Certificate-Based Encryption Without Random Oracles

Paz Morillo and Carla Ràfols
Universitat Politècnica de Catalunya
C/Jordi Girona, 1-3 08034 Barcelona
{paz,crafols}@ma4.upc.edu

Abstract

We present a certificate-based encryption scheme which is fully secure in the standard model. Our scheme is based on the identity-based encryption scheme of Waters [16]. Although some generic constructions from IBE to CBE has been previously proposed, they use the Random Oracle heuristic or provide less practical schemes than ours. Finally, we point out that one of the existing generic constructions going from IBE to CBE is flawed.

Keywords: identity-based encryption, certificate-based encryption, selective-ID security, full IBE security.

1 Introduction

In traditional public key cryptography (PKC) the authenticity of the public keys must be certified by a trusted third party, named as the Certification Authority (CA). The infrastructure required to support traditional PKC is the main difficulty in its deployment. Many of the problems of public key infrastructure (PKI) come from the management of certificates, which include storage, revocation and distribution.

In 1984, Shamir proposed the concept of Identity-Based Encryption (IBE), which sought to reduce the requirements on the infrastructure by using a well-known aspect of the client's identity as its public key. With this approach, certification becomes implicit, that is, the sender of a message does not need to check whether the client is certified or not. Instead, prior to decryption, the receiver must identify himself to a trusted authority, who will send him his private key. The first practical provably secure IBE scheme was proposed by Boneh and Franklin in 2001. It made use of bilinear maps on elliptic curves and it was proven secure in the random oracle model.

A different approach to the problem is the concept of Certificate-Based Encryption, proposed by Gentry in 2003 ([13]). In this model, certificates are part of the secret key, so certification is also implicit. Further, it has two important advantages over IBE: first, there is no key escrow, because certificates are only a part of the secret key, while the other is owned by the user alone (thus eliminating also the the need for a secure channel between the users and the trusted authority). Secondly, the approach to user revocation is neat and simple: certificates have an expiry date at the end of which a new one must be obtained from the CA, which only means that certification is implicit for every period.

The original scheme of Gentry relied heavily on the original IBE scheme of Boneh and Franklin and then on the Fujisaki Okamoto transform to obtain full security in the random oracle model. Recently [16] presented a new identity-based scheme which is secure against chosen-plaintext attacks in the standard model, improving significantly on previous results [8]. It is natural to try building a CBE scheme without random oracles.

Previous results ([11, 17]) for constructing a certificate-based encryption scheme in a generic way from an identity-based scheme exist, but are not comparable in efficiency to our scheme.

1.1 Our results

We present a certificate-based encryption scheme which is fully secure in the standard model and which is much more efficient than any of the previous schemes in the standard model (coming from the generic constructions of [11],[17]). Further we point out a security flaw in the proof of [17].

The proof is divided in three steps. The first two show how to construct a new encryption scheme called ExtendedCBE from the scheme of IBE scheme of Waters. This model satisfies the minimal properties which are necessary to adapt the proof of [9] to obtain a fully secure CBE scheme. Further we point out that whenever a scheme satisfying these minimal properties exist, a fully secure CBE also exists, that is, the last step of our proof can be generalized.

1.2 Organization

In section 2, we focus on the concept of certificate-based encryption and we give an overview of the existing generic constructions. In section 3, we sketch the security proof of our scheme and give a brief account of the results that we are going to use. In section 3 we give the necessary formal definitions. In section 4, 5 and 6 we build our scheme and conclude that the last step of the proof can be generalized.

2 Certificate-based Encryption

In this section we give the formal definitions for CBE, as well as an overview of the generic constructions of [17, 11].

As we noted in the introduction, the interest of certificate-based encryption compared to its predecessor, identity-based encryption, is that it overcomes two of its principal drawbacks, the inherence of the key escrow and the impossibility of revoking the users. Accordingly, the security model considers two types of adversaries, an uncertified client and a dishonest certifier.

The attack of an uncertified client models a client who is not certified for a given period but tries to obtain some information about the encrypted messages for that period. The client may have been certified *before* that period or may be certified *after* that period, so in such an attack, the adversary is allowed to make certification queries and choose the challenge period adaptively. Further, the client is also allowed to choose his pair public key -secret key adaptively and to make decryption queries for any period, including the challenge one.

The certifiers attack models a dishonest CA who wants to decrypt messages intended for a specific client Bob. The CA is allowed to make decryption queries for Bob's messages for any period of its choice.

The attack of the certifier was weakened by Al-Riyami and Paterson, since the original definition of Gentry was inconsistent with the concrete scheme he presented and it also made some assumptions about the underlying IBE scheme which were unnecessarily restrictive. In the original definition, for instance, the certifier could choose a part of its parameters adaptively, but not all IBE schemes allow that. As Al-Riyami and Paterson argue, it is hard to think of a scenario where this security requirement is necessary and the weakened version suffices to model any realistic attack.

2.1 Definitions

A certificate-based encryption scheme is a tuple of six algorithms ($Setup$, $SetKeyPair$, $Certify$, $Consolidate$, Enc, Dec), where:

> **-Setup**$_{CBE}$ 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.
> **-SetKeyPair** is a probabilistic algorithm that takes $params$ as input. It returns a pair public key - private key (PK, SK).
> **-Certify** is a (possibly randomized) algorithm that takes as input $\langle SK_{CA}$, $params$, i, $userinfo\ PK\rangle$. It returns $Cert'_i$, which is sent to the client. Here i identifies i-th time period, while $userinfo \in \Lambda$ contains other information needed to certify the client such as the client's identifying information, and PK is a public key.
> **-Consolidate** is a (possibly randomized) certificate consolidation algorithm taking as input $\langle params, i,\ userinfo, Cert'_i\rangle$ and optionally $Cert_{i-1}$. It returns $Cert_i$.

-Enc is a probabilistic algorithm taking as inputs $\langle params, M, i, userinfo, PK\rangle$ where $M \in \mathcal{M}$ is a message. It returns a ciphertext $C \in \mathcal{C}$ for message M or $\perp$ if PK is not a valid public key.
-Dec is a deterministic algorithm taking as inputs $\langle params, Cert_i, SK, C\rangle$ as input in time period i. 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 $\langle params, M, i, userinfo, PK\rangle$ and (PK, SK) is a valid key-pair, then M is the result of applying algorithm Dec on input $\langle params, Cert_i, SK, C\rangle$, where $Cert_i$ is the output of the $Certify$ and $Consolidate$ algorithms on input $\langle SK_{CA}, params, i, userinfo, PK\rangle$. We note that a concrete CBE scheme need not involve certificate consolidation. In this situation, algorithm $Consolidate$ will simply output $Cert_i = Cert'_i$.

The security model for CBE is defined with the help of two games:

CBE Game 1. Attack of an uncertified client
Setup The challenger runs $Setup$, gives $params$ to the adversary $\mathcal{A}_I$ and keeps SK_{CA} to itself.
Phase 1 The adversary issues queries $q_1, \ldots, q_m$ where each q_j is one of:
a) Certification query $\langle i, userinfo, PK, SK\rangle$. To answer this query, the challenger checks that $userinfo \in \Lambda$ and that $\langle PK, SK\rangle$ is a valid key-pair. If so, it runs $Certify$ on input $\langle SK_{CA}, params, i, userinfo, PK\rangle$ and returns $Cert'_i$; else it returns $\perp$.
b) Decryption query $\langle i, userinfo, PK, SK, C\rangle$, the challenger checks that $\langle PK, SK\rangle$ is a valid key-pair. If so, it generates $Cert_i$ by using algorithms $Certify$ and $Consolidate$ with inputs $\langle SK_{CA}, params, i, userinfo, PK\rangle$ and outputs $Dec_{Cert_i, SK}(C)$, else it returns $\perp$.
These queries may be asked adaptively, that is, they may depend on the answers to previous queries.
Challenge On challenge query $\langle i^*, userinfo^*, PK^*, SK^*, M_0, M_1\rangle$, where $M_0, M_1 \in \mathcal{M}$ are of equal length, the challenger checks that $userinfo^* \in \Lambda$ and that $\langle PK^*, SK^*\rangle$ is a valid key pair. If so, it chooses a random bit b and returns $C^* = Enc_{i^*, userinfo^*, PK^*}(M_b)$; else it returns $\perp$.
Phase 2 As in phase 1, except that decryption queries $\langle i^*, userinfo^*, PK^*, SK^*, C^*\rangle$ are disallowed.
Guess The adversary $\mathcal{A}_I$ outputs a guess $b' \in \{0, 1\}$.
The adversary wins the game if $b = b'$. We define the advantage of $\mathcal{A}_I$ as $Adv_{\mathcal{A}_I} := |Pr[b = b'] - \frac{1}{2}|$.

CBE Game 2. Attack of the certifier
Setup The challenger runs $Setup$, gives $params$ and SK_{CA} to the adversary $\mathcal{A}_{II}$. The challenger then runs $SetKeyPair$ to obtain a key-pair $\langle PK, SK\rangle$ and gives PK to the adversary $\mathcal{A}_{II}$
Phase 1 The adversary issues decryption queries $q_1, \ldots, q_m$ where each q_j is a decryption query $\langle i, userinfo, PK, C\rangle$. On this query, the challenger generates $Cert_i$ by using algorithms $Certify$ and $Consolidate$ with inputs $\langle SK_{CA}, params, i, userinfo, PK\rangle$ and outputs $Dec_{Cert_i, SK}(C)$, else it returns $\perp$.
These queries may be asked adaptively, that is, they may depend on the answers to previous queries.
Challenge On challenge query $\langle i^*, userinfo^*, M_0, M_1\rangle$, where $M_0, M_1 \in \mathcal{M}$ are of equal length, the challenger checks that $userinfo^* \in \Lambda$. If so, it chooses a random bit b and returns $C^* = Enc_{i^*, userinfo^*, PK}(M_b)$; else it returns $\perp$.
Phase 2 As in phase 1.
Guess The adversary $\mathcal{A}_{II}$ outputs a guess $b' \in \{0, 1\}$.
The adversary wins the game if $b = b'$. We define the advantage of $\mathcal{A}_{II}$ as $Adv_{\mathcal{A}_{II}} := |Pr[b = b'] - \frac{1}{2}|$.

Definition A CBE scheme is said to be secure against an adaptive chosen ciphertext attack (or IND-CBE-CCA secure) if no probabilistic polynomially bounded adversary has non-negligible advantage in security parameter k in either CBE Game 1 or CBE Game 2.

2.2 Generic constructions

It is clear that the notion of IBE and CBE are very closely related and in fact all the generic constructions that have been proposed so far start from an IBE scheme IND-ID-CCA secure. The construction of [2] which goes from certificateless public key cryptography to certificate-based public key cryptography, was not fully correct according to [14].
The first remark that one ought to make is that these constructions suffer from the same drawback than ours, namely, that the most efficient IBE scheme secure in the standard model is based on the scheme of Waters [16], in a way that we will detail later. The resulting scheme IND-ID-CCA secure has several problems, mainly that the reduction is far from tight and the parameters are too long (these problems come from the scheme of Waters).
Our construction is much more efficient than these generic ones, since while the scheme we propose does only add one pairing, two exponentiations, a MAC and an encapsulation to the original encryption process of the resulting IBE scheme, in the existing generic constructions it is used in combination with a public key encryption scheme [11] or it is even used twice for double encryption [17].

The proposal of Dodis and Katz: In [11], Dodis and Katz study the security of double encryption. They point out that double encryption with two different public key scheme (cascade encryption, as they sometimes call it), $E_{pk_1}(E_{pk_2}(M))$ does not necessarily yield full security, even if the two public key schemes are IND-CCA secure. We are going to use some remarks of this paper to criticize the proof of Yum and Lee below.
They also give a generic construction for CBE. The certifier generates the parameters for an identity-based encryption scheme IND-ID-CCA and the user chooses a pair public key- secret key (SK, PK) for a public key encryption scheme IND-ID-CCA secure. To encrypt a message, the sender generates a pair secret key - verifier's key (sk, vk) of a one-time signature scheme. The message for i, Bob is then divided into two shares $M_1 \oplus M_2$. M_1 is encrypted using the public key of the user Bob and label vk and M_2 is encrypted in the identity-based scheme with respect to identity $(Bobinfo||periodi||PK)$ and label vk. The two resulting ciphertexts are then signed using a one-time signature $\sigma = Sig_{sk}(C_1, C_2)$ to obtain full security, the so-called IND-CBE-CCA security.

The proposal of Yum and Lee At EuroPKI 2004, Yum and Lee proved the equivalence between identity-based and certificate-based encryption, that is, whenever a fully secure IBE exists (that is IND-ID-CCA), a fully secure IND-CBE-CCA exists, and conversely, the existence of a CBE scheme IND-CBE-CCA implies the existence of an IND-ID-CCA secure IBE scheme.
Briefly, their construction is as follows. They generate the parameters for two different instantiations of the IBE scheme, which yield two pairs, $(params_{CA},\ msk_{CA})$ and $(params_{user}, msk_{user})$. Then msk_{CA} will serve as a the certifier's master secret key in the CBE scheme and the user secret and public key (PK, SK) will be the public key and the secret key corresponding to identity $userinfo$ in the second instantiation of the IBE. Encryption is done by running twice the IBE encryption algorithm ID_{Enc}, first with inputs $\langle M, userinfo, params_{user}\rangle$ and output C', then with input $\langle C', (userinfo,\ i, PK),\ params_{CA}\rangle$.
We note that this construction does not achieve the required security for certificate-based schemes, at least in the case of an attack of the certifier. We outline how would an attack form a certifier work. Remember that the certifier is equipped with his own secret key and that it is allowed to make decryption queries, with the natural limitation that he cannot ask for the challenge ciphertext. The attack begins once the certifier obtains the challenge ciphertext C^* for $userinfo^*, i^*, PK^*$.
1. The certifier generates the certificate for $userinfo^*, periodi^*, PK^*$.
2. This certificate is used to decrypt and obtain $C' = ID_{Enc}(M_b, userinfo^*, params_{user})$.
3. Reencrypt and set $C'' = ID_{Enc}(C', (userinfo^*,\ i^*, PK),\ params_{CA})\rangle$.
4. Ask the decryption oracle for the decryption of C''.

3 Our construction

3.1 A powerful tool for obtaining full security in the standard model

In 2004 [10], Canetti, Halevi and Katz introduced a generic construction in the standard model from any IBE scheme IND-sID-CPA secure to a public key encryption scheme.
Briefly, their idea was to take the public key and private key of the user to be the parameters and the master key of of an IBE scheme, respectively. To encrypt a message, the sender must generate a pair (sk, vk) of a one-time signature scheme, encrypt with respect to identity vk and send $\langle C = E_{vk}(M), vk, \sigma = Sig_{sk}(C)\rangle$.
Informally, this works because decryption queries in the PKE scheme become extraction queries in the IBE scheme. Namely, if there is an adversary $\mathcal{B}$ against the PKE scheme, then, it can be used to build an adversary $\mathcal{A}$ against the IBE scheme. The decryption queries of $\mathcal{B}$, $\langle C, vk, \sigma\rangle$, are answered by $\mathcal{A}$ by asking the challenger for the secret key corresponding to identity vk. The only difference between the real game and the simulated game occurs if $\mathcal{B}$ asks for the decryption of a ciphertext with vk^*, where vk^* is the verifier's key of a one-time signature scheme that $\mathcal{A}$ has chosen as challenge identity in the initialization step. But this would only occur with negligible probability before the challenge phase, and also after, because we assume the one-time signature scheme to be secure in the sense of strong unforgeability.
Boneh and Katz improved this construction and made it much more efficient, specially improving on key generation. Their idea was to use message authentication codes instead of signatures. The key for the MAC cannot then be the identity, though, because the identity must go on the open. The solution is to use also a commitment. In the resulting scheme, then a random value r is encapsulated to obtain (r, com, dec) and then the message $M||dec$ is encrypted with respect to com. The proof is somewhat trickier because only the receiver can make the verification, but the main idea behind it is the same as in [10]. In our construction we will use this technique [9].
Further the technique of [10] can also be extended to go from a l-HIBE which is selective identity chosen plaintext secure to an $(l-1)$-HIBE which is selective-identity chosen ciphertext secure (IND-sID-CCA, see for example [7]). In particular this means it is possible to construct a IBE scheme IND-ID-CCA secure from a 2-HIBE scheme IND-sID-CPA secure.

3.2 The scheme of Waters

The first IBE fully secure in the standard model was proposed by [8] and has been recently improved by Waters [16]. The scheme of Waters is only IND-ID-CPA secure, but if extended to a 2-HIBE scheme the afore mentioned construction of Boneh and Katz would apply, obtaining then an IBE scheme fully secure in the standard model.
However, since the construction of Boneh and Katz only requires selective-identity chosen plaintext security and the scheme of Waters has a security reduction which is not tight, it is more convenient to extend the level in the second level using the scheme of Boneh and Boyen [7] which is selective-identity chosen-plaintext secure, an idea which Waters himself suggests in [16].
This yields a 2-HIBE scheme satisfying a very unusual definition of security, namely, where the suffix of the challenge identity (the second level) must be chosen before the beginning of the attack but the prefix (the first level) is chosen at the challenge step.

3.3 The construction of Gentry

As we said, the construction of a CBE scheme of Gentry relies very much on the IBE scheme of Boneh and Franklin. As an intermediate step in their construction, they build a scheme called BasicIBE, and Gentry introduces a scheme called BasicCBE. Without going into details, the only difference between both schemes is that Boneh and Franklin use a BLS [3] signature as a decryption key and Gentry uses an aggregate BGLS

[5] signature.
The scheme of Gentry is then constructed applying the Fujisaki Okamoto transform to BasicCBE, and Boneh and Franklin also obtain the full scheme in this way.
It is reasonable to do the same thing with respect to the scheme of Waters. Thus, it is possible to obtain a CBE scheme which is IND-CBE-CPA secure in a straightforward way. The problem is now to obtain CCA security in the standard model.

3.4 Proof's strategy

A first approach would be to try to follow the suggestion of Waters and build an hybrid 2-HIBE using the schemes of Waters and Boneh-Boyen. From this scheme apply the result of [9] to obtain a fully secure IBE and then construct a CBE scheme by using an aggregate BGLS signature. If this proof worked, then we would have proven the full security of our scheme without having to resort to any unusual cryptographic primitives. However, when building the scheme in this way we only managed to prove a weaker notion of security.
The strategy we follow instead is: we build the same hybrid HIBE as we specified in section 3.2 and then build another scheme called ExtendedCBE by using a BGLS signature instead of a BLS one. Then we adapt the proof of Boneh and Katz to obtain full security.

4 Review on Pairings

Bilinear Diffie-Hellman Parameter Generator A randomized algorithm $\mathcal{IG}$ is a BDH parameter generator if it takes as input security parameter $k \geq 0$, runs in time polynomial in k and returns the description of two groups $\mathbb{G}$, $\mathbb{G}_1$ of the same prime order p together with the description of an admissible pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}$. Formally, the output of $\mathcal{IG}(1^k)$ is $\langle \mathbb{G}, \mathbb{G}_1, e \rangle$.
Admissible pairings are maps $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$ with the following properties:
1. Bilinear: $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$
2. Non-degenerate: $e(g, g) \neq 1_{\mathbb{G}_1}$ for all $g \in \mathbb{G}$
3. Computable: there exists an efficient algorithm to compute $e(g_1, g_2)$ for any $g_1, g_2 \in \mathbb{G}$.

The **BDH problem** in $\mathbb{G}$ is as follows: given a tuple $g, g^a, g^b, g^c \in \mathbb{G}$ as input, output $e(g, g)^{abc}$.
An algorithm $\mathcal{A}$ has advantage ε in solving the BDH problem in $\mathbb{G}$ if:
$Pr[\mathcal{A}(g, g^a, g^b, g^c) = e(g, g)^{abc}] \geq \varepsilon$,
where the probability is over the random choice of generator g in $\mathbb{G}^*$, the random choice of a, b, c in $\mathbb{Z}_p^*$, and the random bits used by $\mathcal{A}$.
Similarly we say that an algorithm $\mathcal{B}$ that outputs $b \in \{0, 1\}$ has advantage ε in solving the **decisional BDH** problem in $\mathbb{G}$ if: $|Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g, g)^{abc}) = 0] - Pr[\mathcal{B}(g, g^a, g^b, g^c, T) = 0]| \geq \varepsilon$,
where the probability is over the random choice of generator $g \in \mathbb{G}^*$, the random choice of $T \in \mathbb{G}_1$, and the random bits consumed by $\mathcal{B}$.
Definition The (Decisional) (t, ε)-Bilinear Diffie Hellman (BDH) assumption holds in $\mathbb{G}$ if no t-time algorithm has advantage at least ε in solving the (Decisional) BDH problem in $\mathbb{G}$.

5 Security definitions

The building blocks for our scheme will be the identity-based scheme of Waters [16] only IND-ID-CPA secure, the identity-based scheme of Boneh and Boyen [7] (only IND-sID-CPA secure) and the technique of [9] (which makes use of a message authentication code and a encapsulation scheme). For the proof we will need to define some very unusual primitives and their security model, which we hope to motivate in the next section. Here only the definitions are introduced.

The security definitions for a message authentication code and an encapsulation mechanism are basically quoted from [9].

5.1 Message Authentication

Definition A *message authentication code* is a pair of PPT algorithms $(Mac, Vrfy)$, where:
1. Mac is an algorithm which takes as input a message M and a secret key sk and outputs a string tag.
2. $Vrfy$ takes as input a message M, a secret key sk, and a string tag. It outputs either 1 or 0, in case it succeeds or not.
The security requirement we will need for our construction is the same as in [9], that is, one-time security. More formally,

Definition A message authentication code $(Mac, Vrfy)$ is secure against a *one-time chosen-message attack* if the success probability of any PPT adversary $\mathcal{A}$ in the following game is negligible in the security parameter k:
1. A random key $sk \in \{0,1\}^k$ is chosen.
2. $\mathcal{A}$ outputs a message M and is given in return $tag = Mac_{sk}(M)$.
3. $\mathcal{A}$ outputs a pair (M', tag').
We say that $\mathcal{A}$ succeeds if $(M, tag) \neq (M', tag')$ and $Vrfy_{sk}(M', tag') = 1$
In the above, the adversary succeeds even if $M = M'$ but $tag \neq tag'$.

5.2 Encapsulation

Definition An encapsulation scheme is a triple of PPT algorithms $(Setup_{ENC}, \mathcal{S}, \mathcal{R})$ such that:
1. $Setup_{ENC}$ takes as input the security parameter 1^k and outputs a string pub.
2. $\mathcal{S}$ takes as input 1^k and pub, and outputs (r, com, dec) with $r \in \{0,1\}^k$. We refer to com as the public commitment string and dec as the de-commitment scheme string.
3. $\mathcal{R}$ takes as input (pub, com, dec) an outputs $r \in \{0,1\}^k \cup \{\perp\}$.

Definition An encapsulation scheme $(Setup, \mathcal{S}, \mathcal{R})$ is secure if it satisfies both hiding and binding as follows:
Hiding The following is negligible in the security parameter k for all PPT $\mathcal{A}$
$|Pr[(pub \leftarrow Setup_{ENC}(1^k);\ r_0 \leftarrow \{0,1\}^k; (r_1, com, dec) \leftarrow \mathcal{S}(1^k, pub);\ b \in \{0,1\}) : \mathcal{A}(1^k, pub, com, r_b) = b] - \frac{1}{2}|$
Binding The following is negligible in the security parameter k for all PPT $\mathcal{A}$
$|Pr[(pub \leftarrow Setup_{ENC}(1^k);\ (r, com, dec) \leftarrow \mathcal{S}(1^k, pub)) : dec' \leftarrow \mathcal{A}(1^k, pub, r, com, dec);\ \mathcal{R}(pub, com, dec') \notin \{\perp, r\}]$

5.3 HIBE

A m-HIBE consists of four algorithms: $Setup_{HIBE}, KeyGen, Enc, Dec$, where m is polynomial in the security parameter k:

> **-Setup**$_{HIBE}$ is a probabilistic algorithm taking as input a security parameter k. It returns msk (the Public Key Generator's master secret key) and the public parameters $params$.
> **-KeyGen** is a possibly randomized algorithm that takes as input an identity $ID = (I1, \ldots, Ij)$ $(j \leq m)$ and outputs the secret key corresponding to ID, d_{ID}.
> **-Enc** is a probabilistic algorithm that takes as input $\langle params,\ M,\ ID, \rangle$. It returns a ciphertext $C = Enc_{params,ID}(M)$.
> **-Dec** is a deterministic algorithm taking as input $\langle params, C, ID,\ d_{ID} \rangle$. It returns a plaintext M.

Naturally, we require that if C is the result of running algorithm Enc with input $\langle params, M, ID\rangle$, then M is the result of applying algorithm Dec with input $\langle params, C, ID, d_{ID}\rangle$.

A new definition of security for a 2-HIBE

A 2-HIBE is secure against *2nd-level selective identity chosen plaintext attacks* if for all polynomially bounded functions $l()$ the advantage of any PPT adversary $\mathcal{A}$ in the following game is negligible in the security parameter k:

Init $\mathcal{A}$ outputs a sufix $I2^* \in \{0,1\}^{l(k)}$ of the identity it wants to attack. (That is, the challenge identity will be of the form $(I1, I2^*)$).
Setup $Setup_{HIBE}(1^k, l(k))$ outputs $(msk, params)$. The adversary is given $params$.
Phase 1 The adversary issues private key or extraction queries $q_1, \ldots, q_m$ for identities $\langle ID_i\rangle$, $i = 1 \ldots m$, which can be either in level one $ID_i = I1$ or level two $ID_i = (I1, I2)$, with $I2 \neq I2^*$. The challenger responds by running algorithm $KeyGen$ to generate the private key d_{ID_i} corresponding to the public key $\langle ID_i\rangle$. Then d_{ID_i} is sent to the adversary. These queries may be asked adaptively, that is, depending on the answers to preceding queries.
Challenge When the adversary decides that phase 1 is over it outputs two messages M_0 and M_1 and a first level identity $I1^*$ on which it wants to be challenged. This identity should not have been the subject of a private key query in phase 1. The challenger flips a fair coin to obtain a bit b and sets the challenge ciphertext to be $C = Enc_{params,(I1^*,I2^*)}(M_b)$.
Phase 2 As in phase 1, except with the additional restriction that $\mathcal{A}$ may not ask for the secret key corresponding to identity $I1^*$ (but may ask for the secret key corresponding to identities $(I1^*, I2)$, where $I2 \neq I2^*$).
Guess The adversary outputs a guess $b' \in \{0,1\}$. The adversary wins the game if $b = b'$.
We define the advantage of the adversary $\mathcal{A}$ in this game as: $Adv_{\mathcal{A}} = |Pr[b = b'] - \frac{1}{2}|$.

5.4 Extended CBE

The concept that we are going to define next is very unusual in identity-based cryptography, but it is motivated by the requirements of the security proof.
Our model is a depth two tree encryption scheme, where a given message M can be encrypted for a 1st level entity or a 2nd level entity, and where 1st level entities can decrypt any of the messages intended for a 2nd level entity.
In this model there is also a certification authority (CA) and a number of clients, each of whom chooses a pair public key, secret key (PK, SK). Each client has also an identifying public information *userinfo*. The time is divided into different periods (the number of which does not necessarily have to be specified beforehand). For each time period the certification authority computes a certificate $Cert_i'$, from its own master secret key SK_{CA} and $\langle userinfo, i, PK\rangle$ and sends it to the authorized clients, who may perform some operations on the certificate to obtain $Cert_i$.
In our scheme, then, the entities in the 1st level are certified clients, that is, messages are encrypted for a certain period, a certain public information identifying the client and a certain public key. To decrypt such a message, both the secret key of the client and the updated certificate $Cert_i$ are needed. The entities in the first level are noted $(userinfo, period\ i, PK)$.
The 2nd level entities will also be called sons of certified clients and will be noted $((userinfo, i, PK), I2)$. When a message is intended for a second level entity, the key necessary to decrypt is derived from both $Cert_i$ and SK. However, this same key will not be useful to decrypt any message for any of its siblings, i.e entities $((userinfo, i, PK), I2')$, where $Cert_i$ is the certificate corresponding to $(userinfo, i, PK)$ and $I2 \neq I2'$.
The keys for the sons of the certified clients are computed by the clients and sent to their sons.
For the security model, two types of adversary are considered. Again, these types respond to the needs of the last proof, and it is hard to motivate them otherwise. Type I adversary is a client who can adaptively choose

its private/ public key pair, its identifying public information and make certification queries for any period and extraction queries for any second level entity (with a suffix different than the second level challenge identity).
Type II adversary has access to the certifier's master secret key and can also make extraction queries for any entity in level 2.
In both types of attack, the entity attacked must be in the second level, since this is the case that will be used in the last proof.

Definition An extended CBE scheme consists of seven algorithms: ($Setup_{EXTCBE}$, $SetKeyPair$, $Certify$, $Consolidate, KeyGen2$, Enc, Dec), where:

> **-Setup**$_{EXTCBE}$ 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.
> **-SetKeyPair** is a probabilistic algorithm that takes as input *params*. It returns a public key PK and a private key SK.
> **-Certify** is a (possibly randomized) algorithm that takes as input $\langle SK_{CA}$, $params$, i, $userinfo$, $PK\rangle$. It returns $Cert'_i$, which is sent to the client. Here i is a string identifying a time period, while $userinfo \in \Lambda$ contains other information needed to certify the client such as the client's identifying information, and PK is a public key.
> **-Consolidate** is a (possibly randomized) certificate consolidation algorithm taking as input $\langle params, i, userinfo, Cert'_i\rangle$ and optionally $Cert_{i-1}$.
> **-KeyGen2** is a (possibly randomized) algorithm that takes as input $params$, a pair (PK, SK), a period i, a string $userinfo \in \Lambda$, the updated certificate $Cert'_i$ corresponding to this input and a second level identity $I2$. It then generates the secret key SK_{ID} necessary corresponding to second level entity to decrypt all ciphertexts intended for identity $((i, userinfo, PK), I2)$.
> **-Enc** is a probabilistic algorithm taking as input $\langle ID, M\rangle$, where ID is the string identifying either a certified client or a son of a certified client and $M \in \mathcal{M}$ is a message. It returns a ciphertext $C \in \mathcal{C}$ for message M.
> **-Dec** is a deterministic algorithm taking as inputs $\langle params, ID, SK_{ID}, C\rangle$ as input in time period i, where ID is a string corresponding either to a first or a second level entity. If ID identifies a first level entity then SK_{ID} is the pair $(Cert'_{periodi}, SK)$, else it is the output of algorithm $KeyGen2$ with these inputs. Algorithm Dec 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 $\langle i, userinfo, params, PK, M\rangle$ and (PK, SK) is a valid key-pair, then M is the result of applying algorithm Dec on input $\langle params, Cert_i, SK, C\rangle$, where $Cert_i$ is the output of the $Certify$ and $Consolidate$ algorithms on input $\langle SK_{CA}, params, i, userinfo \in \Lambda, PK\rangle$.
> We note that a concrete ExtendedCBE scheme need not involve certificate consolidation. In this situation, algorithm $Consolidate$ will simply output $Cert_i = Cert'_i$.

Security for Extended CBE is defined with the help of two different games.

Extended CBE Game 1

> **Init** The adversary $\mathcal{B}_I$ outputs a second level identity $I2^* \in \{0,1\}^{l(k)}$ it wants to attack.
> **Setup**: The challenger runs $Setup_{EXTCBE}(1^k, l(k))$, gives $params$ to the adversary and keeps SK_{CA} to itself.
> **Phase 1** The adversary issues queries $q_1, \ldots, q_m$ where each q_j is:
> a) a certification query $\langle i, userinfo, PK, SK\rangle$. To answer this query, the challenger checks that $userinfo \in \Lambda$ and that $\langle PK, SK\rangle$ is a valid key-pair and runs algorithm $Certify$ on these inputs. The output $Cert'_i$ is the answer to the query.

b) an extraction query $\langle ID, SK\rangle$, where $ID = ((i, userinfo, PK)), I2)$ is a second level identity. To answer this query, the challenger checks that $\langle PK, SK\rangle$ is a valid key-pair. Then it runs algorithms $Certify$, $Consolidate$ and $KeyGen2$ with the adequate inputs.
These queries may be asked adaptively, that is, they may depend on the answers to previous queries.
Challenge On challenge query $\langle i^*, userinfo^*, PK^*, SK^*, M_0, M_1\rangle$, where $M_0, M_1 \in \mathcal{M}$ are of equal length, the challenger checks that $userinfo^* \in \Lambda$ and that $\langle PK^*, SK^*\rangle$ is a valid key pair. If so, it chooses a random bit b and returns $C^* = Enc_{ID^*}(M_b)$, where $ID^* = ((i^*, userinfo^*, PK^*), I2^*)$, else it returns $\perp$.
Phase 2 As in phase 1, except that certification queries $\langle i^*, userinfo^*, PK^*, SK^*\rangle$ are no longer allowed, but decryption queries for any identity $ID = ((i^*, userinfo^*, PK^*), I2)$, with $I2 \neq I2^*$ are.
Guess The adversary outputs a guess $b' \in \{0, 1\}$.

The adversary wins the game if $b = b'$. We define the advantage of $\mathcal{B}_I$ as $Adv_{\mathcal{B}_I} := |Pr[b = b'] - \frac{1}{2}|$.

Extended CBE Game 2

Init The adversary outputs a second level identity $I2^* \in \{0, 1\}^{l(k)}$ it wants to attack.
Setup: The challenger runs $Setup_{EXTCBE}(1^k, l(k))$ and gives $params$ and SK_{CA} to the adversary. Then it runs algorithm $SetKeyPair$ to obtain a challenge pair (PK, SK) and gives PK to the adversary.
Phase 1 The adversary issues queries $q_1, \ldots, q_m$ where each q_j is an extraction query $\langle((i, userinfo, PK)), I2)$ for a second level identity. To answer this query, the challenger checks that $userinfo \in \Lambda$. If so, it generates $Cert_i$ by using algorithms $Certify$ and $Consolidate$ with these inputs. Then it runs algorithm $KeyGen2$ with these inputs.
These queries may be asked adaptively, that is, they may depend on the answers to previous queries.
Challenge On challenge query $\langle i^*, userinfo^*, M_0, M_1\rangle$, where $M_0, M_1 \in \mathcal{M}$ are of equal length, the challenger checks that $userinfo^* \in \Lambda$. If so, it chooses a random bit b and returns $C^* = Enc_{ID^*}(M_b)$, where $ID^* = ((i^*, userinfo^*, PK), I2^*)$, else it returns $\perp$.
Phase 2 As in phase 1.
Guess The adversary outputs a guess $b' \in \{0, 1\}$.

The adversary wins the game if $b = b'$. We define the advantage of $\mathcal{B}_{II}$ as $Adv_{\mathcal{B}_{II}} := |Pr[b = b'] - \frac{1}{2}|$.

Definition An Extended CBE scheme is said to be secure against adaptive chosen plaintext attack (or IND-extCBE-CPA secure) if for all polynomially bounded functions $l()$, no probabilistic polynomially bounded adversary has non-negligible advantage in security parameter k in either CBE Game 1 or CBE Game 2.

6 First construction: an hybrid 2-HIBE scheme

In the rest of the article, given a string $\lambda = \lambda_1 \ldots \lambda_n \in \{0, 1\}^n$, let $\nu_\lambda \subset \{1 \ldots n\}$ be the set of indices j for which $\lambda_j = 1$.
Let identities in the first level be n-bit strings and identities in the second level be elements of $\mathbb{Z}_p$. We will refer to identities in general as ID, and if we need to distinguish if the identity is in the first or second level we will write either $I1$ or $(I1, I2)$, respectively. The proposed scheme is the IBE scheme of Waters [16] when restricted to the first level and the scheme of Boneh and Boyen [7] in the second. Therefore, an adversary against our scheme has to specify at first which identity in the second level it is going to attack, that is,the suffix of the challenge identity. No identity with that suffix can be subject to an extraction query.

New2-HIBE

Setup$_{HIBE}$ Input: 1^k.
Run $\mathcal{IG}$ on input 1^k and obtain $\langle \mathbb{G}, \mathbb{G}_1, e\rangle$, $\mathbb{G}, \mathbb{G}_1$ of order p.
Choose $g, g_2, f_2 \leftarrow \mathbb{G}^*$, $\alpha \leftarrow \mathbb{Z}_p$. Set $g_1 = g^\alpha \in \mathbb{G}$
Choose $u', u_1, \ldots, u_n \leftarrow \mathbb{G}$. Set $U = (u', u_1, ..., u_n)$.
The space of messages is $\mathbb{G}_1$ and the system parameters are $params = (U, p, n, \mathbb{G}, \mathbb{G}_1, e, g, g_1, g_2, f_2)$.
The PKG's master secret key is $msk = \alpha$. Define the following function $F_2 : \mathbb{Z}_p \longrightarrow \mathbb{G}$ as $F_2(x) = g_1^x f_2$.

KeyGen Input: $\langle params, msk, ID\rangle$.
To generate the private key corresponding to ID, d_{ID} do:
(a) if ID is in level 1, the PKG sets $r_1 \leftarrow \mathbb{Z}_p$ and sets: $d_{I1} = (d_0, d_1) = (g_2^\alpha(u^{'} \prod_{j\in\nu_{I1}} u_j)^{r_1}, g^{r_1})$.
(b) Else, the PKG chooses $r_1, r_2 \leftarrow \mathbb{Z}_p$ and sets: $d_{(I1,I2)} = (d_0, d_1, d_2) = (g_2^\alpha(u^{'} \prod_{j\in\nu_{I1}} u_j)^{r_1} F_2(I_2)^{r_2}, g^{r_1}, g^{r_2})$.
Obviously, any identity in level 1 $I1$ with secret key $d_{ID} = (d_0, d_1)$, can compute the secret key for all of its children by choosing $r_2 \leftarrow \mathbb{Z}_p$ and computing $d_{(I1,I2)} = (d_0 F_2(I2)^{r_2}, d_1, g^{r_2})$.

Enc Input: $\langle M, ID\rangle$.
Choose $t \leftarrow \mathbb{Z}_p$.
Set $C = (C_1, C_2, C_3) = (Me(g_1, g_2)^t, g^t, (u^{'} \prod_{j\in\nu_{I1}} u_j)^t)$ if user is in level 1, else set $C = (C_1, C_2, C_3, C_4) = (Me(g_1, g_2)^t, g^t, (u^{'} \prod_{j\in\nu_{I1}} u_j)^t, F_2(I2)^t)$.

Dec Input: $\langle C, ID\rangle$.
(a) If ID is in level 1, compute:
$$\frac{C_1 e(d_1, C_3)}{e(d_0, C_2)} = \frac{Me(g_1, g_2)^t e(g^{r_1}, (u' \prod_{j\in\nu_{I1}} u_j)^t)}{e(g_2^\alpha(u' \prod_{j\in\nu_{I1}} u_j)^{r_1}, g^t)} = \ldots = M$$
(b) Else, compute:
$$\frac{C_1 e(d_1, C_3) e(d_2, C_4)}{e(d_0, C_2)} = \frac{Me(g_1, g_2)^t e(g^{r_1}, (u' \prod_{j\in\nu_{I1}} u_j)^t) e(g^{r_2}, F_2(I2)^t)}{e(g_2^\alpha(u' \prod_{j\in\nu_{I1}} u_j)^{r_1} F_2(I2)^{r_2}, g^t)} = \ldots = M$$

6.1 Security Proof

For the security reduction we distinguish between first and second level extraction queries. The number of first level extraction queries is q_E and the number of extraction queries for the second level q_D.

Theorem The previously defined New 2-HIBE is (t, q_E, q_D, ϵ) 2nd-level selective identity secure if the $(t + O(\epsilon^{-2} ln(\epsilon^{-1})\lambda^{-1} ln(\lambda^{-1}) + q_D), \frac{\epsilon}{32(n+1)q_E})$ Decisional Bilinear Diffie Hellman Assumption holds in $\mathbb{G}$ (where $\lambda = \frac{1}{8(n+1)q_E}$ and it is assumed that each exponentiation in $\mathbb{G}$ takes unit time).
Proof
Let $\mathcal{C}$ be an adversary against the 2-HIBE hybrid scheme, then we are going to use $\mathcal{C}$ to build an adversary $\mathcal{D}$ against DBDH in $\mathbb{G}$.
$\mathcal{D}$ is given as input a 5-tuple (g, g^a, g^b, g^c, T), which could be either a random tuple or a BDH-tuple. Set $g_1 = g^a$, $g_2 = g^b$, $g_3 = g^c$.
Adversary $\mathcal{D}$ will output a guess γ' as to whether the challenge tuple is a BDH tuple or not, that is, whether γ is equal to 1 or 0, respectively. $\mathcal{D}$ interacts with $\mathcal{C}$ as follows:

Init Adversary $\mathcal{C}$ outputs the second level challenge identity $I2^* \in \mathbb{Z}_p$. That means that in the challenge, $\mathcal{C}$ may ask to be challenged on any identity of the form $ID = (I1, I2^*)$.

Setup Adversary $\mathcal{D}$ first sets $m = 4q_E$ and chooses an integer, k, between 0 and n. It then chooses a random n-length vector, $\overrightarrow{x} = (x_i)$, and a value x'. The components of the vector and x' are chosen u.a.r. among the integers between 0 and $m-1$. By X', we denote the pair $(x', \overrightarrow{x})$. Additionally, $\mathcal{D}$ also chooses $y', y_1, \dots y_n \in \mathbb{Z}_p$.
Finally, $\mathcal{D}$ also picks $\alpha_2 \leftarrow \mathbb{Z}_p$. These values are all kept internal to adversary $\mathcal{D}$.
Given a set $\nu \subset \{0, \dots n\}$, we define the following functions and values:

(a)$F(\nu) = (p - mk) + x' + \Sigma_{i \in \nu} x_i$
(b)$J(\nu) = y' + \Sigma_{i \in \nu} y_i$
(c)

$$K(\nu) = \begin{cases} 0 & \text{if } x' + \Sigma_{i \in \nu} x_i \equiv 0 \mod m \\ 1 & \text{otherwise} \end{cases}$$

(d)$f_2 = g_1^{-I2^*} g^{\alpha_2} \in \mathbb{G}$
(e) $F_2 : \mathbb{Z}_p \to \mathbb{G}$, defined as $F_2(x) = g_1^{x - I2^*} g^{\alpha_2} = g_1^x f_2$
(f) $U = (u', u_1, \dots u_n)$, where $u' = g_2^{p-km+x'} g^{y'}$ and $u_i = g_2^{x_i} g^{y_i}$ for $i = 1 \dots n$

Then $\mathcal{C}$ is given $params = (U, p, n, \mathbb{G}, \mathbb{G}_1, e, g, g_1, g_2, f_2)$.

Phase 1 $\mathcal{C}$ issues private key queries q_l for different identities ID_l, to which $\mathcal{D}$ responds in the following way:

a) If $ID_l = I1l$ is in level 1, $\mathcal{D}$ checks if $K(\nu_{I1l}) = 0$. If this is the case, it aborts and outputs a random guess γ'.
Else, it chooses $r_l \leftarrow \mathbb{Z}_p$ and sets $d_{I1l} = (d_{0l}, d_{1l}) = (g_1^{-\frac{J(\nu_{I1l})}{F(\nu_{I1l})}} (u' \prod_{j \in \nu_{I1l}} u_j)^{r_l}, g^{r_l})$.
Set $s_l := r_l - \frac{a}{F(\nu_{I1l})}$. Note that the following two equalities hold:

$$\begin{aligned} d_{0l} &= g_1^{-\frac{J(\nu_{I1l})}{F(\nu_{I1l})}} (u' \textstyle\prod_{j \in \nu_{I1l}} u_j)^{r_l} \\ &= g_1^{-\frac{J(\nu_{I1l})}{F(\nu_{I1l})}} (g_2^{F(\nu_{I1l})} g^{J(\nu_{I1l})})^{r_l} \\ &= g_2^a (g_2^{F(\nu_{I1l})} g^{J(\nu_{I1l})})^{r_l} \\ &= g_2^a (g_2^{F(\nu_{I1l})} g^{J(\nu_{I1l})})^{-\frac{a}{F(\nu_{I1l})}} (g_2^{F(\nu_{I1l})} g^{J(\nu_{I1l})})^{r_l} \\ &= g_2^a (u' \Pi_{j \in \nu_{I1l}} u_j)^{r_l - \frac{a}{F(\nu_{I1l})}} \\ &= g_2^a (u' \Pi_{j \in \nu_{I1l}} u_j)^{s_l} \end{aligned}$$

$$\begin{aligned} d_{1l} &= g_1^{\frac{-1}{F(\nu_{I1l})}} g_l^r \\ &= g^{r_l - \frac{a}{F(\nu_{I1l})}} \\ &= g^{s_l} \end{aligned}$$

Therefore, $d_{I1l} = (d_{0l}, d_{1l}) = (g_2^\alpha (u' \prod_{i \in \nu_{I1l}} u_j)^{s_l}, g^{s_l})$ is a valid key for identity ID_l and is given to adversary $\mathcal{C}$.

b) If it is in level 2, i.e $ID_l = (I1l, I2l)$, then $\mathcal{D}$ checks if $I2l = I2^*$, in which case it aborts and outputs a random guess γ', else it chooses $r_{1l}, r_{2l} \leftarrow \mathbb{Z}_p$ and sets $d_{ID_l} = (d_{0l}, d_{1l}, d_{2l}) = (g_2^{\frac{-\alpha_2}{I2l - I2^*}} (u' \prod_{j \in \nu_{I1l}} u_j)^{r_{1l}} F_2(I2l)^{r_{2l}}$, $(u' \prod_{j \in \nu_{I1l}} u_j)^{r_{1l}}$, $g_2^{\frac{-1}{I2l - I2^*}} g^{r_{2l}})$.
Let $s_l = r_{2l} - \frac{b}{I2l - I2^*}$. Then, d_{ID_l} is a valid secret key for ID_l, since the following two equalities hold:

$$
\begin{array}{ll}
d_{0l} & = (u'\prod_{j\in\nu_{I1l}} u_j)^{r_{1l}} g_2^{\frac{-\alpha_2}{I2l-I2^*}} F_2(I2l)^{r_{2l}} \\
= & (u'\prod_{j\in\nu_{I1l}} u_j)^{r_{1l}} g_2^{\frac{-\alpha_2}{I2l-I2^*}} (g_1^{I2l-I2^*} g^{\alpha_2})^{r_{2l}} \\
= & (u'\prod_{j\in\nu_{I1l}} u_j)^{r_{1l}} g_2^{a} (g_1^{I2l-I2^*} g^{\alpha_2})^{r_{2l}-\frac{b}{I2l-I2^*}} \\
= & g_2^a (u'\prod_{j\in\nu_{I1l}} u_j)^{r_{1l}} F_2(I2l)^{s_l}
\end{array}
$$

$$
\begin{array}{ll}
d_{2l} & = g_2^{\frac{-1}{I2l-I2^*}} g^{r_{2l}} \\
= & g^{r_{2l}-\frac{b}{I2l-I2^*}} \\
= & g^{s_l}
\end{array}
$$

Challenge When $\mathcal{C}$ decides that Phase 1 is over, it outputs two messages $M_0, M_1 \in \mathbb{G}_1$ of equal length and a level one identity $I1^*$ on which it wants to be challenged. If $x' + \Sigma_{i\in\nu_{I1^*}} \neq km$, $\mathcal{D}$ aborts and outputs a random bit γ'. Else, it flips a fair coin to obtain a random bit δ and responds with the ciphertext $C = (M_\delta T, g_3, g_3^{J(\nu_{I1^*})}, g_3^{\alpha_2})$. Since $F_2(I2^*)^c = (g^{\alpha_2})^c = g_3^{\alpha_2}$ and $g_3^{J(\nu_{I1^*})} = (g^{J(\nu I1^*)})^c = (g^{J(\nu I1^*)} g_2^{F(\nu_{I1^*})})^c = (u'\prod_{i\in\nu_{I1^*}} u_j)^c$ (because $F(\nu_{I1^*}) = 0 \bmod p$), then C will only be a ciphertext for M_δ if $T = e(g,g)^{abc}$.

Phase 2 As in phase 1, except that queries for identity $I1^*$ are no longer allowed, while queries for any of its children (except with suffix $I2^*$) are.

Guess Finally $\mathcal{C}$ outputs a guess δ'. The simulator $\mathcal{D}$ outputs $\gamma' = 1$ if $\delta = \delta'$, else it outputs $\gamma' = 0$.

Artificial Abort The probability of aborting when making first level extraction queries is not necessarily independent of the probability of $\mathcal{C}$ making a correct guess of the bit δ, since different sets of queries may have a different probability of aborting. The additional step artificial abort is introduced in order to compute the abort probability, as detailed below.
If $\overrightarrow{v} = v_1 \ldots v_{q_E}$ is the vector of all first level extraction queries made and v^* is the first level challenge identity, the following function is defined:

$$
\tau(X', \overrightarrow{v}, v^*) = \begin{cases} 0 & \text{if } (K(v_1) = 1) \wedge \ldots \wedge (K(v_{q_E}) = 1) \wedge (x' + \Sigma_{i\in\nu_{v^*}} x_i = km) \\ 1 & \text{otherwise} \end{cases}
$$

Note that the function evaluates to zero for a given set of extraction and challenge queries and simulation values X' when those choices lead to an abort.
The probability of aborting for a given set of queries $v^*, \overrightarrow{v}$, $\eta = Pr_{X'}[\tau(X', \overrightarrow{v}, v^*)]$ is sampled $O(\epsilon^{-2} ln(\epsilon^{-1})\lambda^{-1} ln(\lambda^{-1}))$ times, by choosing random X' and evaluating $\tau(X', \overrightarrow{v}, v^*)$ (sampling does not involve running the adversary again). The estimated value is η', while λ is the lower bound on the probability of not aborting for any set of queries (see [16] on how to compute λ).
We force adversary $\mathcal{D}$ to abort with probability $\frac{\eta'-\lambda}{\eta'}$ and take a random guess γ' if $\eta' \geq \lambda$. Otherwise, adversary $\mathcal{D}$ does not abort.
If $\mathcal{D}$ has not aborted at this point, it checks whether the guess of adversary $\mathcal{C}$, δ', is equal to δ. If so it outputs the guess $\gamma' = 1$, else it outputs $\gamma' = 0$.

Analysis When the input tuple is a random tuple, then $Pr[\gamma' = 1] = \frac{1}{2}$.
On the other hand, when the input tuple is a Diffie Hellman tuple:
$Pr[\gamma' = 1] = Pr[\gamma' = 1|abort]Pr[abort] + Pr[\gamma' = 1|\overline{abort}]Pr[\overline{abort}]$
Clearly, when the adversary does not abort, then $\mathcal{C}$ makes the correct guess with advantage ϵ, so $Pr[\gamma' = 1|\overline{abort}] = \frac{1}{2} + \epsilon$, while $Pr[\gamma' = 1|abort] = \frac{1}{2}$, because then the simulator outputs a random guess.
The probability of aborting comes exclusively from the first level extraction queries and the challenge query. In other words, the simulator aborts if and only if the simulator in the security proof of Waters [16] would also abort (making the same choices for $U, \overrightarrow{x}$, etc). Therefore, the probability of aborting can be calculated exactly in the same way as in the IBE scheme of Waters and the theorem follows.

7 Second construction: an Extended CBE scheme

We do not include algorithm *Consolidate* because it is trivial in this scheme, that is, if the outputs of the algorithm are $\langle params, periodi, userinfo, Cert'_i\rangle$, it simply outputs $Cert_{periodi} = Cert'_i$ (as it is also the case in [13]). This will also be the case for our final scheme New CBE.

ExtendedCBE

Setup$_{EXTCBE}$: Input: 1^k.
Run $\mathcal{IG}$ on input 1^k and obtain $\langle \mathbb{G}, \mathbb{G}_1, e\rangle$, $\mathbb{G}, \mathbb{G}_1$ of order p.
Choose $g, g_2, f_2 \leftarrow \mathbb{G}^*$, $\alpha \leftarrow \mathbb{Z}_p$. Set $g_1 = g^\alpha \in \mathbb{G}$
Choose $u', u_1, \ldots, u_n \leftarrow \mathbb{G}$. Set $U = (u', u_1, ..., u_n)$ and choose a collision resistant hash function $H_1 : \{0,1\}^* \rightarrow \{0,1\}^n$.
The space of messages is $\mathbb{G}_1$ and the system parameters are $params = (U, p, n, \mathbb{G}, \mathbb{G}_1, e, g, g_1, g_2, f_2, H_1)$. The CA's master secret key is $SK_{CA} = \alpha$.

SetKeyPair Input: *params*.
Choose $\beta \leftarrow \mathbb{Z}_p$, $h_2 \leftarrow \mathbb{G}$ and sets $h_1 = g^\beta \in \mathbb{G}$. The user's secret key is $SK = (\beta, h_2^\beta)$ and his public key is $PK = (h_1, h_2)$.
Define the following function $F_{2,h_1} : \mathbb{Z}_p \longrightarrow \mathbb{G}$ as $F_{2,h_1}(x) = g_1^x h_1^x f_2$.

Certify Input: $\langle$ *params*, SK_{CA}, *periodi*, *userinfo*, $(h_1, h_2)\rangle$.
Let $I1 = H_1(periodi||userinfo||(h_1, h_2))$. Pick $r_1 \leftarrow \mathbb{Z}_p$ and output: $(Cert_{i0}, Cert_{i1}) = (g_2^\alpha(u' \prod_{j\in\nu_{I1}} u_j)^{r_1}, g^{r_1})$.

KeyGen2 Input: $\langle params, \beta, (Cert_0, Cert_1)\ periodi, userinfo, (h_1, h_2), I2\rangle$.
Compute $I1 = H_1(periodi||userinfo||(h_1, h_2))$. Choose $r_2 \leftarrow \mathbb{Z}_p$. Set: $SK_{ID} = (d_0, d_1, d_2) = (g_2^\alpha(u' \prod_{j\in\nu_{I1}} u_j)^{r_1} F_{2,h_1}(I_2)^{r_2} h_2^\beta, g^{r_1}, g^{r_2})$.

Enc Input: $\langle params, M, i, userinfo, (h_1, h_2)\rangle$ and, optionally $I2$.
Choose $t \leftarrow \mathbb{Z}_p$.
Set $C = (C_1, C_2, C_3) = (Me(g_1, g_2)^t e(h_1, h_2)^t, g^t, (u' \prod_{j\in\nu_{I1}} u_j)^t)$ if user i is in level 1, else $C = (C_1, C_2, C_3, C_4) = (Me(g_1, g_2)^t e(h_1, h_2)^t, g^t, (u' \prod_{j\in\nu_{I1}} u_j)^t, F_{2,h_1}(I2)^t)$.

Dec Input: $\langle params, C, i, userinfo, (h_1, h_2), SK_{ID}, ID\rangle$, where SK_{ID} is $(d_0, d_1) = (Cert_{i0}h_2^\beta, Cert_{i1})$ if ID is a first level identity and (d_0, d_1, d_2) is the output of algorithm $KeyGen2$ defined above otherwise.
Set $I1 = H_1(periodi||userinfo||(h_1, h_2))$.
(a) If ID is in level 1, compute:

$$\frac{C_1 e(d_1, C_3)}{e(d_0, C_2)} = \frac{Me(g_1, g_2)^t e(h_1, h_2)^t e(g^{r_1}, (u' \prod_{j\in\nu_{I1}} u_j)^t)}{e(g_2^\alpha(u' \prod_{j\in\nu_{I1}} u_j)^{r_1} h_2^\beta, g^t)} = \ldots = M$$

(b) Else, compute:

$$\frac{C_1 e(d_1, C_3) e(d_2, C_4)}{e(d_0, C_2)} = \frac{Me(g_1, g_2)^t e(h_1, h_2)^t e(g^{r_1}, (u' \prod_{j\in\nu_{I1}} u_j)^t) e(g^{r_2}, F_2(I2)^t)}{e(g_2^\alpha(u' \prod_{j\in\nu_{I1}} u_j)^{r_1} F_2(I2)^{r_2} h_2^\beta, g^t)} = \ldots = M$$

7.1 Security proof: Adversary in game 1 against Extended CBE

Theorem Assuming H_1 to be a collision resistant hash function, if an adversary $\mathcal{B}_I$ succeeds in Extended CBE-Game 1 against the previously defined ExtendedCBE scheme, in time t, with advantage at most ϵ and making at most q_C certification queries and q_E extraction queries for second level identities, then there is an adversary $\mathcal{C}$ which succeeds in time $t' \leq t - \Theta(q_C + q_E)$ and with advantage ϵ in the game against the New 2-HIBE scheme. (where it is assumed that each evaluation of the hash function H_1 and each exponentiation in $\mathbb{G}$ take unit time).

Proof
Algorithm $\mathcal{C}$ interacts with algorithm $\mathcal{B}_I$ as follows:

Init When $\mathcal{B}_I$ outputs a second level identity $I2^*$ it wants to attack, $\mathcal{C}$ outputs the same identity.

Setup: The challenger runs $Setup_{HIBE}$, gives $params_{HIBE}$ to the adversary $\mathcal{C}$ and keeps msk to itself. Then $params_{EXTCBE} = (params_{HIBE}, H_1)$, where H_1 is a collision resistant hash function $H_1 : \{0,1\}^* \rightarrow \{0,1\}^n$, are given to the adversary $\mathcal{B}_I$.

Phase 1 Adversary $\mathcal{B}_I$ issues queries $q_1, \ldots, q_m$ where each q_j is:

a) a certification query $\langle periodi, userinfo, (h_1, h_2), (\beta, h_2^\beta)\rangle$. To answer this query, $\mathcal{C}$ checks that $userinfo \in \Lambda$ and that $\langle (h_1, h_2), (\beta, h_2^\beta)\rangle$ is a valid key-pair. If so, it asks the challenger for the secret key corresponding to identity $I1 = H_1((periodi||userinfo||(h_1, h_2)))$. This same answer is given to $\mathcal{B}_I$.

b) an extraction query $\langle ID, (\beta, h_2^\beta)\rangle$, where $ID = ((userinfo, i, (h_1, h_2)), I2)$ is a second level identity and $I2 \neq I2^*$. To answer this query, $\mathcal{C}$ checks that $\langle (h_1, h_2), (\beta, h_2^\beta)\rangle$ is a valid key-pair. If so, it asks the challenger for the secret key corresponding to identity $(H_1(periodi||userinfo||PK), I2) = (I1, I2) = ID$, and obtains $d_{ID} = (d_0, d_1, d_2) = (g_2^\alpha (u' \prod_{j\in\nu_{I1}} u_j)^{r_1} F_2(I_2)^{r_2}, g^{r_1}, g^{r_2})$.
Then $\mathcal{C}$ gives $\mathcal{B}_I$ the tuple $SK_{ID} = (d_0 h_2^\beta d_2^{\beta I2}, d_1, d_2)$. This is a valid secret since the following holds:

$$\begin{aligned} F_{2,h_1}(I2)^{r_2} &= (f_2 g_1^{I2} h_1^{I2})^{r_2} \\ &= F_2(I2)^{r_2} (h_1^{I2})^{r_2} \\ &= F_2(I2)^{r_2} (g^{r_2})^{\beta I2} \\ &= F_2(I2)^{r_2} d_2^{\beta I2} \end{aligned}$$

Therefore, $SK_{ID} = (d_0 h_2^\beta d_2^{\beta I2}, d_1, d_2) = (g_2^\alpha h_2^\beta (u' \prod_{j\in\nu_{I1}} u_j)^{r_1} F_{2,h_1}(I_2)^{r_2}, g^{r_1}, g^{r_2})$ is of the correct form.
These queries may be asked adaptively, that is, they may depend on the answers to previous queries.

Challenge On challenge query $\langle ID^*, (\beta^*, (h_2^*)^{\beta*}), M_0, M_1\rangle$, where $ID^* = (periodi^*, userinfo^*, (h_1^*, h_2^*))$ and $M_0, M_1 \in \mathcal{M}$ of equal length, $\mathcal{C}$ checks that $userinfo^* \in \Lambda$ and that $\langle (h_1^*, h_2^*), (\beta^*, (h_2^*)^{\beta*})\rangle$ is a valid key pair. If not, it outputs $\perp$, else it makes a challenge query $\langle M_0, M_1, I1^*\rangle$, where $I1^* = H_1(ID^*)$. The challenger responds by flipping a fair coin to choose a random bit b and returning the ciphertext $C = (C_1, C_2, C_3, C_4) = (Me(g_1, g_2)^t, g^t, (u' \prod_{j\in\nu_{I1*}} u_j)^t, F_2(I2^*)^t)$. Then, $\mathcal{C}$ gives $\mathcal{B}_I$ the challenge ciphertext $C^* = (C_1 e(C_2, h_2^{\beta*}), C_2, C_3, C_4(C_2)^{I2*\beta*}) = (Me(g_1, g_2)^t e(h_1, h_2)^t, g^t, (u' \prod_{j\in\nu_{I1*}} u_j)^t, F_{2,h_1}(I2^*)^t)$.

Phase 2 As in phase 1, except that certification queries $\langle ID^*, (\beta^*, (h_2^*)^{\beta*})\rangle$ are no longer allowed, but decryption queries for any identity $ID = ((periodi^*, userinfo^*, (h_1^*, h_2^*)), I2)$, with $I2 \neq I2^*$ are.

Guess Adversary $\mathcal{B}_I$ outputs a guess $b' \in \{0,1\}$, and $\mathcal{C}$ outputs the same guess.
The view of $\mathcal{B}_I$ is exactly the same as in the real attack, therefore the theorem follows.

7.2 Adversary in game 2 against ExtendedCBE

Theorem Assuming H_1 to be a collision resistant hash function, if an adversary $\mathcal{B}_{II}$ succeeds in Game 2 against the previously defined ExtendedCBE scheme, in time t, with advantage at most ϵ and making at most q_E extraction queries for second level identities, then there is an adversary $\mathcal{C}$ which succeeds in time $t' \leq t - \Theta(q_E)$ and with advantage ϵ in the game against the New 2-HIBE scheme. (where it is assumed that each evaluation of the hash function H_1 and each exponentiation in $\mathbb{G}$ take unit time).

Proof
Algorithm $\mathcal{C}$ interacts with algorithm $\mathcal{B}_{II}$ as follows:

Init Adversary $\mathcal{B}_{II}$ outputs a second level identity $I2^*$ it wants to attack. Then $\mathcal{C}$ outputs the same identity.

Setup The challenger runs $Setup_{HIBE}$, gives $params_{HIBE} = (U, p, n, \mathbb{G}, \mathbb{G}_1, e, g, g_1', g_2', f_2)$ to the adversary $\mathcal{C}$ and keeps $msk = (g_2')^{\alpha'}$ to itself.
Adversary $\mathcal{C}$ runs algorithm $SetKeyPair$ to obtain a pair public key - secret key $((h_1', h_2'), (h_2')^{\beta'})$. Then SK_{CA} and $params_{EXTCBE}$ are given to $\mathcal{B}_{II}$, where $SK_{CA} = g_2^{\alpha} = (h_2')^{\beta'}$ and $params_{EXTCBE} = (U, p, n, \mathbb{G}, \mathbb{G}_1, e, g, g_1 = h_1', g_2 = h_2', f_2, H_1)$, where $H_1 : \{0,1\}^* \rightarrow \{0,1\}^n$ is a collision resistant hash function. Finally, $\mathcal{C}$ gives to $\mathcal{B}_{II}$, the challenge public key $(h_1 = g_1', h_2 = g_2')$.

Phase 1 The adversary $\mathcal{B}_{II}$ issues queries $q_1, \ldots, q_m$ where each q_j is an extraction query for a second level identity $ID = ((periodi, userinfo, PK), I2)$ and $I2 \neq I2^*$.
Then $\mathcal{C}$ asks for the secret key corresponding to $(H_1(periodi||userinfo||PK), I2) = (I1, I2)$, and obtains $d_{(I1,I2)} = (d_0, d_1, d_2) = ((g_2')^{\alpha'}(u' \prod_{j\in\nu_{I1}} u_j)^{r_1} F_2(I2)^{r_2}, g^{r_1}, g^{r_2}) = (h_2^{\beta}(u' \prod_{j\in\nu_{I1}} u_j)^{r_1} F_2(I_2)^{r_2}, g^{r_1}, g^{r_2})$. Then $\mathcal{C}$ gives to $\mathcal{B}_I$ the secret key $SK_{ID} = (d_0 g_2^{\alpha} d_2^{\alpha I2}, d_1, d_2)$. This is a valid secret key, since the following holds:

$$\begin{aligned} F_{2,h_1}(I2)^{r_2} &= (f_2 g_1^{I2} h_1^{I2})^{r_2} \\ &= F_2(I2)^{r_2}(g_1^{I2})^{r_2} \\ &= F_2(I2)^{r_2}(g^{r_2})^{\alpha I2} \\ &= F_2(I2)^{r_2} d_2^{\alpha I2} \end{aligned}$$

These queries may be asked adaptively, that is, they may depend on the answers to previous queries.

Challenge On challenge query $\langle periodi^*, userinfo^*, M_0, M_1\rangle$, where $M_0, M_1 \in \mathcal{M}$ are of equal length, $\mathcal{C}$ checks that $userinfo^* \in \Lambda$. If any of these steps fails, it outputs $\perp$, else it makes the challenge query $\langle M_0, M_1, I1^*\rangle$, where $I1^* = H_1(periodi^*, userinfo^*, (h_1^*, h_2^*))$. To respond to this query, the challenger flips a fair coin to obtain a random bit b and returns $C = Enc_{params,(I1^*,\ I2^*)}(M_b) = (C_1, C_2, C_3, C_4) = (Me(g_1', g_2')^t, g^t, (u' \prod_{j\in\nu_{I1^*}} u_j)^t, F_2(I2^*)^t) = (Me(h_1, h_2)^t, g^t, (u' \prod_{j\in\nu_{I1^*}} u_j)^t, F_2(I2^*)^t)$. Then adversary $\mathcal{C}$ sets the challenge ciphertext to be $C^* = (C_1 e(C_2, g_2^{\alpha}), C_2, C_3, C_4(C_2)^{\alpha I2^*}) = (Me(g_1, g_2)^t e(h_1, h_2)^t, g^t, (u' \prod_{j\in\nu_{I1^*}} u_j)^t, F_{2,h_1}(I2^*)^t)$.

Phase 2 As in phase 1.

Guess The adversary $\mathcal{B}_{II}$ outputs a guess $b' \in \{0,1\}$, and $\mathcal{C}$ outputs the same guess.
The view of $\mathcal{B}_{II}$ is exactly the same as in the real attack, therefore the theorem follows.

8 A new CBE scheme without random oracles

NewCBE

Setup$_{CBE}$: Input: 1^k.
Run $\mathcal{IG}$ on input 1^k and obtain $\langle \mathbb{G}, \mathbb{G}_1, e\rangle$, $\mathbb{G}, \mathbb{G}_1$ of order p.
Choose $g, g_2, f_2 \leftarrow \mathbb{G}^*$, $\alpha \leftarrow \mathbb{Z}_p$. Set $g_1 = g^\alpha \in \mathbb{G}$
Choose $u', u_1, \ldots, u_n \leftarrow \mathbb{G}$. Set $U = (u', u_1, ..., u_n)$. Let $H_1 : \{0,1\}^* \longrightarrow \{0,1\}^n$, $H_2 : \{0,1\}^* \longrightarrow \mathbb{Z}_p$ be two collision resistant hash functions.
Run $Setup_{ENC}(1^k)$ to generate a string pub of an encapsulation scheme. The space of messages is $\mathbb{G}_1$ and the system parameters are $params = (U, p, n, \mathbb{G}, \mathbb{G}_1, e, g, g_1, g_2, f_2, H_1, H_2, pub)$. The CA's master secret key is $SK_{CA} = \alpha$.

SetKeyPair Input: $params$.
The user chooses $\beta \leftarrow \mathbb{Z}_p$, $h_2 \leftarrow \mathbb{G}$ and sets $h_1 = g^\beta \in \mathbb{G}$. The user's secret key is $SK = (\beta, h_2^\beta)$ and his public key is $PK = (h_1, h_2)$.
We define the following function $F_{2,h_1} : \mathbb{Z}_p \longrightarrow \mathbb{G}$ as $F_{2,h_1}(x) = g_1^x h_1^x f_2$.

Certify Input: $\langle$ $params$, SK_{CA}, $periodi$, $userinfo$, $(h_1, h_2)\rangle$.
Let $I1 = H_1(periodi||userinfo||(h_1, h_2))$. Pick $r \leftarrow \mathbb{Z}_p$ and output: $(Cert_{i0}, Cert_{i1}) = (g_2^\alpha(u' \prod_{j\in\nu_{I1}} u_j)^r, g^r)$.

Enc Input: $\langle params$, M, i, $userinfo$, $(h_1, h_2)\rangle$.
(a) Encapsulate a random value r by running $\mathcal{S}(1^k, pub)$ to obtain (r, com, dec)
(b) Let $I2 = H_2(com)$ and $I1 = H_1(periodi||userinfo||(h_1, h_2))$. Choose $t \leftarrow \mathbb{Z}_p$ and encrypt in the following way:
Set $C = ((M||dec)\ e(g_1, g_2)^t e(h_1, h_2)^t, g^t, (u' \prod_{j\in\nu_{I1}} u_j)^t, F_{2,h_1}(I2)^t)$.
(c) Compute $tag = Mac_r(C)$.
(d) Send $\langle com, C, tag\rangle$.

Dec Input: $\langle params, (Cert_{i0}, Cert_{i1}),\ (\beta, h_2^\beta),\ C,\ i,\ userinfo,\ (h_1, h_2)\rangle$, where $C = \langle com, (C_1, C_2, C_3, C_4), tag\rangle$.
Let $I1 = H_1(periodi||userinfo||(h_1, h_2))$, $I2 = H_2(com)$
(a) Derive the secret key corresponding to this period and com, by choosing $r_2 \leftarrow \mathbb{Z}_p$
$SK_{com,i} = (d_0, d_1, d_2) = (Cert_{i0} h_2^\beta F_{2,h_1}(I2)^{r_2}, Cert_{i1}, g^{r_2})$
(b) Decrypt in the following way:

$$\frac{C_1 e(d_1, C_3) e(d_2, C_4)}{e(d_0, C_2)} =$$

$$= \frac{(M||dec)\ e(g_1, g_2)^t e(h_1, h_2)^t e(g^r, (u'\Pi_{j\in\nu_{I1}} u_j)^t) e(g^{r_2}, F_{2,h_1}(I2)^t)}{e(g_2^\alpha h_2^\beta (u'\Pi_{j\in\nu_{I1}} u_j)^r F_{2,h_1}(I2)^{r_2}, g^t)} =$$

$$= \ldots = M||dec$$

(c) Obtain the string $r = \mathcal{R}(pub, com, dec)$ and verify if $tag = Mac_r(C)$. If this is the case, M is the correct decryption of C, else decryption fails.

8.1 Security Proof: Attack of an uncertified client

It is important to note that this and the following proof are generic, since they do not make use of any special properties of the underlying schemes.

Theorem Assuming the message authentication code and the encapsulation scheme used in New CBE above satisfy the security definitions given in sections 5.1 and 5.2, then, if an adversary $\mathcal{A}_I$ succeeds in time t and with advantage ϵ against the previously defined New CBE, then there is an adversary in game 1 against ExtendedCBE which succeeds with advantage negligibly close to ϵ and in time $t' \le t - \Theta(q_D)$, where each evaluation of the hash function H_2, pairing computation in $\mathbb{G}$, and execution of algorithms $\mathcal{R}$ and $Vrfy$ take unit time.

Proof
Algorithm $\mathcal{B}_I$ interacts with algorithm $\mathcal{A}_I$ as follows:

Init $\mathcal{B}_I$ runs Setup$_{ENC}(1^k, l(k))$ to generate pub, and runs $\mathcal{S}(1^k, pub)$ to obtain (r^*, com^*, dec^*). $\mathcal{B}_I$ outputs com^* as the second level identity it wants to attack.

Setup The challenger runs $Setup_{EXTCBE}(1^k)$ to generate SK_{CA} and $params_{EXTCBE}$. Then $params_{EXTCBE}$ are given to $\mathcal{B}_I$. Then $\mathcal{A}_I$ is given $params_{CBE} = (params_{EXTCBE}, H_2, pub)$, where $H_2 : \{0,1\} \to \mathbb{Z}_p$ is a collision resistant hash function.

Phase 1 $\mathcal{A}_I$ outputs queries $q_1, ...q_m$ where each of the q_i is:
a) Certification query $\langle periodi, userinfo, (h_1, h_2), (\beta, h_2^{\beta})\rangle$. To answer this query, $\mathcal{B}_I$ checks that $userinfo \in \Lambda$ and that $\langle (h_1, h_2), (\beta, h_2^{\beta})\rangle$ is a valid key-pair. If so, it makes this same certification query to the challenger.
b) Decryption queries $\langle i, userinfo, (h_1, h_2), (\beta, h_2^{\beta}),\ com, C, tag\rangle$. $\mathcal{B}_I$ checks that $com \neq com^*$ and that $\langle (h_1, h_2), (\beta, h_2^{\beta})\rangle$ is a valid key-pair. If this is not the case it outputs $\perp$, else it makes a second level extraction query for $\langle ID, (\beta, h_2^{\beta})\rangle = \langle ((periodi, userinfo,\ (h_1, h_2)), I2),\ (\beta, h_2^{\beta})\rangle$, where $I2 = H_2(com)$. Then $\mathcal{B}_I$ obtains the corresponding secret key SK_{ID} and uses it to decrypt C, obtain $M||dec$ and $r = \mathcal{R}(pub, com, dec)$ and $Vrfy_r(C, tag) = 1$. If any of these steps fails, $\mathcal{B}_I$ outputs $\perp$.

Challenge On challenge query $\langle I1^*, PK^*, SK^*, M_0, M_1\rangle = \langle periodi^*, userinfo^*, (h_1^*, h_2^*), (\beta^*, (h_2^*)^{\beta *}), M_0, M_1\rangle$, where $M_0, M_1 \in \mathcal{M}$ are of equal length, $\mathcal{B}_I$ checks that $userinfo^* \in \Lambda$ and that (PK^*, SK^*) is a valid key-pair. If so, it submits to the challenger the challenge query: $\langle I1^*, SK^*, M_0\ ||dec^*, M_1||dec^*\rangle$. The challenger chooses a random bit b and returns $C = Enc_{(I1^*,\ I2^*)}\ (M_b\ ||dec^*)$; else it returns $\perp$. Finally, $\mathcal{B}_I$ computes $tag^* = Mac_{r^*}(C)$ and sets the challenge ciphertext to be $C^* = \langle com^*, C, tag^*\rangle$.

Phase 2 As in Phase 1, except that certification queries for $\langle periodi^*, userinfo^*,\ (h_1^*, h_2^*), (\beta^*, (h_2^*)^{\beta *})\rangle$, are no longer allowed (but decryption queries for $\langle i^*,\ userinfo^*, (h_1^*, h_2^*),\ (\beta^*, (h_2^*)^{\beta *})\rangle$ are).

Guess Finally, $\mathcal{A}_I$ outputs a guess $b' \in \{0,1\}$. This same guess is output by $\mathcal{B}_I$.
A ciphertext is *valid* if it does not lead the simulator to abort in either CBE-game 1 or CBE-game 2 against NewCBE. Valid1 is the event that $\mathcal{A}_I$ ever makes a decryption query $\langle I1, (\beta, h_2^{\beta}), com^*, C, tag\rangle$ which is valid where $I1 = (periodi, userinfo, (h_1, h_2))$. We implicitly assume that $\langle com^*, C, tag\rangle \neq \langle com^*, C^*, tag^*\rangle$, since it occurs with only negligible probability before the challenge and it is disallowed after it.
Note that the only difference between the real game and the simulated game is when event Valid1 occurs.

Claim Pr[Valid1] is negligible.

We omit the proof here since it is a paraphrase of the proof of Boneh and Katz, except that now, to answer decryption queries the simulator is going to make second level extraction queries to the challenger instead of extraction queries as in the original proof of [9]. We just point out that this follows because of the security of the encapsulation and the commitment schemes.
Therefore, the theorem follows since:

$Pr[b' = b] = Pr[b' = b|\overline{\text{abort}}]Pr[\overline{\text{abort}}] + Pr[b' = b|\text{abort}]\Pr[\text{abort}] = (\frac{1}{2} + \epsilon)(1 - \Pr[\text{Valid1}]) + \frac{1}{2}\Pr[\text{Valid1}] = \frac{1}{2} + \epsilon(1 - \Pr[\text{Valid1}])$

8.2 Attack of the certifier

Theorem Assuming the message authentication code and the encapsulation scheme used in New CBE satisfy the security definitions given in sections 5.1 and 5.2, if an adversary $\mathcal{A}_{II}$ succeeds in time t and with advantage ϵ against the previously defined New CBE, then there is an adversary in game 2 against ExtendedCBE which succeeds with advantage negligibly close to ϵ in time $t' \leq t - \Theta(q_D)$, where it is assumed that every evaluation of H_2, pairing computation in $\mathbb{G}$, execution of algorithm $\mathcal{R}$ and $Vrfy$ take unit time.

Proof
Algorithm $\mathcal{B}_{II}$ interacts with algorithm $\mathcal{A}_{II}$ as follows:

Init $\mathcal{B}_{II}$ runs $\text{Setup}_{ENC}(1^k, l(k))$ to generate pub, and runs $\mathcal{S}(1^k, pub)$ to obtain (r^*, com^*, dec^*). $\mathcal{B}_{II}$ outputs com^* as the second level identity it wants to attack.

Setup The challenger runs $Setup_{EXTCBE}(1^k)$ to generate $SK_{CA} = g_1^{\alpha}$ and $params_{EXTCBE}$. It also runs algorithm $SetKeyPair$ to obtain a challenge public key - secret key pair $((h_1, h_2), (\beta, h_2^{\beta}))$. Then $params_{CBE} = (params_{EXTCBE}, H_2, pub)$ and SK_{CA} are given to $\mathcal{A}_{II}$, where $H_2 : \{0, 1\} \to \mathbb{Z}_p$ is a collision resistant hash function. The user's public key $PK = (h_1, h_2)$ is also given to $\mathcal{A}_{II}$.

Phase 1 $\mathcal{A}_{II}$ outputs queries $q_1, ...q_m$ where each q_i is a decryption query $\langle i,\ userinfo,\ com,\ C, tag\rangle$. $\mathcal{B}_{II}$ checks that $com \neq com^*$. If this is not the case it outputs $\perp$, else it makes a second level extraction query for $ID = ((periodi, userinfo, (h_1, h_2)), I2)$, where $I2 = H_2(com)$. The challenger responds to this query with the secret key SK_{ID} and $\mathcal{B}_{II}$ uses it to decrypt C, obtain $M||dec$ and $r = \mathcal{R}(pub, com, dec)$. Then $\mathcal{B}_{II}$ checks that $Vrfy_r(C, tag) = 1$. If any of these steps fails, $\mathcal{B}_{II}$ outputs $\perp$, else it responds to this query with M.

Challenge On challenge query $\langle periodi^*, userinfo^*,\ M_0, M_1\rangle$, where $M_0, M_1 \in \mathcal{M}$ are of equal length, $\mathcal{B}_{II}$ checks that $userinfo^* \in \Lambda$. If so, it submits to the challenger the challenge query: $\langle periodi^*,\ userinfo^*,\ M_0||dec^*, M_1||dec^*\rangle$. The challenger chooses a random bit b and returns $C^* = Enc_{(I1^*, I2^*)}\ (M_b\ ||dec^*)$, where $(I1^*, I2^*) = ((userinfo^*,\ periodi^*,\ (h_1, h_2)),\ H_2(com^*))$. If any of these steps fails, $\mathcal{B}_{II}$ returns $\perp$. Finally, $\mathcal{B}_{II}$ computes $tag^* = Mac_{r^*}(C)$ and sets the challenge ciphertext to be $\langle com^*, C^*, tag^*\rangle$.

Phase 2 As in Phase 1.

Guess Finally, $\mathcal{A}_{II}$ outputs a guess $b' \in \{0, 1\}$. This same guess is output by $\mathcal{B}_{II}$.

A ciphertext is *valid* if it does not lead the simulator to abort in either CBE-game 1 or CBE-game 2 against NewCBE. Valid2 is the event that $\mathcal{A}_{II}$ ever makes a decryption query $\langle I1, (\beta, h_2^{\beta}), com^*, C, tag\rangle$ which is valid where $I1 = (periodi^*, userinfo^*)$. We implicitly assume that $\langle com^*, C, tag\rangle \neq \langle com^*, C^*, tag^*\rangle$, since it occurs with only negligible probability before the challenge and it is disallowed after it.
Note that the only difference between the real game and the simulated game is when event Valid2 occurs.

Claim Pr[Valid2] is negligible.

We omit the proof here since it is again a paraphrase of the proof of Boneh and Katz, except that now, to answer decryption queries the simulator is going to make second level extraction queries to the challenger instead of extraction queries as in the original proof of [9]. As before, the theorem follows from the preceding claim.

9 Conclusion

In this paper we show how to use the techniques of Boneh and Katz in order to obtain full security for a CBE scheme. We reduce the problem to building an ExtendedCBE scheme, which seems a reasonable goal. If the result of Water is improved, the strategy of our proof could still be used in a straight forward way if the improved scheme made use of BLS signatures and it could support a second level of any IND-ID-CPA secure IBE scheme.

References

[1] Al- Riyami and K.G. Paterson. Certificateless public key cryptography. In Adv. In Cryptology - ASIACRYPT 2003, LNCS vol. 2894, pp. 452 - 47, Springer-Verlag, 2003.

[2] Al-Riyami and K.G. Paterson. CBE from CL-PKE: A generic construction and efficient scheme. *Public Key Criptography - PKC 2005*, LNCS vol. 3386, pp. 398- 415, 2005.

[3] D.Boneh, B.Lynn and H.Shacham. Short signatures from the Weil Pairing. In *Proc. of Asiacrypt 2001*, LNCS vol. 2248, pp. 514-532, Springer-Verlag, 2001.

[4] M. Bellare and A. Palacio. Protecting against Key Exposure: Strongly Key- Insulated Encryption with Optimal Threshold. Available ar http://eprint.iacr.org/2002/064, 2002.

[5] D.Boneh, C.Gentry, B.Lynn and H.Shacham. Aggregate and Verifiably Encrypted Signatures from the Weil Pairing. In *Adv. in Cryptology - Eurocrypt 2003*, LNCS vol. 2248, pp. 514-532, Springer- Verlag, 2003.

[6] D.Boneh and M.Franklin. Identity-Based Encryption From The Weil Pairing, *Adv.in Cryptology - Crypto 2001*, LNCS vol. 2139, pp.213-229, Springer-Verlag, 2001.

[7] D.Boneh and X.Boyen. Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles, *Adv.in Cryptology - Eurocrypt 2004*, LNCS vol. 3027, pp. 223-238, Springer-Verlag, 2004.

[8] D.Boneh and X.Boyen. Secure Identity Based Encryption Without Random Oracles, *Adv.in Cryptology - Crypto2004*, LNCS vol. 3152, pp. 443-459, Springer-Verlag, 2004.

[9] D.Boneh and J.Katz. Improved Efficiency for CCA-Secure Cryptosystems built using Identity-Based Encryption. In *Proceedings of CT-RSA 2005*, LNCS vol. 3376, Springer-Verlag, 2005.

[10] R.Canetti, S.Halevi and J.Katz. Chosen Ciphertext Security from Identity-Based Encryption, *Adv. in Cryptology -Eurocrypt 2004*, LNCS vol. 3027, Springer-Verlag, pp. 207-222, 2004.

[11] Y.Dodis and J.Katz. Chosen-Ciphertext Security of Multiple Encryption, *Theory of Cryptography Conference - TCC 2005*, LNCS vol. 3378, Springer-Verlag, pp. 188-209, 2005.

[12] E. Fujisaki and T.Okamoto, Secure integration of asymmetric and symmetric encryption schemes, *Adv. in Cryptology - Crypto 1999*, LNCS vol. 1666, pp. 537- 554, Springer- Verlag, 1999.

[13] C. Gentry. Certificate-Based Encryption and the Certificate-Revocation Problem, *Adv.in Cryptology - Eurocrypt 2003*, LNCS vol. 2656, pp. 272-291, Springer-Verlag, 2003.

[14] B.G.Kang and J.H.Park Is it possible to have CBE form CL-PKE? Available at http://eprint.iacr.org/2005/431, 2005.

[15] A.Shamir. Identity-based cryptosystems and signature schemes, *Adv. In Cryptology- Crypto 1984*, LNCS vol. 196, pp. 47-53, Springer-Verlag, 1985.

[16] B.Waters. Efficient Identity-Based Encryption Without Random Oracles, *Adv. in Cryptology- Eurocrypt 2005*, LNCS vol. 3494 , pp. 114-127, Springer-Verlag, 2005.

[17] D.H.Yum and P.J.Lee. Identity-based cryptography in public key management. In *EuroPKI 2004*, LNCS vol. 3093, pp. 71- 84, Springer-Verlag, 2004.