose solution to the problem of public key revocation. As with identity-based cryptography, certificateless cryptograph does not provide a mechanism whereby a sender can be informed that an old public key has either expired or has been declared invalid. This is a major problem for the practical deployment of certificateless cryptography.

One solution that has been suggested for use with identity-based cryptography is to append a validity period to the end of an entity's identity *ID*. The entity is only given the private key for his identity at the beginning of validity period; hence, if the key is revoked, then the key will only be valid for a short period of time and the damage that the entity can cause to the system is limited. This solution can also be applied to certificateless cryptosystem, but it is not suitable for all applications and this issue must be taken into consideration when deciding upon the use of certificateless cryptography.

Acknowledgements The author is indebted to the provable security working group in ECRYPT's AZTEC lab for their help in discussing the topics of this paper. In particular, Caroline Kudla deserves credit for many interesting discussions at the start of this project. The author would also like to thank John Malone-Lee, Nigel Smart, Kenny Paterson and the anonymous reviewers for their comments on several versions of this paper, and to David Galindo for pointing out his work on the Yum–Lee constructions. The author gratefully acknowledge the financial support of the EPSRC.

References

1. Abe, M., Gennaro, R., Karosawa, K., Shoup, V.: Tag-KEM/DEM: A new framework for hybrid encryption. In: R. Cramer (ed.) Advance in Cryptology – Eurocrypt 2005, *Lecture Notes in Computer Science*, vol. 3494, pp. 128–146. Springer-Verlag (2005)
2. Al-Riyami, S.: Cryptographic schemes based on elliptic curve pairings. Ph.D. thesis, Royal Holloway, University of London (2004). Available from `http://www.isg.rhul.ac.uk/~kp/sattthesis.pdf`

3. Al-Riyami, S.S., Paterson, K.G.: Certificateless public key cryptography. In: C.S. Laih (ed.) Advances in Cryptology – Asiacrypt 2003, *Lecture Notes in Computer Science*, vol. 2894, pp. 452–473. Springer-Verlag (2003)
4. Al-Riyami, S.S., Paterson, K.G.: CBE from CL-PKE: A generic construction and efficient schemes. In: S. Vaudenay (ed.) Public Key Cryptography – PKC 2005, *Lecture Notes in Computer Science*, vol. 3386, pp. 398–415. Springer-Verlag (2005)
5. Au, M.H., Chen, J., Liu, J.K., Mu, Y., Wong, D.S., Yang, G.: Malicious KGC attack in certificateless cryptography. In: Proc. ACM Symposium on Information, Computer and Communications Security. ACM Press (2007)
6. Baek, J.: Important note on "Certificateless public key encryption without pairing" (2007). Available from `http://www1.i2r.a-star.edu.sg/~jsbaek/`
7. Baek, J., Safavi-Naini, R., Susilo, W.: Certificateless public key encryption without pairing. In: J. Zhou, J. Lopez (eds.) Proceedings of the 8th International Conference on Information Security (ISC 2005), *Lecture Notes in Computer Science*, vol. 3650, pp. 134–148. Springer-Verlag (2005)
8. Baek, J., Wang, G.: Repairing a security-mediated certificateless encryption scheme from PKC 2006 (2006). Available from `http://eprint.iacr.org/2006/159`
9. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: S. Vaudenay (ed.) Advances in Cryptology – Eurocrypt 2006, *Lecture Notes in Computer Science*, vol. 4004, pp. 409–426. Springer-Verlag (2006)
10. Bentahar, K., Farshim, P., Malone-Lee, J., Smart, N.P.: Generic constructions of identity-based and certificateless KEMs (2005). Available from `http://eprint.iacr.org/2005/058`
11. Boldyreva, A., Fischlin, M., Palacio, A., Warinschi, B.: A closer look at PKI: Security and efficiency. In: T. Okamoto, X. Wang (eds.) Public Key Cryptography – PKC 2007, *Lecture Notes in Computer Science*, vol. 4450, pp. 458–475. Springer-Verlag (2007)
12. Boneh, D., Katz, J.: Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In: A. Menezes (ed.) Topics in Cryptology - CT-RSA 2005, *Lecture Notes in Computer Science*, vol. 3376, pp. 87–103. Springer-Verlag (2005)
13. Boyen, X., Mei, Q., Waters, B.: Direct chosen ciphertext security from identity-based techniques. In: Proc. of the 12th ACM Conference on Computer and Communications Security, pp. 320–329 (2005)
14. Cheng, Z., Comley, R.: Efficient certificateless public key encryption (2005). Available from `http://eprint.iacr.org/2005/012/`
15. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM Journal on Computing **33**(1), 167–226 (2004)
16. Dent, A.W., Libert, B., Paterson, K.G.: Certificateless encryption schemes strongly secure in the standard model (2007). Unpublished Manuscript
17. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Transactions on Information Theory **22**, 644–654 (1976)
18. Dodis, Y., Katz, J.: Chosen-ciphertext security of multiple encryption. In: J. Kilian (ed.) Theory of Cryptography – TCC 2005, *Lecture Notes in Computer Science*, vol. 3378, pp. 188–209. Springer-Verlag (2005)
19. Fujisaki, E., Okamoto, T.: How to enhance the security of public-key encryption at minimal cost. In: H. Imai, Y. Zheng (eds.) Public Key Cryptography, *Lecture Notes in Computer Science*, vol. 1560, pp. 53–68. Springer-Verlag (1999)
20. Galindo, D., Morillo, P., Ràfols, C.: Breaking Yum and Lee generic constructions of certificate-less and certificate-based encryption schemes. In: A.S. Atzeni, A. Lioy (eds.) Public Key Infrastructure: Third European PKI Workshop (EuroPKI 2006), *Lecture Notes in Computer Science*, vol. 4043, pp. 81–91. Springer-Verlag (2006)
21. Huang, Q., Wong, D.S.: Generic certificateless encryption in the standard model. In: A. Miyaji, H. Kikuchi, K. Rannenberg (eds.) Advances in Information and Computer Security (IWSEC 2007), *Lecture Notes in Computer Science*, vol. 4752, pp. 278–291. Springer-Verlag (2007)
22. Huang, Q., Wong, D.S.: Generic certificateless key encapsulation mechanism. In: J. Pieprzyk, H. Ghodosi, E. Dawson (eds.) Information Security and Privacy (ACISP 2007), *Lecture Notes in Computer Science*, vol. 4586, pp. 215–299. Springer-Verlag (2007)
23. Lai, J., Kou, K.: Self-generated-certificate public key encryption without pairing. In: T. Okamoto, X. Wang (eds.) Public Key Cryptography – PKC 2007, *Lecture Notes in Computer Science*, vol. 4450, pp. 476–489. Springer-Verlag (2007)
24. Libert, B., Quisquater, J.J.: On constructing certificateless cryptosystems from identity based encryption. In: M. Yung, Y. Dodis, A. Kiayias, T. Malkin (eds.) Public Key Cryptography – PKC 2006, *Lecture Notes in Computer Science*, vol. 3958, pp. 474–490. Springer-Verlag (2006)
25. Liu, J.K., Au, M.H., Susilo, W.: 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)
26. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM **21**, 120–126 (1978)
27. Shamir, A.: Identity-based cryptosystems and signature schemes. In: G.R. Blakley, D. Chaum (eds.) Advances in Cryptology – Crypto '84, *Lecture Notes in Computer Science*, vol. 196, pp. 47–53. Springer-Verlag (1984)
28. Shi, Y., Li, J.: Provable efficient certificateless public key encryption (2005). Available from `http://eprint.iacr.org/2005/287/`
29. Shoup, V.: Sequences of games: A tool for taming complexity in security proofs (2004). Available from `http://eprint.iacr.org/2004/332/`
30. Waters, B.: Efficient identity-based encryption without random oracles. In: R. Cramer (ed.) Advances in Cryptology - EUROCRYPT 2005, *Lecture Notes in Computer Science*, vol. 3494, pp. 114–127. Springer-Verlag (2005)
31. Yum, D.H., Lee, P.J.: Generic construction of certificateless encryption. In: A.L. *et al.* (ed.) Computational Science and Its Applications ICCSA 2004: Part I, *Lecture Notes in Computer Science*, vol. 3043, pp. 802–811. Springer-Verlag (2004)
32. Yum, D.H., Lee, P.J.: Identity-based cryptography in public key management. In: S.K. Katsikas, S. Gritzalis, J. Lopez (eds.) Public Key Infrastructure: First European PKI Workshop (EuroPKI 2004), *Lecture Notes in Computer Science*, vol. 3093, pp. 71–84. Springer-Verlag (2004)
33. Zhang, Z., Feng, D.: On the security of a certificateless public-key encryption (2005). Available from `http://eprint.iacr.org/2005/426`
34. Zhang, Z., Wong, D.S., Xu, J., Feng, D.: Certificateless public-key signature: Security model and efficient construction. In: J. Zhou, M. Yung, F. Bao (eds.) Applied Cryptography and Network Security, *Lecture Notes in Computer Science*, vol. 3989, pp. 293–308. Springer-Verlag (2006)

A Security Proofs for the Dodis-Katz Construction

In this section we will support our claims that the Dodis-Katz generic construction of a certificateless encryption scheme (see Section 3.5) is secure in the Weak Type Ia† and Malicious Weak Type II† models.

A.1 Security Definitions

We require the use of a secure public-key encryption scheme, identity-based encryption scheme and one-time signature scheme. In this section we will define the basic security notions for these primitives.

A public-key encryption scheme with labels is a triple of probabilistic, polynomial-time algorithms $(PK.Gen, PK.Encrypt, PK.Decrypt)$ where a message m is encrypted using a label ℓ and a public key pk as follows:

$$C = PK.Encrypt^{\ell}(m, pk) \ .$$

Decryption of a ciphertext C using a label ℓ and a private key sk is denoted as:

$$m = PK.Decrypt^{\ell}(C, sk)$$

We require that

$$m = PK.Decrypt^{\ell}(C, sk) \quad \text{whenever} \quad C = PK.Encrypt^{\ell}(m, pk)$$

for all key pairs $(pk, sk) = PK.Gen(1^k)$, messages m, and labels ℓ. The encryption scheme is IND-CCA2 secure if every polynomial-time attacker $\mathscr{A} = (\mathscr{A}_1, \mathscr{A}_2)$ has negligible advantage in winning the following game:

1. The challenger generates the key pair $(pk, sk) = PK.Gen(1^k)$.
2. The attacker executes $\mathscr{A}_1$ on the input pk. During its execution $\mathscr{A}_1$ may adaptively query a decryption oracle with any ciphertext $C \in \mathscr{C}$ and label $\ell \in \{0,1\}^*$. The decryption oracle responds with $PK.Decrypt^{\ell}(C, sk)$. $\mathscr{A}_1$ terminates by outputting two equal length messages (m_0, m_1), a label ℓ^* and some state information *state*.
3. The challenger randomly chooses a bit b and computes the challenge ciphertext $C^* = PK.Encrypt^{\ell^*}(m_b, pk)$.
4. The attacker executes $\mathscr{A}_2$ on the input $(C^*, state)$. During its execution $\mathscr{A}_2$ may query a decryption oracle with any ciphertext/label pair $(C, \ell) \neq (C^*, \ell^*)$. The decryption oracle responds as before. $\mathscr{A}_2$ terminates by outputting a guess b' for b.

The attacker wins the game if $b = b'$. The attacker's advantage is defined to be $|Pr[b = b'] - 1/2|$.

In a similar manner, we define an identity-based encryption scheme with label as a quadruple of probabilistic, polynomial-time algorithms $(ID.Gen, ID.Extract, ID.Encrypt, ID.Decrypt)$. Encryption of a message m with a label ℓ under an identity ID is denoted

$$C = ID.Encrypt^{\ell}(m, mpk, ID) \ .$$

Decryption of a ciphertext C with a label ℓ is denoted

$$m = ID.Decrypt^{\ell}(C, d_{ID})$$

where $d_{ID} = ID.Extract(ID, msk)$ is the private key for the identity ID. As usual we require that

$$m = IDK.Decrypt^{\ell}(C, d_{ID}) \quad \text{whenever} \quad C = ID.Encrypt^{\ell}(m, mpk, ID)$$

for all master key pairs $(mpk, msk) = ID.Gen(1^k)$, identities ID, private keys $d_{ID} = ID.Extract(ID, msk)$, messages m, and labels ℓ. An identity-based encryption scheme is IND-ID-CCA2 secure if every polynomial-time attacker $\mathscr{A} = (\mathscr{A}_1, \mathscr{A}_2)$ has negligible advantage in winning the following game:

1. The challenger generates a master key pair $(mpk, msk) = ID.Gen(1^k)$.
2. The attacker executes $\mathscr{A}_1$ on the input mpk. During its execution $\mathscr{A}_1$ may adaptively query a decryption oracle with any ciphertext $C \in \mathscr{C}$, label $\ell \in \{0,1\}^*$ and identity $ID \in \{0,1\}^*$. The decryption oracle computes a private decryption key $d_{ID} = ID.Extract(ID, msk)$ and responds with $ID.Decrypt^{\ell}(C, d_{ID})$. The attacker may also query an extraction oracle with any identity ID. The extraction oracle responds with $ID.Extract(ID, msk)$. $\mathscr{A}_1$ terminates by outputting two equal length messages (m_0, m_1), a label ℓ^*, an identity ID^*, and some state information *state*.
3. The challenger randomly chooses a bit b and computes the challenge ciphertext $C^* = ID.Encrypt^{\ell^*}(m_b, ID^*)$.
4. The attacker executes $\mathscr{A}_2$ on the input $(C^*, state)$. During its execution $\mathscr{A}_2$ may query a decryption oracle with any triple $(C, \ell, ID) \neq (C^*, \ell^*, ID^*)$. The decryption oracle responds as before. The attacker may also query the extraction oracle as before. $\mathscr{A}_2$ terminates by outputting a guess b' for b.

The attacker wins the game if $b = b'$ and the attacker never queried the extraction oracle on ID^*. The attacker's advantage is defined in the usual way.

Lastly, a one-time signature scheme is a triple of probabilistic polynomial-time algorithms $(Sig.Gen, Sig.Sign, Sig.Verify)$. For a key pair $(vk, sk) = Sig.Gen(1^k)$, a signature on a message m is denoted

$$\sigma = Sig.Sign(m, sk)$$

and a verification of a signature σ on a message m is denoted

$$Sig.Verify(\sigma, m, vk) \in \{\top, \bot\}$$

where $\top$ denotes that the signature is accepted and $\bot$ denotes that the signature is rejected. The scheme is defined to be unforgeable if every polynomial-time attacker $\mathscr{A} = (\mathscr{A}_1, \mathscr{A}_2)$ has negligible probability of winning the following game:

1. The challenger generates the key pair $(vk, sk) = Sig.Gen(1^k)$.
2. The attacker executes $\mathscr{A}_1$ on the input vk. $\mathscr{A}_1$ terminates by outputting a message m and some state information *state*.
3. The challenger computes $\sigma = Sig.Sign(m, sk)$.
4. The attacker executes $\mathscr{A}_2$ on the input $(\sigma, state)$. $\mathscr{A}_2$ terminates by outputting a message m^* and a signature σ^*.

The attacker wins if $Sig.Verify(\sigma^*, m^*, vk) = \top$ and $(m^*, \sigma^*) \neq (m, \sigma)$.

A.2 Weak Type Ia† Security

We show that the Type I security of the construction depends upon the security of the public-key encryption scheme if the attacker learns the partial private key of the challenge identity and upon the security of the identity-based encryption scheme if the attacker replaces the public key of the challenge identity.

Theorem 1 *If there exists a polynomial-time attacker $\mathscr{A}$ that has advantage $Adv^{CL}_{\mathscr{A}}(k)$ in breaking the Dodis-Katz encryption scheme and that makes q_{req} queries to the Request Public Key oracle, then there exists*

- *an attacker $\mathscr{B}$ that has advantage $Adv^{PK}_{\mathscr{B}}(k)$ in breaking the underlying public-key encryption scheme,*
- *an attacker $\mathscr{B}$ that has advantage $Adv^{ID}_{\mathscr{B}}(k)$ in breaking the underlying identity-based encryption scheme, and*
- *an attacker $\mathscr{B}$ that has probability $Adv^{Sig}_{\mathscr{B}}(k)$ in breaking the underlying one-time signature scheme*

such that

$$Adv^{CL}_{\mathscr{A}}(k) \leq \frac{q_{req}}{2} Adv^{PK}_{\mathscr{B}}(k) + \frac{1}{2} Adv^{ID}_{\mathscr{B}}(k) + Adv^{Sig}_{\mathscr{B}}(k) \ . \tag{3}$$

Proof The crux of the proof involves guessing whether $\mathscr{A}$ will replace the public key of the challenge identity or request the partial private key of the challenge identity. We do this by randomly choosing a bit $c \in \{0,1\}$. If $c = 0$ then we assume that $\mathscr{A}$ will request the partial private key of challenge identity. If $c = 1$ then we assume that $\mathscr{A}$ does not request the partial private key of the challenge identity (and therefore has the option of replacing the public key of the challenge identity). We use the unforgeability of the signature scheme to enable us to construct decryption oracles.

Assume there exists an attacker $\mathscr{A}$ that has advantage $Adv^{\mathrm{CL}}_{\mathscr{A}}(k)$ in breaking the CL-PKE scheme. Assume that $\mathscr{A}$ makes at most q_{req} queries to the request public key oracle. For simplicity we shall assume that $\mathscr{A}$ never makes a query to any other oracle on an identity before it has made a query to the request public key oracle on that identity, and that $\mathscr{A}$ will not output a challenge identity on which it has not queried the request public key. We prove the theorem using game hopping techniques [9,29]. Let S_i be the event that $\mathscr{A}$ wins Game i.

Game 1 In Game 1 the challenger interacts with the attacker exactly as described in the Weak Type Ia† model (see Section 2.3.2). In other words, the attacker $\mathscr{A}$ plays the following attack game:

1. The challenger generates a master key pair $(mpk, msk) = \mathtt{Setup}(1^k)$.
2. The attacker executes $\mathscr{A}_1$ on *mpk*. During its execution $\mathscr{A}_1$ may have access to certain oracles (described subsequently). $\mathscr{A}_1$ terminates by outputting an identity ID^*, two messages of equal length (m_0, m_1), and some state information *state*.
3. The challenger randomly chooses a bit $b \in \{0,1\}$ and a random string r, and computes

   $(vk^*, sk^*) = Sig.Gen(1^k)$
   $C_1^* = ID.Encrypt^{vk^*}(r, mpk, ID^*)$
   $C_2^* = PK.Encrypt^{vk^*}(m_b \oplus r, pk_{ID^*})$
   $\sigma^* = Sig.Sign((C_1^*, C_2^*), sk^*)$

   using the value of pk_{ID^*} currently associated with the identity ID^*. The challenge ciphertext is defined to be $C^* = (C_1^*, C_2^*, vk^*, \sigma^*)$.
4. The attacker executes $\mathscr{A}_2$ on the input $(C^*, state)$. During its execution $\mathscr{A}_2$ may have access to certain oracles (described subsequently). $\mathscr{A}_2$ terminates by outputting a guess b' for b.

The attacker wins the game if $b = b'$ and its advantage is defined to be:

$$Adv_{\mathscr{A}}^{\mathrm{CL}}(k) = |Pr[S_1] - 1/2| \tag{4}$$

The attacker has access to the Request Public Key, Replace Public Key, Extract Partial Private Key, Extract Secret Value, Weak SK Decrypt, and Decrypt oracles.

Game 2 In Game 2 the challenger changes the way the Decrypt and Weak SV Decrypt oracles work. In both cases, after the challenge ciphertext has been issued (i.e. in step 4 of the attack game), before attempting to decrypt a ciphertext (C_1, C_2, vk, σ) for the identity ID^*, the challenger checks to see if $vk = vk^*$. If so, then the challenger returns $\perp$ without decrypting the ciphertext. Otherwise the challenger decrypts the ciphertext as normal.

The attacker in Game 2 receives exactly the same information as the attacker in Game 2 unless $\mathscr{A}_2$ submits a ciphertext (C_1, C_2, vk^*, σ) to the Decrypt or Weak SV Decryption oracle for the identity ID^* and for which $(C_1, C_2, \sigma) \neq (C_1^*, C_2^*, \sigma^*)$ and $Sig.Verify(\sigma, (C_1, C_2), vk^*) = \top$. Let E be the event that the attacker makes such a query.

We claim that if $Pr[E]$ is non-negligible, then there exists an attacker $\mathscr{B}$ that breaks the unforgeability of the one-time signature scheme. $\mathscr{B}_1$ runs as follows:

1. $\mathscr{B}_1$ receives the challenge public key vk^*.
2. $\mathscr{B}_1$ generates a master key pair $(mpk, msk) = \mathtt{Setup}(1^k)$.
3. $\mathscr{B}_1$ executes $\mathscr{A}_1$ on the input *mpk*. If $\mathscr{A}$ makes any oracle queries, then $\mathscr{B}$ can answer them using its knowledge of the master private key *msk*. $\mathscr{A}_1$ terminates by outputting an identity ID^*, two messages of equal length (m_0, m_1) and (possibly) some state information.
4. $\mathscr{B}_1$ randomly chooses a bit $b \in \{0,1\}$, a random string r, and computes the challenge ciphertext

   $C_1^* = ID.Encrypt^{vk^*}(r, mpk, ID^*)$

   and

   $C_2^* = PK.Encrypt^{vk^*}(m_b \oplus r, pk_{ID^*})$

   using the value of pk_{ID^*} currently associated with the identity ID^*. $\mathscr{B}_1$ outputs the "message" (C_1^*, C_2^*) to the challenger.

The challenger will compute a signature σ^* on (C_1^*, C_2^*) using the signature key that corresponds to vk^*, and pass σ^* to the second phase of the attacker $\mathscr{B}_2$. $\mathscr{B}_2$ runs as follows:

1. $\mathscr{B}_2$ forms the complete challenge ciphertext $C^* = (C_1^*, C_2^*, vk^*, \sigma^*)$.
2. $\mathscr{B}_2$ executes $\mathscr{A}_2$ on the input C^* (and any state information output by $\mathscr{A}_1$). If $\mathscr{A}$ makes a Weak SV Decrypt or Decrypt oracle query, then $\mathscr{B}$ first checks whether it is a query on a ciphertext (C_1, C_2, vk^*, σ) where $(C_1, C_2, \sigma) \neq (C_1^*, C_2^*, \sigma^*)$ and

   $Sig.Verify(\sigma, (C_1, C_2), vk^*) = \top$.

   If so, $\mathscr{B}$ outputs σ as a forgery on the message (C_1, C_2). Otherwise, $\mathscr{B}$ answers the decryption oracle query using its knowledge of the master private key. $\mathscr{B}$ answers all other oracles using its knowledge of the master private key. $\mathscr{A}$ terminates by outputting a guess b' for b.
3. B_2 outputs $\perp$.

It is clear to see that $\mathscr{B}$ correctly simulates the environment for $\mathscr{A}$ and that $\mathscr{B}$ wins the unforgeability game if and only if E occurs. Hence, $Pr[E] = Adv_{\mathscr{B}}^{\mathrm{Sig}}(k)$, which is negligible by assumption.

Hence, we have that $|Pr[S_1] - Pr[S_2]| \leq Pr[E] = Adv_{\mathscr{B}}^{\mathrm{Sig}}(k)$.

Game 3 In Game 3 we introduce our guess as to whether the attacker will request the partial private key for the challenge identity or not. The attack game now becomes

1. The challenger generates a guess $c \in \{0,1\}$
2. The challenger generates a master key pair $(mpk, msk) = \mathtt{Setup}(1^k)$.
3. The attacker executes $\mathscr{A}_1$ on *mpk*. $\mathscr{A}_1$ terminates by outputting an identity ID^*, two messages of equal length (m_0, m_1), and some state information *state*.
4. The challenger randomly chooses a bit $b \in \{0,1\}$ and computes the challenge ciphertext by choosing a random string r and computing

   $(vk^*, sk^*) = Sig.Gen(1^k)$
   $C_1^* = ID.Encrypt^{vk^*}(r, mpk, ID^*)$
   $C_2^* = PK.Encrypt^{vk^*}(m_b \oplus r, pk_{ID^*})$
   $\sigma^* = Sig.Sign((C_1^*, C_2^*), sk^*)$

   using the value of pk_{ID^*} currently associated with the identity ID^*. The challenge ciphertext is defined to be $C^* = (C_1^*, C_2^*, vk^*, \sigma^*)$.
5. The attacker executes $\mathscr{A}_2$ on the input $(C^*, state)$. $\mathscr{A}_2$ terminates by outputting a guess b' for b.

We also change the conditions in which the attacker wins the game. Let F be the event that $\mathscr{A}$ queries the Extract Partial Private Key oracle on the challenge identity.

- If $c = 0$ and F occurs, then the attacker wins the game if $b' = b$,
- If $c = 1$ and F occurs, then the attacker wins the game with probability 1/2 (i.e. the value b' is ignored and the attacker is assumed to have output a random guess for b),
- If $c = 0$ and $\neg F$ occurs, then the attackers win the game with probability 1/2 (as above),
- If $c = 1$ and $\neg F$ occurs, then the attacker wins the game if $b' = b$.

This means that exactly half the time, the attacker's guess is ignored and we assume the attacker outputs correct guess for b with probability 1/2. We show that the attacker's advantage in Game 3 is exactly half of that in Game 2. Note that the events S_2, F and $c = 0$ are independent, but that S_3 is not independent of either F or $c = 0$.

$$\begin{aligned}
Pr[S_3] &= Pr[S_3 \mid F \wedge c = 0]Pr[F \wedge c = 0] \\
&\quad + Pr[S_3 \mid F \wedge c = 1]Pr[F \wedge c = 1] \\
&\quad + Pr[S_3 \mid \neg F \wedge c = 0]Pr[\neg F \wedge c = 0] \\
&\quad + Pr[S_3 \mid \neg F \wedge c = 1]Pr[\neg F \wedge c = 1] \\
&= \frac{1}{2}Pr[S_3 \mid F \wedge c = 0]Pr[F] \\
&\quad + \frac{1}{2}Pr[S_3 \mid F \wedge c = 1]Pr[F] \\
&\quad + \frac{1}{2}Pr[S_3 \mid \neg F \wedge c = 0]Pr[\neg F] \\
&\quad + \frac{1}{2}Pr[S_3 \mid \neg F \wedge c = 1]Pr[\neg F] \\
&= \frac{1}{2}Pr[S_3 \mid F \wedge c = 0]Pr[F] \\
&\quad + \frac{1}{2}Pr[S_3 \mid \neg F \wedge c = 1]Pr[\neg F] \\
&\quad + \frac{1}{4}Pr[F] + \frac{1}{4}Pr[\neg F] \\
&= \frac{1}{2}Pr[S_3 \mid F \wedge c = 0]Pr[F] \\
&\quad + \frac{1}{2}Pr[S_3 \mid \neg F \wedge c = 1]Pr[\neg F] + \frac{1}{4} \\
&= \frac{1}{2}Pr[S_2 \mid F \wedge c = 0]Pr[F]
\end{aligned}$$

$$+\frac{1}{2}Pr[S_2 \mid \neg F \wedge c=1]Pr[\neg F]+\frac{1}{4}$$
$$= \frac{1}{2}Pr[S_2 \mid F]Pr[F]+\frac{1}{2}Pr[S_2 \mid \neg F]Pr[\neg F]+\frac{1}{4}$$
$$= \frac{1}{2}Pr[S_2]+\frac{1}{4}\ .$$

Therefore,

$$|Pr[S_3]-\frac{1}{2}| = \frac{1}{2}|Pr[S_2]-\frac{1}{2}|\ .$$

Thus, if we can show that $\mathscr{A}$'s advantage in Game 3 is the negligible, then we can infer that $\mathscr{A}$'s advantage in Game 2 is negligible. In order to do this we note that

$$|Pr[S_3]-\frac{1}{2}| = |\frac{1}{2}Pr[S_3|c=0]+\frac{1}{2}Pr[S_3|c=1]-\frac{1}{2}|$$
$$\leq \frac{1}{2}|Pr[S_3|c=0]-\frac{1}{2}|+\frac{1}{2}|Pr[S_3|c=1]-\frac{1}{2}|\ .$$

We now show that there exists an attacker $\mathscr{B}$ against the IND-CCA2 security of the public-key encryption scheme that has advantage related to $|Pr[S_3|c=0]-1/2|$ and an attacker $\mathscr{B}$ against the IND-ID-CCA2 security of the identity-based encryption scheme that has advantage $|Pr[S_3|c=1]-1/2|$. The assumption that the public-key encryption scheme and the identity-based encryption scheme are secure then completes the proof.

The case when $c=0$. We reduce the security of the scheme to the security of the public key encryption scheme. Hence, we construct an adversary $\mathscr{B}$ for the public key encryption scheme that has an advantage that is non-negligibly related to $|Pr[S_3|c=0]-1/2|$. For this proof, it is important to recall that if $\mathscr{A}$ extracts the partial private key of the challenge identity then $\mathscr{A}$ will not replace the public key of the challenge identity or query the secret value oracle on the challenge identity.

The algorithm $\mathscr{B}_1$ runs as follows. For simplicity we suppress the *state* variables that are passed between the algorithms.

1. Receive the challenge public key pk^*.
2. Randomly choose an index $i \in \{1,2,\ldots,q_{req}\}$.
3. Generate a master key pair $(mpk,msk) = ID.Gen(1^k)$.
4. Execute $\mathscr{A}_1$ on the input mpk. $\mathscr{B}$ simulates $\mathscr{A}$'s oracle as follows:
   – Request Public Key. If this is the i-th request to the request public key oracle, then $\mathscr{B}$ returns pk^*. Otherwise, $\mathscr{B}$ generates a key pair $(pk_j,sk_j) = PK.Gen(1^k)$, stores the entire key pair and returns pk_j.
   – Replace Public Key. $\mathscr{B}$ stores the new public key value pk for the identity (along with the existing original public/private key values).
   – Extract Partial Private Key. $\mathscr{B}$ computes the partial private key $d_{ID} = ID.Extract(ID,msk)$ and returns the result to $\mathscr{A}$.
   – Extract Secret Value. $\mathscr{B}$ returns the private key sk_j associated with the identity ID whenever $j \neq i$. If $j=i$ then we have guessed the index i incorrectly and $\mathscr{B}$ terminates by outputting a random bit b'.
   – Weak SV Decrypt. Since $\mathscr{B}$ knows the partial private key for every identity, and $\mathscr{A}$ must supply the secret value sk for the identity (i.e. the decryption key for the public-key encryption scheme component), $\mathscr{B}$ can compute weak decryptions using the normal decryption algorithm.
   – Decrypt. If $\mathscr{A}$ queries the decrypt oracle on an identity other than the i-th identity queried to the request public key oracle, then $\mathscr{B}$ can compute decryptions using the normal decryption algorithm. If $\mathscr{A}$ makes a decryption oracle query on the i-th identity, then the identity-based share s_1 can be recovered using the partial private key for ID and the public key share s_2 can be recovered using $\mathscr{B}$'s decryption oracle. Thus, $\mathscr{B}$ can simulate the decryption algorithm.

   $\mathscr{A}_1$ terminates by outputting two equal length messages (m_0,m_1) and an identity ID^*.
5. If the identity ID^* was not the i-th query to the request public key oracle, then $\mathscr{B}$ aborts and outputs $\perp$.
6. Compute $(vk^*,sk^*) = Sig.Sign(1^k)$.
7. Choose a random string r of the same length as m_0
8. Output the two message $(m_0 \oplus r, m_1 \oplus r)$ and the label vk^*.

The challenger will now pick a random bit b and compute

$$C_2^* = PK.Encrypt^{vk^*}(m_b \oplus r, pk^*)\ .$$

$\mathscr{B}_2$ runs as follows:

1. Receive the challenge ciphertext C_2^*.
2. Compute $C_1^* = ID.Encrypt^{vk^*}(r,ID)$.
3. Compute $\sigma^* = Sig.Sign((C_1,C_2),sk^*)$.
4. Set $C^* = (C_1^*,C_2^*,vk^*,\sigma^*)$.
5. Execute $\mathscr{A}_2$ on C^*. $\mathscr{B}$ can answer all oracle queries that $\mathscr{A}$ makes using the same algorithms as before, except for the weak decryption and decryption oracles. $\mathscr{B}$ answers $\mathscr{A}$ these oracle queries as follows.
   – Weak SV Decrypt. If the query is for the identity ID^* and on a ciphertext (C_1,C_2,vk^*,σ), then $\mathscr{B}$ returns $\perp$ to $\mathscr{A}$. Otherwise, $\mathscr{B}$ decrypts the ciphertext as before. Note that we may still make the decryption oracle query as we will be querying the oracle with a label $vk \neq vk^*$; hence, the query will always be legal.
   – Decrypt. If the query is for the identity ID^* and on a ciphertext (C_1,C_2,vk^*,σ), then $\mathscr{B}$ returns $\perp$ to $\mathscr{A}$. Otherwise, $\mathscr{B}$ decrypts the ciphertext as before. Note that we may still make the decryption oracle query as we will be querying the oracle with a label $vk \neq vk^*$; hence, the query will always be legal.

   $\mathscr{A}_2$ terminates by outputting a guess b' for b.
6. If $\mathscr{A}$ has not queried the extract partial private key oracle on the challenge identity ID^*, then output a random bit. Otherwise output b'.

Note that $\mathscr{B}$ perfectly simulates the oracles to which $\mathscr{A}$ has access (owing to the fact that we disallowed decryption oracle queries for which $vk = vk^*$ in Game 2). Furthermore, if $\mathscr{A}$ succeeded in breaking the security of the certificateless encryption scheme in Game 3 (when $c=0$) and $\mathscr{B}$ chooses the correct index i for the challenge identity, then $\mathscr{B}$ succeeds in breaking the IND-CCA2 security of the public-key encryption scheme. Hence,

$$Adv_{\mathscr{B}}^{PK}(k) \geq \frac{|Pr[S_3|c=0]-1/2|}{q_{req}}$$

which is negligible by assumption.

The case when c=1 We reduce the security of the scheme to the security of the identity-based encryption scheme. Hence, we construct an adversary $\mathscr{B}$ for the identity-based encryption scheme that has an advantage which is non-negligibly related to $|Pr[S_3|c=1]-1/2|$. It is important to recall that if $\mathscr{A}$ replaces the challenge public key, then $\mathscr{A}$ cannot extract the partial private key of the challenge identity.

The algorithm $\mathscr{B}_1$ runs as follows.

1. Receive the master public key mpk.
2. Execute $\mathscr{A}_1$ on the input mpk. $\mathscr{B}$ simulates $\mathscr{A}$'s oracles as follows:
   – Request Public Key. $\mathscr{B}$ simply executes $PK.Gen$ to generate a new key pair (pk,sk), stores both parts, and returns pk.
   – Replace Public Key. $\mathscr{B}$ stores the new public key value pk for the identity (along with the existing original key pair).
   – Extract partial private key. $\mathscr{B}$ queries its $ID.Extract$ oracle and returns the result.
   – Extract Secret Value. $\mathscr{B}$ returns the private key sk associated with the identity.
   – Weak SV Decrypt. $\mathscr{B}$ follows the decryption algorithm for the certificateless encryption scheme, except that it uses its decryption oracle to obtain the decryption of C_1.
   – Decrypt. $\mathscr{B}$ follows the decryption algorithm for the certificateless encryption scheme, except that it uses its decryption oracle to obtain the decryption of C_1.

   $\mathscr{A}_1$ terminates by outputting two equal-length messages (m_0,m_1) and a challenge identity ID^*.
3. If $\mathscr{A}$ has queried the extract partial private key oracle for ID^* then $\mathscr{B}$ aborts and outputs a random bit.

4. $\mathscr{B}$ generates $(vk^*, sk^*) = Sig.Sign(1^k)$.
5. $\mathscr{B}$ generates a random string r of the same length as m_0.
6. $\mathscr{B}_1$ outputs the two messages $(m_1 \oplus r, m_0 \oplus r)$, the label vk^* and the identity ID^*.

The challenger will pick a random bit b and compute

$C_1^* = ID.Encrypt^{vk^*}(m_{1-b} \oplus r, ID)$.

$\mathscr{B}_2$ runs as follows:

1. Receive the challenge ciphertext C_1^*.
2. Compute $C_2^* = PK.Encrypt^{vk^*}(m_0 \oplus m_1 \oplus r, pk_{ID})$.
3. Compute $\sigma^* = Sig.Sign((C_1, C_2), sk^*)$.
4. Set $C^* = (C_1^*, C_2^*, vk^*, \sigma^*)$.
5. Executes $\mathscr{A}_2$ on C^*. $\mathscr{B}$ answers all of $\mathscr{A}$'s oracle queries exactly as before, except for the extract partial private key, weak decrypt and decrypt oracles. $\mathscr{B}$ answers these oracle queries as follows:
   – Extract Partial Private Key. If $\mathscr{A}$ queries the extract partial private key oracle on ID^*, then $\mathscr{B}$ aborts and outputs a random bit. Otherwise $\mathscr{B}$ answers the query as before.
   – Weak SV Decrypt. If $\mathscr{A}$ queries the weak decrypt oracle with a ciphertext (C_1, C_2, vk^*, σ) for the identity ID^*, then $\mathscr{B}$ returns $\perp$. Otherwise, $\mathscr{B}$ answers the oracle query as before. Note that we may still make the decryption oracle query as we will be querying the oracle with a label $vk \neq vk^*$; hence, the query will always be legal.
   – Decrypt. If $\mathscr{A}$ queries the decrypt oracle with a ciphertext (C_1, C_2, vk^*, σ) for the identity ID^*, then $\mathscr{B}$ returns $\perp$. Otherwise, $\mathscr{B}$ answers the oracle query as before. Note that we may still make the decryption oracle query as we will be querying the oracle with a label $vk \neq vk^*$; hence, the query will always be legal.

   $\mathscr{A}_2$ terminates by outputting a guess b' for b.
6. Output b'.

There are several things to note about $\mathscr{B}$'s simulation of the attack game for $\mathscr{A}$. First, note that $\mathscr{B}$ perfectly simulates the oracles that $\mathscr{A}$ has access to in Game 3. Second, note that if the challenger picks the bit $b = 0$, then $\mathscr{B}$ will receive the encryption of the share $s_1 = m_1 \oplus r$ and will construct the encryption of the share $s_2 = m_0 \oplus m_1 \oplus r$. Since s_1 is randomly distributed and $s_1 \oplus s_2 = m_0$, the attacker $\mathscr{A}$ receives a valid and correctly distributed encryption of m_0. Similarly, if the challenger picks the bit $b = 1$, then $\mathscr{A}$ receives a valid and correctly distributed encryption of m_1. Now $\mathscr{B}$ succeeds in breaking the IND-ID-CCA2 security of the identity-based encryption scheme whenever $\mathscr{A}$ breaks the certificateless encryption scheme in the case where $c = 1$. Hence,

$Adv^{\mathrm{ID}}_{\mathscr{B}}(k) = |Pr[S_3|c = 1] - 1/2|$

which is negligible by assumption.

Conclusion Drawing together all of the various game hops, we obtain:

$$\begin{aligned} Adv^{\mathrm{CL}}_{\mathscr{A}}(k) &= |Pr[S_1] - 1/2| \\ &\leq |Pr[S_2] - 1/2| + Adv^{\mathrm{Sig}}_{\mathscr{B}}(k) \\ &\leq \frac{1}{2}|Pr[S_3] - 1/2| + Adv^{\mathrm{Sig}}_{\mathscr{B}}(k) \\ &\leq \frac{1}{2}|Pr[S_3|c = 0] - 1/2| + \frac{1}{2}|Pr[S_3|c = 1] - 1/2| + Adv^{\mathrm{Sig}}_{\mathscr{B}}(k) \\ &\leq \frac{q_{req}}{2} Adv^{\mathrm{PK}}_{\mathscr{B}}(k) + \frac{1}{2} Adv^{\mathrm{PK}}_{\mathscr{B}}(k) + Adv^{\mathrm{Sig}}_{\mathscr{B}}(k) \ . \end{aligned}$$

Therefore, the certificateless encryption scheme is Weak Type Ia† secure if the public-key encryption scheme, identity-based encryption scheme and one-time signature scheme are secure. □

A.3 Malicious Type II† Security

The proof that the Dodis-Katz scheme satisfies the Malicious Weak Type II† security notion is similar to (and substantially easier than) the proof that the scheme satisfies the Weak Type Ia† security notion when the attacker queries the extract partial private key oracle on the challenge identity. We reduce the security of the certificateless encryption scheme to the security of the public-key encryption scheme.

Theorem 2 *If there exists an attacker $\mathscr{A}$ that has advantage $Adv^{CL}_{\mathscr{A}}(k)$ in breaking the Dodis-Katz encryption scheme and that makes q_{req} queries to the Request Public Key oracle, then there exists*

– *an attacker $\mathscr{B}$ that has advantage $Adv^{PK}_{\mathscr{B}}(k)$ in breaking the underlying public-key encryption scheme,*
– *an attacker $\mathscr{B}$ that has probability $Adv^{Sig}_{\mathscr{B}}(k)$ in breaking the underlying one-time signature scheme*

such that

$$Adv^{CL}_{\mathscr{A}}(k) \leq q_{req} Adv^{PK}_{\mathscr{B}}(k) + Adv^{Sig}_{\mathscr{B}}(k) \ . \tag{5}$$

Proof Assume that there exists an attacker $\mathscr{A}$ that has advantage $Adv^{\mathrm{CL}}_{\mathscr{A}}(k)$ in breaking the CL-PKE scheme. Assume that $\mathscr{A}$ makes at most q_{req} queries to the request public key oracle. For simplicity we shall assume that $\mathscr{A}$ never makes a query to any other oracle on an identity before it has made a query to the request public key oracle on that identity, and that $\mathscr{A}$ will not output a challenge identity on which it has not queried the request public key. We will describe an attacker $\mathscr{B}$ for the public key encryption scheme that uses $\mathscr{A}$ as a subroutine and has an advantage $Adv^{\mathrm{PK}}_{\mathscr{B}}(k)$ that is non-negligibly related to $Adv^{\mathrm{CL}}_{\mathscr{A}}(k)$.

Recall that in the Malicious Type II† security model, the attacker generates the master public key mpk and has access to the Request Public Key, Extract Secret Value, and Weak PPK Decrypt oracles.

We use a simple game hopping technique [9,29] to prevent the attacker making certain decryption oracle queries that we would be unable to answer. Let S_i be the event that $\mathscr{A}$ wins Game i. Let Game 1 be the normal Malicious Weak Type II† attack game. Let Game 2 be the same game except that if the attacker submits a ciphertext $C = (C_1, C_2, vk^*, \sigma)$ to the Weak PPK Decrypt oracle after the challenge ciphertext has been issued such that $Sig.Verify(\sigma, (C_1, C_2), vk^*) = \top$ then the oracle returns $\perp$.

Just as in the proof of Theorem 1, we can show that there exists a probabilistic polynomial-time attacker $\mathscr{B}$ against the unforgeability of the one-time signature scheme such that $|Pr[S_1] - Pr[S_2]| \leq Adv^{\mathrm{Sig}}_{\mathscr{B}}(k)$.

We now show that if $\mathscr{A}$ wins Game 2 with non-negligible advantage then we can construct an attacker $\mathscr{B}$ that breaks the underlying encryption scheme with non-negligible advantage too. $\mathscr{B}_1$ runs as follows:

1. Receive the challenge public key pk^*.
2. Randomly choose an index $i \in \{1, 2, \ldots, q_{req}\}$.
3. Execute $\mathscr{A}_0$ on the input 1^k to obtain a maliciously generated master public key value mpk.
4. Execute $\mathscr{A}_1$ on the input mpk. $\mathscr{B}$ simulates $\mathscr{A}$'s oracle as follows:
   – Request Public Key. If this is the i-th request to the request public key oracle, then $\mathscr{B}$ returns pk^*. Otherwise, $\mathscr{B}$ generates a key pair $(pk_j, sk_j) = PK.Gen(1^k)$, stores the entire key pair and returns pk_j.
   – Extract Secret Value. $\mathscr{B}$ returns the private key sk_j associated with the identity ID whenever $j \neq i$. If $j = i$ then we have guessed the index i incorrectly and $\mathscr{B}$ terminates by outputting a random bit b'.
   – Weak PPK Decrypt. If $\mathscr{A}$ queries the Weak PPK Decrypt oracle on an identity other than the i-th identity queried to the Request Public Key oracle, then $\mathscr{A}$ can compute decryptions using the normal decryption algorithm. If $\mathscr{A}$ makes a Weak PPK Decrypt oracle query on the i-th identity, then the identity-based share s_1 can be computed using the (supplied) partial private key and the public key share s_2 can be computed using $\mathscr{B}$'s decryption oracle. Thus, $\mathscr{B}$ can simulate the decryption algorithm.

   $\mathscr{A}_1$ terminates by outputting two equal length messages (m_0, m_1) and an identity ID^*.
5. If the identity ID^* was not the i-th query to the request public key oracle, then $\mathscr{B}$ aborts and outputs $\perp$.
6. Compute $(vk^*, sk^*) = Sig.Sign(1^k)$.
7. Choose a random string r of the same length as m_0
8. Output the two message $(m_0 \oplus r, m_1 \oplus r)$ and the label vk^*.

The challenger will now pick a random bit b and compute

$C_2^* = PK.Encrypt^{vk^*}(m_b \oplus r, pk^*)$.

$\mathscr{B}_2$ runs as follows:

1. Receive the challenge ciphertext C_2^*.
2. Compute $C_1^* = ID.Encrypt^{vk^*}(r, ID)$.
3. Compute $\sigma^* = Sig.Sign((C_1, C_2), sk^*)$.
4. Set $C^* = (C_1^*, C_2^*, vk^*, \sigma^*)$.
5. Execute $\mathscr{A}_2$ on C^*. $\mathscr{B}$ can answer all oracle queries that $\mathscr{A}$ makes using the same algorithms as before, except for the Weak PPK Decrypt oracle. For this oracle, $\mathscr{B}$ answers $\mathscr{A}$'s query as follows.
   - Weak PPK Decrypt. If the query is for the identity ID^* and on a ciphertext (C_1, C_2, vk^*, σ), then $\mathscr{B}$ returns $\perp$ to $\mathscr{A}$. Otherwise, $\mathscr{B}$ decrypts the ciphertext as before. Note that we may still make the decryption oracle query as we will be querying the oracle with a label $vk \neq vk^*$; hence, the query will always be legal.

   $\mathscr{A}_2$ terminates by outputting a guess b' for b.
6. Output b'.

Note that $\mathscr{B}$ perfectly simulates the oracles for $\mathscr{A}$ and $\mathscr{B}$ successfully breaks the encryption scheme if and only if it picked the correct index i and $\mathscr{A}$ broke the certificateless encryption scheme. Hence, $|Pr[S_2] - 1/2| \leq q_{req} \cdot Adv_{\mathscr{B}}^{\mathrm{PK}}(k)$.

Thus we have that $Adv_{\mathscr{A}}^{\mathrm{CL}}(k) \leq Adv_{\mathscr{B}}^{\mathrm{Sig}}(k) + q_{req} \cdot Adv_{\mathscr{B}}^{\mathrm{PK}}$. □