CBE from CL-PKE: A Generic Construction and Efficient Schemes

Sattam S. Al-Riyami and Kenneth G. Paterson

Information Security Group,
Royal Holloway, University of London,
Egham, Surrey, TW20 0EX,
United Kingdom.
sattam@gmail.com,kenny.paterson@rhul.ac.uk

Abstract. We present a new Certificateless Public Key Encryption (CL-PKE) scheme whose security is proven to rest on the hardness of the Bilinear Diffie-Hellman Problem (BDHP) and that is more efficient than the original scheme of Al-Riyami and Paterson. We then give an analysis of Gentry's Certificate Based Encryption (CBE) concept, repairing a number of problems with the original definition and security model for CBE. We provide a generic conversion showing that a secure CBE scheme can be constructed from any secure CL-PKE scheme. We apply this result to our new efficient CL-PKE scheme to obtain a CBE scheme that improves on the original scheme of Gentry.

Keywords: Certificateless Public Key Encryption, CL-PKE, Certificate based Encryption, CBE, pairings.

1 Introduction

Gentry introduced the concept of Certificate Based Encryption (CBE) in [7]. His concept provides an efficient implicit certification mechanism for PKI and allows a form of automatic revocation. Independently, [2] introduced and developed the notion of certificateless public key cryptography (CL-PKC). CL-PKC is designed to overcome the key-escrow limitation of identity-based cryptography [9] without introducing certificates and the management overheads that this entails. CL-PKC is a model for the use of public key cryptography that is intermediate between the identity-based and traditional PKI approaches.

On the surface, CBE and CL-PKC appear to be quite different. In [2], it was recognized, though not explored in any detail, that the two concepts of CBE and certificateless public key encryption (CL-PKE) are in fact related. In this paper, we revisit the work of [2] and [7], providing more efficient schemes and exploring the connections between the concepts of CBE and CL-PKE.

Our first contribution is a new certificateless public key encryption (CL-PKE) scheme that improves on the main scheme of [2] in two ways. Firstly our scheme is more efficient than the scheme in [2]. Secondly, we show that the security of the new scheme rests on the hardness of the Bilinear Diffie-Hellman Problem (BDHP), rather than the non-standard generalized BDHP that was the basis of security for the scheme in [2]. Our security result is proved in the full security model of [2].

Our second contribution is to provide a detailed analysis of the CBE concept of [7]. We point out a number of shortcomings in the definition of CBE as given in [7]. We repair these and then examine Gentry's security model for CBE. Comparing it to the model for CL-PKE given in [2] justifies us in making small changes to Gentry's security model. These still allow the security model to capture the kinds of attacks seen in real-world applications.

The small changes we make to Gentry's model also allow us to make our third contribution: a generic conversion that takes any CL-PKE scheme as input and produces from it a CBE scheme. The security of the CBE scheme in our adaptation of Gentry's model is tightly related to that of the CL-PKE scheme in the security model of [2]. Our result shows that the two concepts – CBE and CL-PKE – are indeed closely connected. We go on to explain why a generic construction going in the opposite direction, starting with a secure CBE scheme and yielding a secure CL-PKE scheme, is unlikely to be forthcoming.

Finally, we apply the generic conversion with our new CL-PKE scheme as input. The result is a secure CBE scheme that is more efficient than the original concrete scheme of [7].

1.1 Related Work

Kang, Park and Hahn [8] considered the signature analogue of certificate-based encryption. Since any certifying information can always be sent along with the actual signature, this concept seems less useful than that of CBE. Yum and Lee [11] gave a generic construction or a certificateless signature scheme from an ID-based signature scheme, proving the former to be secure (in an appropriate model) if the latter is. These authors also considered a generic construction for CL-PKE from identity-based encryption (IBE) [10] and the relationships between IBE, CBE and CL-PKE [12]. However, none of the results concerning the security of CL-PKE schemes proved in [10, 12] actually establishes security in the full security model developed in [2]: certain additional restrictions are always placed on the adversaries. For example, the Type I CL-PKE adversaries in [10, 12] are never permitted to extract the partial private key for the challenge identity. This restriction limiting the power of the adversary was not imposed in [2]. Moreover, no attempt is made in [10, 12] to properly handle decryption queries for identities whose public keys have been changed by the adversary. This issue was dealt with in [2] for concrete schemes by developing novel knowledge-extraction techniques. Thus the generic construction of CL-PKE schemes, secure in the full model of [2], from IBE or CBE schemes, remains an open problem. We return to this issue in Section 5.1.

2 Certificateless Public Key Encryption

In this section, we review the definition and security model for CL-PKE from [2]. We also provide some criticisms of the scheme in [2].

Definition 1. *[2] A CL-PKE scheme is specified by seven algorithms* (Setup, Partial-Private-Key-Extract, Set-Secret-Value, Set-Private-Key, Set-Public-Key, Encrypt, Decrypt) *such that:*

- Setup *is a probabilistic algorithm that takes security parameter k as input and returns the system parameters* params *and* master-key*. The system parameters includes a description of the message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$.*
- Partial-Private-Key-Extract *is a deterministic algorithm that takes* params, master-key *and an identifier for entity A,* $\mathsf{ID}_A \in \{0,1\}^*$*, as inputs. It returns a partial private key D_A.*
- Set-Secret-Value *is a probabilistic algorithm that takes as input* params[1] *and outputs a secret value x_A.*
- Set-Private-Key *is a deterministic algorithm that takes* params*, D_A and x_A as input. The algorithm returns S_A, a (full) private key.*
- Set-Public-Key *is a deterministic algorithm that takes* params *and x_A as input and outputs a public key P_A.*
- Encrypt *is a probabilistic algorithm that takes* params*, $M \in \mathcal{M}$, P_A and* ID_A *as inputs and returns either a ciphertext $C \in \mathcal{C}$ or the null symbol $\perp$ indicating an encryption failure*[2].
- Decrypt *is a deterministic algorithm that takes as inputs* params*, $C \in \mathcal{C}$ and S_A. It returns a message $M \in \mathcal{M}$ or a message $\perp$ indicating a decryption failure.*

Naturally, an output M should result from applying algorithm Decrypt *with inputs* params*, S_A on a ciphertext C generated by using algorithm* Encrypt *with inputs* params*, P_A,* ID_A *on message M.*

Algorithms Set-Private-Key and Set-Public-Key are normally run by an entity A for itself, after running Set-Secret-Value. Usually, A is the only entity in possession of S_A and x_A. Algorithms Setup and Partial-Private-Key-Extract are usually run by a trusted third party, called a key generating centre (KGC) [2].

2.1 Security Model for CL-PKE

The full IND-CCA security model for CL-PKE of [2] is an extension of the IND-ID-CCA model for IBE described in [4]. Below, we list the actions that an

[1] Note that the original definition of this algorithm in [2] also takes ID_A as input; however this string is not used in defining x_A in any concrete schemes, so we omit it here.

[2] The concrete encryption schemes in [2] could fail because the public key fails to have the correct structure. A general encryption algorithm could fail because the public key is not in the right group, for example.

IND-CCA adversary $\mathcal{A}$ against a CL-PKE scheme may carry out and discuss how each action should be handled by the challenger for that adversary.

1. **Extract partial private key of entity** A**:** Challenger $\mathcal{C}$ responds by running algorithm Partial-Private-Key-Extract to generate D_A for entity A.
2. **Extract private key for entity** A**:** If A's public key has not been replaced then $\mathcal{C}$ can respond by running algorithm Set-Private-Key to generate the private key S_A for entity A. It is assumed, as in [2], that the adversary does not make such queries for entities whose public keys have been changed.
3. **Request public key of entity** A**:** $\mathcal{C}$ responds by running algorithm Set-Public-Key to generate the public key P_A for entity A (first running Set-Secret-Value for A if necessary).
4. **Replace public key of entity** A**:** Adversary $\mathcal{A}$ can repeatedly replace the public key P_A for any entity A with any value P'_A of its choice. The current value of an entity's public key is used by $\mathcal{C}$ in any computations or responses to $\mathcal{A}$'s requests.
5. **Decryption query for ciphertext** C **and entity** A**:** In the model of [2], adversary $\mathcal{A}$ can issue a decryption query for any entity and any ciphertext. It is assumed in [2] that $\mathcal{C}$ should properly decrypt ciphertexts, even for those entities whose public keys have been replaced. This is a rather strong property for the security model (after all, the challenger may no longer know the correct private key). However, it ensures that the model captures the fact that changing an entity's public key to a value of the adversary's choosing may give that adversary an advantage in breaking the scheme. For further discussion of this feature, see [2].

The IND-CCA security model of [2] distinguishes two types of adversary. A Type I adversary is able to change public keys of entities at will, but does not have access to the master-key. A Type II adversary is equipped with master-key but is not allowed to replace public keys. This adversary models security against an eavesdropping KGC. The security game proceeds in three phases; in the middle challenge phase, the adversary selects a challenge identifier $\mathsf{ID}_{\mathrm{ch}}$ and corresponding public key P_{ch}, and is given a challenge ciphertext C^*. We provide a detailed description of the two adversary types and the security game next.

CL-PKE Type I IND-CCA Adversary: Adversary $\mathcal{A}_I$ does not have access to master-key. However, $\mathcal{A}_I$ may request public keys and replace public keys with values of its choice, extract partial private and private keys and make decryption queries, all for identities of its choice. $\mathcal{A}_I$ cannot extract the private key for $\mathsf{ID}_{\mathrm{ch}}$ at any point, nor request the private key for any identifier if the corresponding public key has already been replaced. $\mathcal{A}_I$ cannot both replace the public key for the challenge identifier $\mathsf{ID}_{\mathrm{ch}}$ before the challenge phase and extract the partial private key for $\mathsf{ID}_{\mathrm{ch}}$ in some phase. Furthermore, in Phase 2, $\mathcal{A}_I$ cannot make a decryption query on the challenge ciphertext C^* for the combination $(\mathsf{ID}_{\mathrm{ch}}, P_{\mathrm{ch}})$ that was used to encrypt M_b.

CL-PKE Type II IND-CCA Adversary: Adversary $\mathcal{A}_{II}$ does have access to master-key, but may not replace public keys of entities. Adversary $\mathcal{A}_{II}$ can compute partial private keys for itself, given master-key. It can also request public keys, make private key extraction queries and decryption queries, all for identities of its choice. The restrictions on this type of adversary are that it cannot replace public keys at any point, nor extract the private key for $\mathsf{ID}_{\mathrm{ch}}$ at any point. Additionally, in Phase 2, $\mathcal{A}_{II}$ cannot make a decryption query on the challenge ciphertext C^* for the combination $(\mathsf{ID}_{\mathrm{ch}}, P_{\mathrm{ch}})$ that was used to encrypt M_b.

Definition 2. *A CL-PKE scheme is said to be IND-CCA secure if no polynomially bounded adversary $\mathcal{A}$ of Type I or Type II has a non-negligible advantage in the following game:*

Setup: *$\mathcal{C}$ takes a security parameter k as input and runs the* Setup *algorithm. It gives $\mathcal{A}$ the resulting system parameters* params*. If $\mathcal{A}$ is of Type I, then $\mathcal{C}$ keeps* master-key *to itself, otherwise, it gives* master-key *to $\mathcal{A}$.*

Phase 1: *$\mathcal{A}$ issues a sequence of requests described above. These queries may be asked adaptively, but are subject to the rules on adversary behaviour defined above.*

Challenge Phase: *Once $\mathcal{A}$ decides that Phase 1 is over it outputs the challenge identifier $\mathsf{ID}_{\mathrm{ch}}$ and two equal length plaintexts $M_0, M_1 \in \mathcal{M}$. Again, the adversarial constraints given above apply. $\mathcal{C}$ now picks a random bit $b \in \{0,1\}$ and computes C^*, the encryption of M_b under the current public key P_{ch} for $\mathsf{ID}_{\mathrm{ch}}$. Then C^* is delivered to $\mathcal{A}$.*

Phase 2: *Now $\mathcal{A}$ issues a second sequence of requests as in Phase 1, again subject to the rules on adversary behaviour above.*

Guess: *Finally, $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$. The adversary wins the game if $b = b'$. We define $\mathcal{A}$'s advantage in this game to be $Adv(\mathcal{A}) := 2|\Pr[b = b'] - \frac{1}{2}|$.*

2.2 The Concrete Scheme of Al-Riyami and Paterson

In [2], we presented a concrete IND-CCA secure CL-PKE scheme, named FullCL-PKE. The scheme is an adaptation of the pairing-based IBE scheme of [4]; we do not replicate it here. Instead we list three drawbacks of the scheme FullCL-PKE:

1. FullCL-PKE requires three pairing calculations for each encryption (though two of these are required to check the structure of the public key and all three can be eliminated for any subsequent encryptions to the same party). It would be preferable to have a less computationally intensive CL-PKE scheme.
2. Each public key in FullCL-PKE consists of two elements of a group $\mathbb{G}_1$. Shorter keys would be preferable.
3. The security of FullCL-PKE rests on the hardness of the Generalized Bilinear Diffie-Hellman Problem (GBDHP). This problem is less well-established and no harder than the BDHP introduced in [4]. It would be preferable to have a CL-PKE scheme with a more solid security foundation. We will comment further on this point in Section 3.3.

3 A New CL-PKE Scheme

In this section we present a new CL-PKE scheme and study its security. The scheme can be regarded as resulting from the optimization of a double encryption construction for CL-PKE, using the IBE scheme of [4] and the ElGamal public key encryption scheme [5] as components. This immediately suggests a generic construction for a CL-PKE scheme from the combination of an IBE scheme and a (normal) public key encryption (PKE) scheme. Indeed such a construction is possible (and was first suggested to us by Boneh), but it seems to be difficult to prove the resulting scheme secure in the full model of [2] using standard security assumptions about the component IBE and PKE schemes. For that reason we have concentrated here on the concrete scheme and its proof of security.

Before we give our new scheme, we provide some background on pairings and a related computational problem.

3.1 Review of Pairings

Let $\mathbb{G}_1$ denote an additive group of prime order q and $\mathbb{G}_2$ a multiplicative group also of order q. We let P denote a generator of $\mathbb{G}_1$. A pairing is a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ with the following properties:

1. Bilinear: given any $Q, W \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q$, we have
$$\hat{e}(aQ, bW) = \hat{e}(Q, W)^{ab} = \hat{e}(abQ, W) \text{ etc.}$$
2. Non-degenerate: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$.
3. Efficiently computable.

The map $\hat{e}$ is usually derived from either the Weil or Tate pairing on an elliptic curve over a finite field; see [2, 4] for further details and references. The following computational problem was introduced in [4]:

Bilinear Diffie-Hellman Problem (BDHP): Let $\mathbb{G}_1$, $\mathbb{G}_2$, P and $\hat{e}$ be as above. The BDHP in $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$ is as follows: Given $\langle P, aP, bP, cP \rangle$ with uniformly random choices of $a, b, c \in \mathbb{Z}_q^*$, find $\hat{e}(P, P)^{abc} \in \mathbb{G}_2$.

BDH Parameter Generator: As in [4], a randomized algorithm $\mathcal{IG}$ is a BDH parameter generator if $\mathcal{IG}$: (1) takes as input security parameter $k \geq 1$, (2) runs in polynomial time in k, and (3) outputs the description of groups $\mathbb{G}_1$, $\mathbb{G}_2$ of prime order q and a pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Formally, the output of the algorithm $\mathcal{IG}(1^k)$ is $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$.

3.2 The New Scheme

The algorithms for our new CL-PKE scheme $\mathsf{FullCL\text{-}PKE}^\star$ are:

Setup**:** This algorithm runs as follows:

1. Run $\mathcal{IG}$ on input k to generate output $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$.
2. Choose an arbitrary generator $P \in \mathbb{G}_1$.

3. Select a random master-key $s \in \mathbb{Z}_q^*$ and set $P_0 = sP$.
4. Choose cryptographic hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1^*$, $H_2 : \mathbb{G}_2 \to \{0,1\}^n$, $H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_q^*$, $H_4 : \{0,1\}^n \to \{0,1\}^n$ and $H_5 : \mathbb{G}_1 \to \{0,1\}^n$. Here n will be the bit-length of plaintexts.

The system parameters are params= $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_0, H_1, H_2, H_3, H_4, H_5 \rangle$. The master-key is $s \in \mathbb{Z}_q^*$. The message space is $\mathcal{M} = \{0,1\}^n$ and the ciphertext space is $\mathcal{C} = \mathbb{G}_1 \times \{0,1\}^{2n}$.

Partial-Private-Key-Extract: This algorithm takes as input an identifier $\mathsf{ID}_A \in \{0,1\}^*$ for entity A, and carries out the following steps to construct the partial private key for A:

1. Compute $Q_A = H_1(\mathsf{ID}_A) \in \mathbb{G}_1^*$.
2. Output the partial private key $D_A = sQ_A \in \mathbb{G}_1^*$.

Set-Secret-Value: This algorithm takes as inputs params and an identifier ID_A. It selects a random $x_A \in \mathbb{Z}_q^*$ and outputs x_A as A's secret value.

Set-Private-Key: This algorithm takes as inputs params, entity A's partial private key D_A and A's secret value $x_A \in \mathbb{Z}_q^*$. The output of the algorithm is the pair $S_A = \langle D_A, x_A \rangle$. So the private key for A is just the pair consisting of the partial private key and the secret value.

Set-Public-Key: This algorithm takes params and entity A's secret value $x_A \in \mathbb{Z}_q^*$ as inputs and constructs A's public key as $P_A = x_A P$.

Encrypt: To encrypt $M \in \mathcal{M}$ for entity A with identifier $\mathsf{ID}_A \in \{0,1\}^*$ and a public key P_A, perform the following steps:

1. Check that P_A is in $\mathbb{G}_1^*$, if not output $\perp$.
2. Compute $Q_A = H_1(\mathsf{ID}_A) \in \mathbb{G}_1^*$.
3. Choose a random $\sigma \in \{0,1\}^n$.
4. Set $r = H_3(\sigma, M)$.
5. Compute and output the ciphertext:

$$C = \langle rP, \sigma \oplus H_2(\hat{e}(Q_A, P_0)^r) \oplus H_5(rP_A), M \oplus H_4(\sigma) \rangle.$$

Notice that $H_2(\hat{e}(Q_A, P_0)^r)$ is identical to the mask used in the IBE scheme in [4], while $H_5(rP_A)$ is a mask computed using the term rP_A used in the ElGamal encryption scheme.

Decrypt: Suppose $C = \langle U, V, W \rangle \in \mathcal{C}$. To decrypt this ciphertext using the private key $S_A = \langle D_A, x_A \rangle$:

1. Compute $V \oplus H_2(\hat{e}(D_A, U)) \oplus H_5(x_A U) = \sigma'$.
2. Compute $W \oplus H_4(\sigma') = M'$.
3. Set $r' = H_3(\sigma', M')$ and test if $U = r'P$. If not, output $\perp$ and reject the ciphertext. Otherwise, output M' as the decryption of C.

When C is a valid encryption of M using P_A and ID_A, it is easy to see that decrypting C will result in an output $M' = M$. This concludes the description of FullCL-PKE*.

Theorem 1. *Let hash functions H_i, $1 \leq i \leq 5$, be random oracles. Suppose further that there is no polynomially bounded algorithm that can solve the BDHP with non-negligible advantage. Then* FullCL-PKE* *is IND-CCA secure.*

A sketch of the proof of this result is given in Appendix A. A full proof can be found in [1].

3.3 Comparing FullCL-PKE* to FullCL-PKE

The scheme FullCL-PKE of [2] is in many ways superseded by our new scheme FullCL-PKE*:

1. The new scheme only requires one pairing computation per encryption. The only test of validity for public keys P_A in FullCL-PKE* is a simple group membership test $P_A \in \mathbb{G}_1^*$, while testing validity in FullCL-PKE requires two pairing computations. As with FullCL-PKE, the single pairing computation can be replaced with an exponentiation in $\mathbb{G}_2$ if repeated encryption to the same recipient is performed. Decryption costs for the two schemes are similar.
2. Public keys in FullCL-PKE* consist of only one element of $\mathbb{G}_1$ rather than the two required in FullCL-PKE.
3. FullCL-PKE* has better security guarantees as its security is related to the BDHP, rather than the GBDHP (in which the output is a pair $\langle Q, \hat{e}(P,Q)^{abc}\rangle$ for input aP, bP, cP). The GBDHP is an easier problem than the BDHP, in that an algorithm to solve the latter can be used to solve the former simply by setting $Q = P$. Note also that there do exist triples $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e}\rangle$ for which the GBDHP is trivial, but the BDGHP is presumed to be hard. For example, if the order q of $\mathbb{G}_1$ and $\mathbb{G}_2$ is not prime, but instead divisible by a small prime q_0, then to solve the GBDHP with non-negligible probability $1/q_0$, we can select $Q = \frac{q}{q_0}P$ and guess a solution $\langle Q, \hat{e}(P,Q)^x\rangle$, where x is picked at random from $\{0, 1, \ldots, q_0 - 1\}$. It would be interesting to determine if the BDHP and GBDHP are of equal hardness of in the groups of prime order usually selected for applications.

There is one further difference between FullCL-PKE and FullCL-PKE* that we note here. The public keys in the scheme FullCL-PKE, being of the form $\langle X_A = x_A P, Y_A = xP_0\rangle$, are constructed with reference to a specific P_0 and hence a particular KGC. Therefore, the decrypting party can mandate a particular centralised point of control (that is, a particular KGC) from whom its partial private keys will be obtained. On the other hand, the public keys in the scheme FullCL-PKE* do not have this restriction. This means that an encrypting party can select an arbitrary KGC (so long as it uses the same group $\mathbb{G}_1$ in its params as the decrypting party does – though this restriction can be removed using a slightly less efficient scheme) and force the decrypting party to obtain its partial private key from that KGC. The merits and demerits of this property are discussed further in [1].

4 Certificate-Based Encryption

We now turn to a discussion of Certificate-Based Encryption (CBE), as introduced in [7]. In certificate-updating CBE, a Certification Authority (CA) is responsible for pushing fresh certificates to clients in each time period. Informally, a client needs to be in possession of its current certificate in order to be able to decrypt ciphertexts sent to it by other parties during that time period.

We begin by noting some incompatibilities between the generic definition of CBE and the concrete CBE schemes in [7]. We then present a simplified and corrected definition for CBE.

1. As can be seen from the first property in [7, Definition 1], combining an IBE scheme with a standard public key encryption (PKE) scheme is *explicitly* required when building a CBE scheme. This limits the ways in which CBE schemes can be constructed and, as we shall see, is an unnecessary restriction.
2. The first property of [7, Definition 1] requires that PK_{IBE} be an identifiable element of the IBE scheme's parameters that can be labelled as a public key (notice that PK_{IBE} is also used as a distinct computational element during encryption). Given that not every IBE scheme need have this property, this definition limits the IBE schemes that can be used to build CBE schemes. Again, the limitation is unnecessary.
3. Although the combination of IBE and PKE is required by the generic definition of CBE, none of the concrete CBE schemes in [7] actually makes use of explicitly defined IBE or PKE schemes in their construction. It is fairly clear how the concrete CBE schemes have evolved from the IBE scheme of [4], but strictly speaking, none of them meet the generic definition in [7, Definition 1].
4. The generic definition of CBE in [7, Definition 1] uses six algorithms $\mathsf{Gen}_{\mathsf{IBE}}$, $\mathsf{Gen}_{\mathsf{PKE}}$, Upd1, Upd2, Enc and Dec, while the concrete schemes in [7, Section 3] use instead five algorithms Setup, Certification, Encryption and Decryption. Essentially algorithms Upd1 and Upd2 are combined to yield algorithm Certification, but there are no explicit key generation algorithms in the concrete schemes.

To summarize, there are incompatibilities in [7] between the definition of CBE on the one hand and the concrete CBE schemes on the other, as well as a number of unnecessary restrictions in the CBE definition. We now provide an alternative definition for CBE. The concrete schemes in [7] are compatible with our simplified definition.

Definition 3. *A (certificate-updating) CBE scheme is defined by six algorithms* (Setup*,* Set-Key-Pair*,* Certify*,* Consolidate*,* Enc*,* Dec) *such that:*

– Setup *is a probabilistic algorithm taking as input a security parameter k. It returns SK_{CA} (the certifier's master-key) and public parameters* params *that include the description of a string space Λ. Usually, this algorithm is run by the CA.*

- $\mathsf{Set\text{-}Key\text{-}Pair}$ *is a probabilistic algorithm that takes* params *as input. When run by a client, it returns a public key* PK *and a private key* SK.
- $\mathsf{Certify}$ *is a deterministic certification algorithm that takes as input* $\langle SK_{CA}$, params, τ, $\lambda \in \Lambda$, $PK\rangle$. *It returns* Cert'_τ, *which is sent to the client. Here* τ *is a string identifying a time period, while* λ *contains other information needed to certify the client such as the client's identifying information, and* PK *is a public key.*
- $\mathsf{Consolidate}$ *is a deterministic certificate consolidation algorithm taking as input* $\langle \mathsf{params}, \tau, \lambda, \mathsf{Cert}'_\tau\rangle$, *and optionally* $\mathsf{Cert}_{\tau-1}$. *It returns* Cert_τ, *the certificate used by a client in time period* τ.
- Enc *is a probabilistic algorithm taking as inputs* $\langle \tau, \lambda, \mathsf{params}, PK, M\rangle$, *where* $M \in \mathcal{M}$ *is a message. It returns a ciphertext* $C \in \mathcal{C}$ *for message* M[3].
- Dec *is a deterministic algorithm taking* $\langle \mathsf{params}, \mathsf{Cert}_\tau, SK, C\rangle$ *as input in time period* τ. *It returns either a message* $M \in \mathcal{M}$ *or the special symbol* $\perp$ *indicating a decryption failure.*

Naturally, we require that if C *is the result of applying algorithm* Enc *with input* $\langle \tau, \lambda, \mathsf{params}, PK, M\rangle$ *and* $\langle SK, PK\rangle$ *is a valid key-pair, then* M *is the result of applying algorithm* Dec *on input* $\langle \mathsf{params}, \mathsf{Cert}_\tau, SK, C\rangle$, *where* Cert_τ *is the output of the* $\mathsf{Certify}$ *and* $\mathsf{Consolidate}$ *algorithms on input* $\langle SK_{CA}$, params, τ, $\lambda \in \Lambda$, $PK\rangle$. *We write:*

$$\mathsf{Dec}_{\mathsf{Cert}_\tau, SK}(\mathsf{Enc}_{\tau,\lambda,PK}(M)) = M.$$

We note that a concrete CBE scheme need not involve certificate consolidation – see [7, Section 3] for examples. In this situation, algorithm $\mathsf{Consolidate}$ will simply output $\mathsf{Cert}_\tau = \mathsf{Cert}'_\tau$.

4.1 Security Model for CBE

In this section, we present an amended security model for CBE, in accordance with our new definition for CBE. We will comment later on how this model can be further strengthened; however it is not our intention here to completely overhaul the work of [7].

As in [7], security for CBE is defined using two different games and the adversary chooses which game to play. In Game 1, the adversary models an uncertified entity and in Game 2, the adversary models the certifier in possession of the master-key SK_{CA} attacking a fixed entity's public key.

CBE Game 1: The challenger runs Setup, gives params to the adversary $\mathcal{A}_1$ and keeps SK_{CA} to itself. The adversary then interleaves certification and decryption queries with a single challenge query. These queries are answered as follows:

- On certification query $\langle \tau, \lambda, PK, SK\rangle$, the challenger checks that $\lambda \in \Lambda$ and that $\langle PK, SK\rangle$ is a valid key-pair. If so, it runs $\mathsf{Certify}$ on input $\langle SK_{CA}$, params, τ, λ, $PK\rangle$ and returns Cert'_τ; else it returns $\perp$.

[3] We assume that Enc might also output $\perp$ if PK is not a valid public key.

– On decryption query $\langle \tau, \lambda, PK, SK, C \rangle$, the challenger checks that $\lambda \in \Lambda$ and that $\langle PK, SK \rangle$ is a valid key-pair. If so, it generates Cert_τ by using algorithms $\mathsf{Certify}$ and $\mathsf{Consolidate}$ with inputs $\langle SK_{CA}$, params, τ, λ, $PK \rangle$, and outputs $\mathsf{Dec}_{\mathsf{Cert}_\tau, SK}(C)$; else it returns $\perp$.
– On challenge query $\langle \tau_{\text{ch}}, \lambda_{\text{ch}}, PK_{\text{ch}}, SK_{\text{ch}}, M_0, M_1 \rangle$, where $M_0, M_1 \in \mathcal{M}$ are of equal length, the challenger checks that $\lambda_{\text{ch}} \in \Lambda$ and that $\langle PK_{\text{ch}}, SK_{\text{ch}} \rangle$ is a valid key-pair. If so, it chooses a random bit b and returns $C^* = \mathsf{Enc}_{\tau_{\text{ch}}, \lambda_{\text{ch}}, PK_{\text{ch}}}(M_b)$; else it returns $\perp$.

Finally, $\mathcal{A}_1$ outputs a guess $b' \in \{0, 1\}$. The adversary wins the game if $b = b'$ and $\langle \tau_{\text{ch}}, \lambda_{\text{ch}}, PK_{\text{ch}}, SK_{\text{ch}}, C^* \rangle$ was not the subject of a decryption query after the challenge, and $\langle \tau_{\text{ch}}, \lambda_{\text{ch}}, PK_{\text{ch}}, SK_{\text{ch}} \rangle$ was not the subject of any certification query. We define $\mathcal{A}_1$'s advantage in this game to be $\text{Adv}(\mathcal{A}_1) := 2|\Pr[b = b'] - \frac{1}{2}|$.

CBE Game 2: The challenger runs Setup, gives params and SK_{CA} to the adversary $\mathcal{A}_2$. The challenger then runs $\mathsf{Set\text{-}Key\text{-}Pair}$ to obtain a key-pair $\langle PK_{\text{ch}}, SK_{\text{ch}} \rangle$ and gives PK_{ch} to the adversary $\mathcal{A}_2$. The adversary then interleaves certification and decryption queries with a single challenge query. These queries are answered as follows:

– On decryption query $\langle \tau, \lambda, C \rangle$, the challenger checks that $\lambda \in \Lambda$. If not, it returns $\perp$. If so, it generates Cert_τ by using algorithms $\mathsf{Certify}$ and $\mathsf{Consolidate}$ with inputs $\langle SK_{CA}$, params, τ, λ, $PK_{\text{ch}} \rangle$. It then outputs $\mathsf{Dec}_{\mathsf{Cert}_\tau, SK_{\text{ch}}}(C)$.
– On challenge query $\langle \tau_{\text{ch}}, \lambda_{\text{ch}}, M_0, M_1 \rangle$, the challenger checks that $\lambda_{\text{ch}} \in \Lambda$. If so, it chooses random bit b and returns $C^* = \mathsf{Enc}_{\tau_{\text{ch}}, \lambda_{\text{ch}}, PK_{\text{ch}}}(M_b)$; else it returns $\perp$.

Finally, $\mathcal{A}_2$ outputs a guess $b' \in \{0, 1\}$. The adversary wins the game if $b = b'$ and $\langle \tau_{\text{ch}}, \lambda_{\text{ch}}, C^* \rangle$ was not the subject of a decryption query after the challenge. We define $\mathcal{A}_2$'s advantage in this game to be $\text{Adv}(\mathcal{A}_2) := 2|\Pr[b = b'] - \frac{1}{2}|$.

Definition 4. *A certificate-updating CBE scheme is said to be secure against adaptive chosen ciphertext attack (or IND-CBE-CCA secure) if no probabilistic polynomial-time adversary has non-negligible advantage in either CBE Game 1 or CBE Game 2.*

Let us now analyse the CBE security model, and compare it to Gentry's original model in [7]. The only technical difference between our CBE security model and that of [7] is in Game 2. In Gentry's model, the Game 2 adversary is allowed to specify a fresh params in each of its queries. It also supplies the CBE master-key SK_{CA} in decryption queries, so that the challenger is able to provide decryptions. In our model, params and the master-key SK_{CA} are fixed at the beginning of Game 2, and are supplied to the adversary. In both models, the Game 2 adversary attacks a fixed key-pair that is specified by the challenger. We argue that, while the model of [7] is more flexible in Game 2, our adaptation accurately models the kinds of attacks that might be attempted by a CA. One would not expect a CA to change its public parameters on a frequent basis; rather it is more natural to model a CA with fixed public parameters and in possession of the master-key.

In the next section, we will show a generic conversion from CL-PKE to CBE that preserves security. Essentially, this conversion maps CL-PKE to CBE by using extended identifiers in the CL-PKE scheme, while the certificates in the CBE scheme are obtained from the partial private keys in the CL-PKE scheme. The conversion process naturally highlights some strengths and weaknesses in the CBE security model of [7]:

1. A CBE Game 1 adversary must provide a private key SK along with the corresponding public key PK in all of its queries. This enables the challenger to handle decryption queries. By contrast, in CL-PKE, a Type I adversary is allowed to change an entity's public key without needing to show the private key. This gives the adversary more flexibility. For example, the adversary can replace the public key of one entity with that of another (without knowing the corresponding private key). The proofs of security for CL-PKE schemes in [3] and the full version of this paper handle decryption queries using special purpose knowledge extractors. The proof of security for the concrete scheme FullCBE in [7] is also able to remove the requirement of showing the private key.
2. A CBE Game 2 adversary does not get to choose a challenge public key to attack. Instead, it is given a specific public key by the challenger at the start of the game. This is unlike a CL-PKE Type II adversary, who has the freedom to work with multiple public keys and to select any one of them for the challenge query.
3. By contrast, the CBE Game 2 adversary in [7] is allowed to work with multiple values of params and the master-key. So this adversary can change the CBE scheme in *each* query. Both our CBE Game 2 adversary and a Type II CL-PKE adversary are given a fixed params and the master-key at the start of the game. This allows the adversary to 'break' that part of the scheme which the trusted third party is always able to break. We have already justified our making this restriction in CBE above.

5 CBE from CL-PKE

In this section, we present a construction for a CBE scheme using the algorithms of a generic CL-PKE scheme as components. After providing the construction, we prove that the resulting CBE scheme is IND-CBE-CCA secure (according to Definition 4), provided the CL-PKE scheme is IND-CCA secure (in the sense of Definition 2).

Suppose then that Π^{CL} is a CL-PKE scheme with algorithms $\mathsf{Setup}^{\mathrm{CL}}$, Partial-Private-Key-Extract, Set-Secret-Value, Set-Private-Key, Set-Public-Key, Encrypt and Decrypt. We define a CBE scheme Π^{CBE} by defining the six CBE algorithms ($\mathsf{Setup}^{\mathrm{CBE}}$, Set-Key-Pair, Certify, Consolidate, Enc, Dec) in terms of the CL-PKE algorithms.

– $\mathsf{Setup}^{\mathrm{CBE}}$: This algorithm takes a security parameter k and returns SK_{CA} and public parameters $\mathsf{params}^{\mathrm{CBE}}$ that includes the description of a string space Λ. We run algorithm $\mathsf{Setup}^{\mathrm{CL}}$ to obtain $\mathsf{master\text{-}key}^{\mathrm{CL}}$ and $\mathsf{params}^{\mathrm{CL}}$. We set SK_{CA} of Π^{CBE} to be $\mathsf{master\text{-}key}^{\mathrm{CL}}$. We allow Λ to be any subset of $\{0,1\}^*$. We then define $\mathsf{params}^{\mathrm{CBE}}$ by extending $\mathsf{params}^{\mathrm{CL}}$ to include a description of Λ.

- Set-Key-Pair: For a client A, this algorithm takes $\mathsf{params}^{\mathrm{CBE}}$ as input. We extract $\mathsf{params}^{\mathrm{CL}}$ from $\mathsf{params}^{\mathrm{CBE}}$ then run Set-Secret-Value and then Set-Public-Key of Π^{CL} to obtain values x_A and then P_A. The output $\langle PK, SK\rangle$ is defined to be the pair $\langle P_A, x_A\rangle$.
- Certify: This algorithm takes as input $\langle SK_{CA}, \mathsf{params}^{\mathrm{CBE}}, \tau, \lambda, PK\rangle$. We extract $\mathsf{params}^{\mathrm{CL}}$ from $\mathsf{params}^{\mathrm{CBE}}$ and obtain $\mathsf{master\text{-}key}^{\mathrm{CL}}$ from SK_{CA}. We then set $\mathsf{ID}'_A = \mathsf{params}^{\mathrm{CBE}}\|\tau\|\lambda\|PK$ and run algorithm Partial-Private-Key-Extract of Π^{CL} on input $\langle \mathsf{params}^{\mathrm{CL}}, \mathsf{master\text{-}key}^{\mathrm{CL}}, \mathsf{ID}'_A\rangle$ to obtain a partial private key D_A. The output Cert'_τ is defined to be D_A.
- Consolidate: This algorithm takes as input $\langle \mathsf{params}^{\mathrm{CBE}}, \tau, \lambda, \mathsf{Cert}'_\tau\rangle$. It simply outputs the value Cert'_τ.
- Enc: This algorithm takes $\langle \tau, \lambda, \mathsf{params}^{\mathrm{CBE}}, PK, M\rangle$ as input. Here, we assume $M \in \mathcal{M}$, the message space for Π^{CL}. We extract $\mathsf{params}^{\mathrm{CL}}$ from $\mathsf{params}^{\mathrm{CBE}}$. We set $\mathsf{ID}'_A = \mathsf{params}^{\mathrm{CBE}}\|\tau\|\lambda\|PK$ and use the Encrypt algorithm of Π^{CL} with input $\langle \mathsf{params}^{\mathrm{CL}}, M, PK, \mathsf{ID}'_A\rangle$ to obtain a ciphertext $C \in \mathcal{C}$. The output of Enc is defined to be C.
- Dec: This algorithm takes $\langle \mathsf{params}^{\mathrm{CBE}}, \mathsf{Cert}_\tau, SK, C\rangle$ as input in time period τ. We extract $\mathsf{params}^{\mathrm{CL}}$ from $\mathsf{params}^{\mathrm{CBE}}$, set $D_A = \mathsf{Cert}_\tau$, set $x_A = SK$, and run algorithm Set-Private-Key of Π^{CL} on input $\langle \mathsf{params}^{\mathrm{CL}}, D_A, x_A\rangle$ to obtain a private key S_A. Finally, we run algorithm Decrypt of Π^{CL} on input $\langle \mathsf{params}^{\mathrm{CL}}, C, S_A\rangle$ to obtain the output of algorithm Dec.

It is evident from the construction that the message and ciphertext spaces of Π^{CBE} are the same as those of Π^{CL}. It's also clear that partial private keys in Π^{CL} are (roughly speaking) transformed into certificates in Π^{CBE}. In the CBE scheme, we allow Λ to be any subset of $\{0,1\}^*$ for maximum flexibility, while identifiers of the form $\mathsf{params}^{\mathrm{CBE}}\|\tau\|\lambda\|PK$ where $\lambda \in \Lambda$ are used in the CL-PKE scheme.

Next is our main theorem about the IND-CBE-CCA security of the CBE scheme constructed using a CL-PKE scheme as above.

Theorem 2. *Suppose that Π^{CL} is an IND-CCA secure CL-PKE scheme, and suppose that Π^{CL} is used to build a CBE scheme Π^{CBE} as above. Then Π^{CBE} is IND-CBE-CCA secure.*

Proof. The proof can be found in Appendix B. The proof demonstrates a tight relationship between the advantage of a CBE adversary against Π^{CBE} and that of a CL-PKE adversary against Π^{CL}.

It can be seen from examining Appendix B that the security argument used there to prove the CBE scheme secure does not require the use of private keys SK from the CBE scheme in answering any queries. It is therefore possible to prove that Π^{CBE} is secure in a somewhat stronger security model than we have developed in Section 4.1. In the stronger model, the adversary is not required to show any private keys when making queries. As well as being stronger, this seems like a more natural model of real adversarial behaviour. The original model of [7] can also be strengthened in the same way. Thus we see that the CL-PKE concept can lead to improvements in the security of CBE schemes.

5.1 CL-PKE from CBE?

By composing the generic constructions of an IBE scheme from a CBE scheme and a CL-PKE scheme from an IBE scheme in [12], it is possible to use the six algorithms of a generic CBE scheme to construct a CL-PKE scheme. The overall construction uses the encryption and decryption algorithms of the IBE scheme twice in defining the corresponding algorithms of the CL-PKE scheme. The KGC is responsible for setting the keys and parameters for one pair of encryption and decryption algorithms, while individual users control the setting for the other pair. This ensures that the key-escrow property of the IBE scheme is overcome. Note, however, that the security results of [12] only establish the security of the CL-PKE scheme in a security model that is significantly weaker than the full CL-PKE security model developed in [2] and reproduced here in Section 2.1. So the generic construction of a secure CL-PKE scheme from a CBE scheme remains an open problem.

One might consider the direct construction of a CL-PKE scheme from a single instance of a generic CBE scheme, that is, without going via an intermediate IBE scheme and using the algorithms of the CBE scheme only once in defining the CL-PKE scheme. In a generic construction, the Partial-Private-Key-Extract algorithm of the CL-PKE scheme would presumably need to be constructed from the Certify algorithm of the CBE scheme. Then one obstacle to a generic construction is that a CBE scheme requires certain parameters (namely τ and PK) to be included in the inputs to the Certify algorithm, while these are not provided as inputs to the Partial-Private-Key-Extract algorithm in a CL-PKE scheme. This would mean that the necessary parameters would not in general be available as inputs to the Certify algorithm. This implies an important functional difference between the CBE and CL-PKE concepts: in CBE, the algorithm Set-Key-Pair needs to be run before Certify, while in CL-PKE, the corresponding algorithm Partial-Private-Key-Extract can be run before *or* after algorithm Set-Public-Key. In this respect, CL-PKE is more flexible than CBE.

We note that if one is prepared to consider only the special class of CL-PKE schemes in which identifiers include public keys, then one can construct a (special) CL-PKE scheme generically from a single instance of a CBE scheme. One trick needed in the construction is to set τ to a fixed value for every CBE certification query. This kind of CL-PKE scheme was considered in [2], where it was shown that the binding technique allows the CL-PKE scheme to attain a level of trust closer to that of a traditional PKI. Even so, to prove this scheme secure, one must further modify the CBE security model to remove the requirement on the adversary to supply private keys SK in queries. One must also restrict the

CL-PKE Type I adversary to not extract the partial private key for the challenge identifier, to prevent a corresponding CBE adversary from having to make a disallowed certification query. This means that the proof would not be in the full security model of [2].

It may well be possible to modify any particular concrete CBE scheme to produce a CL-PKE scheme that can be proven secure. For example, one might omit certain inputs to the Certify and Consolidate algorithms in order to define Partial-Private-Key-Extract. This is certainly true of the scheme FullCBE of [7]. However, this is not the same as obtaining a truly generic, security-preserving construction of one primitive from the other. Our discussion in this section points to the fact that, while similar in many respects, CBE and CL-PKE are not equivalent concepts. Indeed, we suspect that the generic construction of a fully secure CL-PKE scheme from a CBE scheme may be impossible. We reiterate that this does not contradict the result of Yum and Lee in [12], because they did not use the full security model of [2] when studying the security of their CL-PKE constructions.

5.2 A New CBE Scheme

Our generic construction in Section 5 applies to any CL-PKE scheme and produces an IND-CBE-CCA secure CBE scheme. For example, it can be applied to FullCL-PKE of [2] or to the scheme FullCL-PKE* developed in Section 3. Let us denote the CBE scheme obtained from FullCL-PKE* by FullCBE*. We do not give this scheme explicitly here. Instead we merely note that FullCBE* is more computationally efficient than the scheme FullCBE of [7], requiring only one pairing computation for encryption compared to the two needed in FullCBE.

6 Summary

In this paper, we have examined the relationship between the separate but related concepts of CBE [7] and CL-PKE [2]. We have given a generic construction producing a secure CBE scheme from a secure CL-PKE scheme. In order to obtain this construction, we have had to analyze and repair the CBE definition and security model. We have also given a new, secure CL-PKE scheme FullCL-PKE* that improves on the scheme FullCL-PKE of [2]. The generic construction applied to FullCL-PKE* produces the scheme FullCBE*, which is computationally superior to the scheme FullCBE of [7].

References

1. S.S. Al-Riyami, *Cryptographic schemes based on elliptic curve pairings*, Ph.D. thesis, University of London, 2004.
2. S.S. Al-Riyami and K.G. Paterson. Certificateless public key cryptography. In *Advances in Cryptology – ASIACRYPT 2003*, LNCS vol. 2894, pp. 452–473. Springer, 2003.

3. S.S. Al-Riyami and K.G. Paterson. Certificateless public key cryptography. Cryptology ePrint Archive, Report 2003/126, 2003. `http://eprint.iacr.org/`.
4. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *Proc. CRYPTO 2001*, LNCS vol. 2139, pp. 213–229. Springer, 2001.
5. T. ElGamal A public key cryptosystem and a signature scheme based on Discrete logarithm In G.R. Blakley and D. Chaum, editor, *Proc. CRYPTO 1984*, LNCS vol. 196, pp. 10–18. Springer, 1985.
6. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In M. J. Wiener, editor, *Proc. CRYPTO 1999*, LNCS vol. 1666, pp. 537–554. Springer, 1999.
7. C. Gentry. Certificate-based encryption and the certificate revocation problem. In E. Biham, editor, *Proc. EUROCRYPT 2003*, LNCS vol. 2656, pp. 272–293. Springer, 2003.
8. G. Kang, J.H. Park and S.H. Hahn. A certificate-based signature scheme. In *CT-RSA 2004*, LNCS vol. 2964, pp. 99–111, 2004.
9. A. Shamir. Identity-based cryptosystems and signature schemes. In *Proc. CRYPTO 1984*, LNCS vol. 196, pp. 47–53. Springer, 1984.
10. D.H. Yum and P.J. Lee. Generic construction of certificateless encryption. In *ICCSA 2004*, LNCS vol. 3043, pp. 802–811, 2004.
11. D.H. Yum and P.J. Lee. Generic construction of certificateless signature. In *ACISP 2004*, LNCS vol. 3108, pp. 200–211, 2004.
12. D.H. Yum and P.J. Lee. Identitiy-based cryptography in public key management. In *EuroPKI 2004*, LNCS vol. 3093, pp. 71–84, 2004.

A Sketch Proof of Theorem 1

We provide a sketch of the main ideas in the proof of Theorem 1; the details can be found in [1]. We need to introduce five intermediate encryption schemes. The first, BasicCL-PKE*, is a simpler version of FullCL-PKE* which omits the Fujisaki-Okamoto hybridisation technique [6]. We also make use of the schemes BasicPub and BasicPub^{hy} from [4], and we introduce two ElGamal-like schemes called ElG-BasicPub and ElG-HybridPub. The second of these is obtained from the first by applying the technique of [6]. A key-pair in ElG-BasicPub is of the form $\langle P_A, x_A \rangle$ and the encryption of message M is defined to be $C = \langle rP, M \oplus H_5(rP_A) \rangle$ for r selected at random from $\mathbb{Z}_q^*$.

As in [3], the proof of the theorem is performed in two parts. We relate the advantage of a Type I or Type II attacker against FullCL-PKE* to that of an algorithm to solve BDHP or CDHP respectively. We first consider a Type I adversary.

Type I adversary: We provide a reduction relating the IND-CCA security of FullCL-PKE* to the IND-CPA security of the standard PKE schemes ElG-HybridPub and BasicPub^{hy}. The reduction is similar to the one provided in [3], but simulates H_5 in a special way to ensure that it behaves consistently in the course of the attack. This reduction also makes use of a special-purpose knowledge extraction algorithm to handle decryption queries. Furthermore, in order for this knowledge extractor to have a high success probability, we require (and prove) that the scheme BasicCL-PKE is OWE secure if the BDHP is hard. Thereafter, we use a series of fairly standard results to reduce the security of ElG-HybridPub and $\mathsf{BasicPub}^{hy}$ to the hardness of the CDHP in $\mathbb{G}_1$ or BDHP in $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$, respectively.

Type II adversary: We show that the IND-CCA security of $\mathsf{FullCL\text{-}PKE}^{\star}$ can be reduced to the usual IND-CCA security of a related (normal) public key encryption scheme ElG-HybridPub. The security of ElG-HybridPub is reduced to that of a second public key encryption scheme ElG-BasicPub against OWE adversaries using results of [6]. Finally, we relate the security of ElG-BasicPub to the hardness of the CDHP in $\mathbb{G}_1$.

Since any algorithm to solve the CDHP in $\mathbb{G}_1$ (output by $\mathcal{IG}(k)$) can be used to solve the BDHP in $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$, we finally have that the security of $\mathsf{FullCL\text{-}PKE}^{\star}$ is related to the hardness of the BDHP. For the concrete relationship we refer the reader to the full version of this paper.

B Proof of Theorem 2

We begin by considering in detail the case of a Game 1 adversary against Π^{CBE}.

Let $\mathcal{A}_1$ be a Game 1 IND-CCA adversary against Π^{CBE} with advantage ϵ. We show how to construct from $\mathcal{A}_1$ a Type I IND-CCA adversary $\mathcal{B}_I$ against Π^{CL}. Let $\mathcal{C}$ denote a Π^{CL}-challenger against $\mathcal{B}_I$. $\mathcal{C}$ begins by supplying $\mathcal{B}_I$ with the parameters of Π^{CL}. $\mathcal{B}_I$ mounts an IND-CCA attack on Π^{CL} using help from $\mathcal{A}_1$ as follows.

Adversary $\mathcal{B}_I$ simulates the algorithm $\mathsf{Setup}^{\mathrm{CBE}}$ of Π^{CBE} for $\mathcal{A}_1$. This is done by $\mathcal{B}_I$ setting Λ to be an arbitrary subset of $\{0,1\}^*$ and $\mathsf{params}^{\mathrm{CBE}}$ to be an extension of $\mathsf{params}^{\mathrm{CL}}$ which includes a description of Λ. $\mathcal{B}_I$ then gives $\mathsf{params}^{\mathrm{CBE}}$ to $\mathcal{A}_1$. Now $\mathcal{A}_1$ launches its attack, and $\mathcal{B}_I$ launches Phase 1 of its attack. $\mathcal{A}_I$ interleaves queries of three types, during which $\mathcal{B}_I$ transitions from Phase 1 to the Challenge Phase and on to Phase 2 in a manner to be specified below. These queries are handled by $\mathcal{B}_I$ as follows:

- On certification query $\langle \tau, \lambda, PK, SK \rangle$, adversary $\mathcal{B}_I$ makes a replace public key query for the entity with identifier $\mathsf{ID}'_A = \mathsf{params}^{\mathrm{CBE}} \| \tau \| \lambda \| PK$, replacing the public key with the value PK. Then $\mathcal{B}_I$ makes a partial-private-key extract query to $\mathcal{C}$ for the identifier ID'_A and returns the resulting partial private key to $\mathcal{B}_I$.
- On decryption query $\langle \tau, \lambda, PK, SK, C \rangle$, $\mathcal{B}_I$ makes a replace public key query for the entity with identifier $\mathsf{ID}'_A = \mathsf{params}^{\mathrm{CBE}} \| \tau \| \lambda \| PK$, replacing the public key with the value PK. Then $\mathcal{B}_I$ makes a decryption query on ciphertext C for the entity with identifier ID'_A to $\mathcal{C}$. $\mathcal{B}_I$ relays $\mathcal{C}$'s response to $\mathcal{A}_1$.
- On receiving a challenge query $\langle \tau_{\mathrm{ch}}, \lambda_{\mathrm{ch}}, PK_{\mathrm{ch}}, SK_{\mathrm{ch}}, M_0, M_1 \rangle$, adversary $\mathcal{B}_I$ makes a replace public key query for the entity with identifier $\mathsf{ID}'_{\mathrm{ch}} = \mathsf{params}^{\mathrm{CBE}} \| \tau_{\mathrm{ch}} \| \lambda_{\mathrm{ch}} \| PK_{\mathrm{ch}}$, replacing the public key with the value PK_{ch}. Then $\mathcal{B}_I$ terminates Phase 1 of its attack and enters the challenge phase, sending $\mathsf{ID}'_{\mathrm{ch}}$ and messages M_0, M_1 to $\mathcal{C}$. Challenger $\mathcal{C}$ responds with a challenge ciphertext C^* which is the encryption of message M_b (for some bit b)

for identifier $\mathsf{ID}'_{\mathrm{ch}}$ and public key PK_{ch} in the scheme Π^{CL}. Then $\mathcal{B}_I$ forwards C^* to $\mathcal{A}_1$ as the response to A_1's challenge query and begins Phase 2 of its attack. It is easy to see from the definition of Π^{CBE} that C^* is equal to the output of algorithm Enc of Π^{CBE} on input $\langle \tau_{\mathrm{ch}}, \lambda_{\mathrm{ch}}, \mathsf{params}^{\mathrm{CBE}}, PK_{\mathrm{ch}}, M_b\rangle$.

We further insist that, if $\mathcal{B}_I$ is forced during the course of its simulation to replace the public key for $\mathsf{ID}'_{\mathrm{ch}}$ before the challenge phase and make a partial-private-key extract query on $\mathsf{ID}'_{\mathrm{ch}}$ in some phase, then $\mathcal{B}_I$ aborts. Likewise, we insist that if $\mathcal{B}_I$ is in Phase 2 and is forced to relay a decryption query on ciphertext C^* for identifier $\mathsf{ID}'_{\mathrm{ch}}$ and public key PK_{ch}, then $\mathcal{B}_I$ aborts. Since $\mathsf{ID}'_{\mathrm{ch}}$ is the challenge identifier relayed to $\mathcal{B}_I$'s challenger and C^* is the challenge ciphertext, these abort conditions ensure that $\mathcal{B}_I$ is a well-behaved CL-PKE Type I adversary whenever it does *not* abort.

Guess: Eventually, $\mathcal{A}_1$ should make a guess b' for b. Then $\mathcal{B}_I$ outputs b' as its guess for b.

Analysis: We now analyze the behaviour of $\mathcal{B}_I$ and $\mathcal{A}_1$ in this simulation. We claim that if algorithm $\mathcal{B}_I$ does not abort during the simulation then algorithm $\mathcal{A}_1$'s view is identical to its view in the real attack. Moreover, if $\mathcal{B}_I$ does not abort then $2|\Pr[b = b'] - \frac{1}{2}| = \epsilon$. We justify this claim as follows. Adversary $\mathcal{B}_I$'s responses to decryption and certification queries are as seen by $\mathcal{A}_1$ in a real attack, provided of course that $\mathcal{B}_I$ does not abort. Furthermore, the challenge ciphertext C^* is a valid Π^{CBE} encryption of M_b where $b \in \{0, 1\}$ is random. Thus, by definition of algorithm $\mathcal{A}_1$ we have that $2|\Pr[b = b'] - \frac{1}{2}| = \epsilon$.

The probability that $\mathcal{B}_I$ does not abort during the simulation remains to be calculated. $\mathcal{B}_I$ can abort for two reasons. The first reason is that $\mathcal{B}_I$ may be forced to replace the public key for $\mathsf{ID}'_{\mathrm{ch}}$ before the challenge phase and make a partial-private-key extract query on $\mathsf{ID}'_{\mathrm{ch}}$ in some phase. This combination of replace public key query and partial-private-key extract query can only arise from $\mathcal{A}_1$ making a $\mathsf{Certify}$ query on an input $\langle \tau_{\mathrm{ch}}, \lambda_{\mathrm{ch}}, PK_{\mathrm{ch}}, SK_{\mathrm{ch}}\rangle$. But this is exactly the certification query which $\mathcal{A}_1$ is forbidden from making. So this event never occurs in $\mathcal{B}_I$'s simulation. The second reason is that $\mathcal{B}_I$ may be forced to relay a decryption query on ciphertext C^* for identifier $\mathsf{ID}'_{\mathrm{ch}}$ and public key PK_{ch} in Phase 2. Because of the way that $\mathcal{B}_I$ relays ciphertexts, this event happens only if $\mathcal{A}_1$ makes a decryption query on input $\langle \tau_{\mathrm{ch}}, \lambda_{\mathrm{ch}}, PK_{\mathrm{ch}}, SK_{\mathrm{ch}}, C^*\rangle$ after having received its challenge ciphertext. However, $\mathcal{A}_1$ is forbidden from making precisely this decryption query. So this event never occurs in $\mathcal{B}_I$'s simulation.

To summarize, Algorithm $\mathcal{B}_I$ never aborts, provides a perfect simulation of $\mathcal{A}_1$'s challenger and has an advantage ϵ in guessing bit b. Thus we have shown that a Game 1 CBE adversary against Π^{CBE} with advantage ϵ can be used to construct a CL-PKE Type I adversary against Π^{CL} with an identical advantage. Since Π^{CL} is secure against CL-PKE Type I adversaries, we can deduce the Π^{CBE} is secure against CBE Game 1 adversaries.

Using similar ideas, we can also show that a CBE Game 2 adversary against Π^{CBE} can be used to construct a CL-PKE Type II adversary against Π^{CL}. Since Π^{CL} is secure against CL-PKE Type II adversaries, we can deduce that Π^{CBE} is secure against CBE Game 2 adversaries. This completes the proof.