CCA2-Secure Threshold Broadcast Encryption with Shorter Ciphertexts

Vanesa Daza[1], Javier Herranz[2], Paz Morillo[3] and Carla Ràfols[3]

[1] Dept. D'Enginyeria Informàtica i Matemàtiques,
Universitat Rovira i Virgili
Av. Països Catalans 26, E-43007 Tarragona, Spain
e-mail: vanesa.daza@urv.cat

[2] Inst. d'Investigació en Intel·ligència Artificial,
Consejo Superior de Investigaciones Científicas
Campus UAB s/n, E-08193 Bellaterra, Spain
e-mail: jherranz@iiia.csic.es

[3] Dept. Matemàtica Aplicada IV,
Universitat Politècnica de Catalunya
C. Jordi Girona 1-3, E-08034 Barcelona, Spain
e-mail: {paz,crafols}@ma4.upc.edu

April 4, 2007

Abstract

In a threshold broadcast encryption scheme, a sender chooses (ad-hoc) a set of n receivers and a threshold t, and then encrypts a message by using the public keys of all the receivers, in such a way that the original plaintext can be recovered only if at least t receivers cooperate. Previously proposed threshold broadcast encryption schemes have ciphertexts whose length is $\mathcal{O}(n)$. In this paper, we propose new schemes, for both PKI and identity-based scenarios, where the ciphertexts' length is $\mathcal{O}(n-t)$. The construction uses secret sharing techniques and the Canetti-Halevi-Katz transformation to achieve chosen-ciphertext security. The security of our schemes is formally proved under the Decisional Bilinear Diffie-Hellman (DBDH) Assumption.

1 Introduction

In a threshold public key encryption scheme a message is encrypted and sent to a group of receivers, in such a way that the cooperation of at least t of them (where t is the threshold) is necessary in order to recover the original message. Such schemes have many applications in situations where one wants to avoid that a single party has all the power/responsibility to protect or obtain some critical information. The usual strategy to implement this idea is the following: the set of receivers, which is decided on from the beginning, runs an interactive setup protocol which takes as input a threshold (chosen by themselves) and outputs a public key for the set and shares of the matching secret key.

The fact that the set of receivers and the threshold are set from the beginning can limit the applications of these schemes in real life. One can imagine that the sender of the message, who wants to protect some information, may want to decide who will be the designated receivers in an ad-hoc way, just before encrypting the message, and also decide the threshold of receivers which will be necessary to recover the information (e.g. depending on the secrecy level of the message to be sent). With this motivation in mind, a scheme for this situation would have the following properties:

1. There is no setup phase or predefined groups. Each potential receiver has his own pair of secret/public keys.

2. The sender chooses (ad-hoc) the set of receivers $\mathcal{P}$ and the threshold t for the decryption. Then he encrypts the message by using the public keys of all the receivers in $\mathcal{P}$.

3. A ciphertext corresponding to the pair $(\mathcal{P}, t)$ can only be decrypted if at least t members of $\mathcal{P}$ cooperate by using their secret keys. Otherwise, it is computationally infeasible to obtain any information about the plaintext.

Note that, when $t = 1$, the resulting scheme will be a *broadcast encryption scheme* [10], where a sender encrypts a message in such a way that any member of the set of receivers can decrypt it. For this reason, we have decided to use the name *threshold broadcast encryption scheme* (TBE scheme, for short) to refer to this kind of schemes. Other possible names could be dynamic threshold encryption (as used in [11]) or ad-hoc threshold encryption. To the best of our knowledge, only two works have dealt with this extension of the concept of broadcast encryption. In [11] the authors propose a scheme based on RSA; even if the authors claim that the length of the ciphertexts is constant, the ciphertext contains an integer modulo N, where N is the product of all the RSA moduli of the receivers. Therefore, the actual length of the ciphertext is $\mathcal{O}(n)$, where n is the number of receivers. In [9], the authors propose a TBE scheme for identity-based scenarios; again, the length of the ciphertexts is $\mathcal{O}(n)$. In this same work [9], and previously in [7], a naive solution to the problem of threshold broadcast encryption was sketched: the sender distributes the message m into n pieces m_i, by using a threshold secret sharing scheme [12], and then encrypts each m_i by using the public key of the i-th receiver. The length of the ciphertext is also $\mathcal{O}(n)$.

In this paper we propose two new threshold broadcast encryption schemes, one for PKI-based scenarios and one for identity-based scenarios, where the length of the ciphertexts is $\mathcal{O}(n - t)$, being n the number of receivers and t the threshold for the decryption. We do not include the description of the set of receivers when we measure the length of the ciphertext; such a description can be quite short (for example, if the receivers are all the members of a company) or can be $\mathcal{O}(n)$-long, if the best/only way to describe the set is by including all the public keys of the receivers.

The idea in the design of our schemes is to combine the following tools: (1)

a chosen plaintext selective-ID secure identity-based encryption scheme; (2) some secret sharing techniques to create, for each encryption, an ad-hoc master public key whose matching master secret key will be distributed among the receivers of the message; (3) and the generic transformation due to Canetti, Halevi and Katz [8], to achieve chosen-ciphertext security. This last transformation has already been used by Boneh, Boyen and Halevi to construct a standard threshold encryption scheme [5] where interaction is not required among the decrypting servers (which differs from other previous schemes [14, 7]). This property is very desirable in our scenario of threshold broadcast encryption, where the receivers are chosen ad-hoc and maybe they do not know each other.

Our scheme for PKI-based scenarios uses a scheme by Boneh and Boyen [4] as the basic selective-ID secure identity-based encryption scheme. In this way, security of the resulting threshold broadcast encryption scheme under chosen-ciphertext attacks can be proved in the standard model. For the identity-based scenario, it is necessary to combine the techniques in the scheme of Boneh-Franklin [6] and the techniques in [4]; in this way, the maximum security of the obtained identity-based threshold broadcast encryption scheme is achieved in the random oracle model.

The rest of the work is organized as follows. In Section 2 we recall some tools (secret sharing, bilinear pairings, one-time signatures) that will be necessary for the construction of our schemes. In Section 3 we give the general definitions of the protocols and the security definitions for threshold broadcast encryption schemes. We propose our PKI-based scheme in Section 4, and we formally prove its security by reduction to the hardness of the Decisional Bilinear Diffie-Hellman (DBDH) problem. The description of our identity-based scheme is sketched in Section 5. Finally, we conclude our work in Section 6.

2 Preliminaries

2.1 Threshold Secret Sharing Schemes

The idea of *secret sharing schemes* was independently introduced by Shamir [12] and Blakley [3]. A (d, N)-threshold secret sharing scheme is a method by means of which a special figure, usually called *dealer*, distributes a secret s among a set $\mathcal{P} = \{R_1, \ldots, R_N\}$ of N players. Each player R_i privately receives from the dealer a piece of information s_i (or *share*). Then, those subsets with at least d players can recover the secret s from their shares, while subsets containing less than d players do not obtain any information at all about the secret.

Shamir's secret sharing scheme [12] solves this problem by means of polynomial interpolation. Let $GF(q)$ be a finite field with $q > N$ elements, and let $s \in GF(q)$ be the secret to be shared. The dealer picks a polynomial $f(x)$ of degree at most $d-1$, where the constant term of $f(x)$ is s and all other coefficients a_j are selected from $GF(q)$, uniformly and independently, at random. That is, $f(x)$ has the form $f(x) = s + \sum_{j=1}^{d-1} a_j x^j$.

Every player R_i is publicly and uniquely associated to a field element α_i. The dealer privately sends to player R_i his share $s_i = f(\alpha_i)$, for $i = 1, \ldots, N$.

Now, players in a set $A \subset \mathcal{P}$ such that $|A| \geq d$ can recover the secret $s = f(0)$, by using Lagrange interpolation. Actually, players in A can compute the value of the polynomial $f(x)$ evaluated on any point α_j, with the formula:

$$f(\alpha_j) = \sum_{R_i \in A} \lambda_{ij}^A f(\alpha_i) = \sum_{R_i \in A} \lambda_{ij}^A s_i,$$

where

$$\lambda_{ij}^A = \prod_{R_\ell \in A, \ell \neq i} \frac{\alpha_j - \alpha_\ell}{\alpha_i - \alpha_\ell}.$$

On the other hand, it can be proved that players in a subset $B \subset \mathcal{P}$ such that $|B| < d$ do not obtain any information about the polynomial $f(x)$, apart from their shares $\{f(\alpha_k)\}_{R_k \in B}$, of course.

2.2 Bilinear Pairings and Assumptions

Given an additive group $\mathbb{G}_1 = \langle P \rangle$ and a multiplicative group $\mathbb{G}_2$, both with prime order q, we say that they admit a bilinear pairing if there exists a map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ satisfying the following properties:

1. it is bilinear: $e(aP, bP) = e(P, P)^{ab} = e(bP, aP)$, for all $a, b \in \mathbb{Z}_q$;
2. it can be efficiently computed for any possible input pair;
3. it is non-degenerate, which means that $e(P, P) \neq 1$.

Bilinear pairings like the Tate or Weil pairings can be constructed over groups defined on elliptic curves. In the last years, bilinear pairings have been widely used in cryptography, for example in the design of identity-based cryptographic protocols. Identity-based cryptography was introduced by Shamir [13] as an alternative to traditional PKI-based cryptography, to avoid the efficiency problems related to the management of digital certificates which link a user with his public key. In identity-based cryptography, the public key of each user can be directly and publicly obtained from his identity, so an external link is not necessary. The negative point is that the secret keys of the users must be computed by a totally trusted (master) entity.

The security of these cryptographic schemes employing bilinear pairings is based on the assumption that some problems are hard to solve. These problems are adaptations to the bilinear pairing setting of more studied and standard computational problems, like the Decisional Diffie-Hellman (DDH) problem. The security of the schemes that we propose in this paper is based on the hardness of the following problem.

Definition 1. *(Decisional Bilinear Diffie-Hellman problem.) We say that an algorithm $\mathcal{S}$ is a ε'-solver of the DBDH problem if it distinguishes with probability at least $1/2 + \varepsilon'$ between the two following probability distributions:*

$\mathcal{D}_{BDH} = (P, aP, bP, cP, e(P, P)^{abc})$, where a, b, c are chosen uniformly and independently in $\mathbb{Z}_q$,

$\mathcal{D}_{rand} = (P, aP, bP, cP, T)$, where a, b, c are chosen uniformly and independently in $\mathbb{Z}_q$ and T is chosen uniformly and independently in $\mathbb{G}_2$.

In other words, a challenger chooses at random a bit $d \in \{0,1\}$. If $d = 1$, a tuple taken at random from $\mathcal{D}_{BDH}$ is given to $\mathcal{S}$. If $d = 0$, a tuple taken at random from $\mathcal{D}_{rand}$ is given to $\mathcal{S}$. The goal of $\mathcal{S}$ is to guess the bit d with better probability than at random. The Decisional Bilinear Diffie Hellman Assumption states that there does not exist any ε'-solver of the DBDH problem for non-negligible values of ε'.

2.3 One-Time Signatures

A one-time signature scheme $\Sigma = (\Sigma.KG, \Sigma.\text{Sign}, \Sigma.\text{Verify})$ consists of the three typical protocols of a digital signature scheme. $\Sigma.KG(1^k) \rightarrow (SK, VK)$ is the key generation protocol, which outputs a secret signing key SK and a public verification key VK. The signing protocol $\Sigma.\text{Sign}(SK, M) \rightarrow \sigma$ takes as input the signing key and a message M, and outputs a signature σ. Finally, the verification protocol $\Sigma.\text{Verify}(VK, M, \sigma) \rightarrow 1$ or 0 takes as input the verification key, a message and a signature, and outputs 1 if the signature is valid, or 0 otherwise.

Regarding security, we consider an adversary who first receives a verification key VK obtained from $\Sigma.KG(1^k) \rightarrow (SK, VK)$. He can make at most one signature query for a message M of his choice, obtaining as answer a valid signature $\Sigma.\text{Sign}(SK, M) \rightarrow \sigma$, and finally outputs a pair (M', σ'). We say that the adversary succeeds if $(M', \sigma') \neq (M, \sigma)$ and $\Sigma.\text{Verify}(VK, M', \sigma') \rightarrow 1$.

A one-time signature scheme Σ is ε_Σ-secure if any polynomial-time adversary against Σ has a success probability bounded by ε_Σ.

3 Threshold Broadcast Encryption

Roughly speaking, the operations of a threshold broadcast encryption scheme work as follows: the sender chooses a set of receivers and a threshold t, and then encrypts a message by using the public keys of these receivers. Given the resulting ciphertext, the original message can be recovered by any set of at least t of the designated receivers: they use their secret keys to compute partial decryptions which are then combined to obtain the message.

More formally, a threshold broadcast encryption scheme TBE= (TBE.Setup, TBE.KG, TBE.Enc, TBE.PartDec, TBE.Dec) consists of five algorithms:

- The randomized setup algorithm TBE.Setup takes as input a security parameter k and outputs some public parameters `params`, which will be common to all the users of the system. We write `params` $\leftarrow$ TBE.Setup(1^k).

- The randomized key generation algorithm TBE.KG is run by each user R_i. It takes as input some public parameters `params` and returns a pair (PK_i, SK_i) consisting of a public key and a matching secret key; we denote an execution of this protocol as $(PK_i, SK_i) \leftarrow$ TBE.KG(`params`).

- The randomized encryption algorithm TBE.Enc takes as input a set of public keys $\{PK_i\}_{R_i \in \mathcal{P}}$ corresponding to a set $\mathcal{P}$ of n receivers, a threshold t satisfying

$1 \leq t \leq n$, and a message m. The output is a ciphertext C, which contains the description of $\mathcal{P}$ and t; we write $C \leftarrow \text{TBE.Enc}(\mathcal{P}, \{PK_i\}_{R_i \in \mathcal{P}}, t, m)$.

- The (possibly randomized) partial decryption algorithm TBE.PartDec takes as input a ciphertext C for the pair $(\mathcal{P}, t)$ and a secret key SK_i of a receiver $R_i \in \mathcal{P}$. The output is a partial decryption value κ_i or a special symbol $\perp$. We denote with $\kappa_i \leftarrow \text{TBE.PartDec}(C, SK_i)$ an execution of this protocol.
- The deterministic final decryption algorithm TBE.Dec takes as input a ciphertext C for the pair $(\mathcal{P}, t)$ and t partial decryptions $\{\kappa_i\}_{R_i \in A}$ corresponding to receivers in some subset $A \subset \mathcal{P}$. The output is a message m or a special symbol $\perp$. We write $\tilde{m} \leftarrow \text{TBE.Dec}(C, \{\kappa_i\}_{R_i \in A}, A)$.

An important parameter of such schemes is the length of the ciphertext C. When measuring this length, we will not consider the description of the set $\mathcal{P}$: in some cases, the description can consist of the list of all the public keys, which already has length $\mathcal{O}(n)$. In some other cases, the description can be much simpler, for example if the set of receivers is formed by the workers of a company. For the previous TBE schemes in the literature [11, 9], the length of the ciphertexts is $\mathcal{O}(n)$. In this paper we will propose new schemes where the length of the ciphertexts is $\mathcal{O}(n - t)$.

3.1 Security of Threshold Broadcast Encryption Schemes

When formalizing security of standard public key encryption schemes, one usually considers a single challenged public key. This is because it has been shown [1] that security in this model is equivalent to security in a model which considers many public keys.

In threshold broadcast encryption schemes, however, we must consider many public keys when we formalize security, because each encryption and decryption in the system involves many public/secret keys. An attacker can corrupt different users, in two possible ways: registering new public keys for such users, or obtaining the secret key matching with the public key of some previously honest users. The final goal of the attacker is to obtain some information about a message which has been encrypted for a pair $(\mathcal{P}^*, t^*)$ such that the number of corrupted players in $\mathcal{P}^*$ is less than t^*.

For simplicity, we will not consider the first kind of user corruption, where the adversary registers new public keys. The reason is that, in the real world, certification authorities (should) require users to prove the knowledge of the secret key which matches with the public key they are registering. This can be done by means of a Proof of Knowledge [2]. In the game which models the security of threshold broadcast encryption schemes, the attacker is required to perform such a Proof of Knowledge of the secret keys which match with the new public keys he wants to register. Because of the 'proof of knowledge' property of the employed Proof of Knowledge system [2], this is equivalent to requiring the adversary to supply the matching secret key, each time he registers a public key. Therefore, registering new public keys does not give any useful information to the adversary.

Taking all this into consideration, indistinguishability for threshold broadcast encryption schemes is defined by considering the following game that an attacker $\mathcal{A}_{atk}$ plays against a challenger:

$\mathcal{U} = \emptyset$
$\texttt{params} \leftarrow \text{TBE.Setup}(1^k)$
Each time $\mathcal{A}_{atk}$ requires the creation of a new user R_i,
$(PK_i, SK_i) \leftarrow \text{TBE.KG}(\texttt{params})$ is executed and R_i is added to $\mathcal{U}$
$(St, \mathcal{P}^*, t^*, m_0, m_1) \leftarrow \mathcal{A}_{atk}^{Corr, \mathcal{O}_1(\cdot)}(\text{find}, \texttt{params}, \{PK_i\}_{R_i \in \mathcal{U}})$
$\beta \leftarrow \{0,1\}$ at random; $C^* \leftarrow \text{TBE.Enc}(\mathcal{P}^*, \{pk_i\}_{R_i \in \mathcal{P}^*}, t^*, m_\beta)$
$\beta' \leftarrow \mathcal{A}_{atk}^{Corr, \mathcal{O}_2(\cdot)}(\text{guess}, C^*, St)$.

In both phases of the attack, $\mathcal{A}_{atk}$ has access to a corruption oracle $Corr$: $\mathcal{A}_{atk}$ submits to the oracle a user $R_i \in \mathcal{U}$, and must receive as answer his secret key SK_i. Let $\mathcal{U}' \subset \mathcal{U}$ be the subset of users that $\mathcal{A}_{atk}$ has corrupted during the attack. In order to consider meaningful and successful such an attack, we require $|\mathcal{P}^* \cap \mathcal{U}'| < t^*$. Otherwise, $\mathcal{A}_{atk}$ knows the secret key of at least t^* players in $\mathcal{P}^*$ and can decrypt C^* by himself, obtaining m_β.

Depending on the considered kind of attacks, $\mathcal{A}_{atk}$ can also have access to a decryption oracle for ciphertexts C of his choice. As answer, $\mathcal{A}_{atk}$ receives all the information that would be broadcast in a complete decryption process for this tuple; this includes all the partial decryption values and the resulting plaintext. If atk is a chosen plaintext attack (CPA), then there is no access at all, i.e. $\mathcal{O}_1 = \mathcal{O}_2 = \epsilon$. If atk is a partial chosen ciphertext attack (CCA1), then $\mathcal{O}_1 = \text{TBE.PartDec}(\cdot) \cup \text{TBE.Dec}(\cdot)$ and $\mathcal{O}_2 = \epsilon$. Finally, if atk is a full chosen ciphertext attack (CCA2), then $\mathcal{O}_1 = \mathcal{O}_2 = \text{TBE.PartDec}(\cdot) \cup \text{TBE.Dec}(\cdot)$. In this last case, $\mathcal{A}_{\text{CCA2}}$ is not allowed to query the oracle $\mathcal{O}_2$ with the challenge ciphertext C^*.

The advantage of such an adversary $\mathcal{A}_{atk}$ is defined as

$$\text{Adv}(\mathcal{A}_{atk}) = \Pr[\beta' = \beta] - \frac{1}{2}.$$

A threshold broadcast encryption scheme is said to be ε-indistinguishable under atk attacks if $\text{Adv}(\mathcal{A}_{atk}) < \varepsilon$ for any attacker $\mathcal{A}_{atk}$ which runs in polynomial time.

Note that the definitions in this section (with slight changes) are valid also for identity-based scenarios: `params` will include a master public key, whereas the corresponding master secret key is used to compute secret keys for the users (identities) in a Key Extraction protocol. The sets of receivers will be sets of identities $\mathcal{P} = \{ID_1, \ldots, ID_n\}$. In the security game, the adversary is allowed to make key extraction (i.e. corruption) queries in order to obtain the secret keys for identities of his choice; this is reflected by the oracle $Corr$ in the game above.

4 A PKI-Based Threshold Broadcast Encryption Scheme with $|C| = \mathcal{O}(n - t)$

The idea behind the design of our scheme is to combine some threshold secret sharing techniques with the Canetti et al. [8] generic transformation, applied to the first selective-ID secure identity-based encryption scheme proposed by Boneh and Boyen in [4]. Note that the same idea has been used to construct a standard threshold encryption scheme (with fixed threshold and set of receivers) in [5]. The resulting schemes (ours and the one in [5]) enjoy two very good properties: security can be proved in the standard model and no interaction is needed among the receivers at the time of decryption.

The public parameters of our TBE scheme will be the public parameters of the scheme in [4] along with part (all but one element) of the master public key. The remaining element of the master public key will be computed ad-hoc by the sender of the message, from the public keys of the n receivers, in such a way that the corresponding ad-hoc master secret key is distributed into the secret keys of the receivers, by means of a (n, N)-threshold secret sharing scheme, where $N = 2n - t$.

Then, the sender generates a fresh pair of keys (SK, VK) for a one-time signature scheme, and encrypts the desired message by using the resulting identity-based public parameters and the identity $ID = VK$. Intuitively, n shares of the master secret key would be necessary to compute the secret key for the identity $ID = VK$ and therefore decrypt the ciphertext. The sender creates a subset of $n - t$ dummy users, out of the set of receivers, and adds to the ciphertext the secret decryption information that these users would provide. As a result, only t other partial decryption values, coming from the designated set of receivers, will be necessary to correctly perform decryption. Following the techniques in [8], the ciphertext is signed with SK. The resulting signature and the verification key VK are appended to the ciphertext.

In the two following sections, we detail the protocols of the proposed schemes, as well as the security proof.

4.1 The Scheme

Let $\Sigma = (\Sigma.KG, \Sigma.\text{Sign}, \Sigma.\text{Verify})$ be a secure one-time signature scheme. The five algorithms of the new TBE scheme work as follows.

Setup. Given a security parameter k, it generates a prime number q with k bits, and groups $\mathbb{G}_1 = \langle P \rangle$, $\mathbb{G}_2$ with order q which admit a bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, as described in Section 2.2. A hash function $h : \{0,1\}^* \to \mathbb{Z}_q$ is chosen, along with two random elements P_1 and Q from $\mathbb{G}_1$. The output of the protocol is $\texttt{params} = (q, \mathbb{G}_1, \mathbb{G}_2, P, e, h, P_1, Q)$. Any user R_i of the scheme will be publicly associated to a different element $\alpha_i \in \mathbb{Z}_q$ (for the secret sharing scheme). This can be done by defining $\alpha_i = g(R_i)$ for some public and collision-resistant hash function $g : \{0,1\}^* \to \mathbb{Z}_q$.

Key Generation. Each player (potential receiver) R_i chooses at random a value $\gamma_i \in \mathbb{Z}_q^*$. The public key is $PK_i = \gamma_i P$, whereas the secret key is $SK_i = \gamma_i P_1$.

Encryption. The goal is to encrypt a message $m \in \mathbb{G}_2$ addressed to some set $\mathcal{P} = \{R_1, \ldots, R_n\}$ of n receivers, with threshold t for the decryption. Let $\lambda_{i0}^{\mathcal{P}}$ be the Lagrange coefficients defined in Section 2.1, useful to compute the value of a polynomial in the point $\alpha_0 = 0$ from the value of the polynomial in the points $\{\alpha_i\}_{R_i \in \mathcal{P}}$. The sender must act as follows.

1. Run $\Sigma.KG(1^k) \rightarrow (SK, VK)$.
2. Define $P_2 = \sum\limits_{R_i \in \mathcal{P}} \lambda_{i0}^{\mathcal{P}} PK_i$.
3. Choose a set $\tilde{\mathcal{P}}$ of $n-t$ (dummy) players, such that $\tilde{\mathcal{P}} \cap \mathcal{P} = \emptyset$. For each $R_j \in \tilde{\mathcal{P}}$, consider the corresponding $\alpha_j \in \mathbb{Z}_q$ and then define $PK_j = \sum\limits_{R_i \in \mathcal{P}} \lambda_{ij}^{\mathcal{P}} PK_i$.
4. Choose at random $s \in \mathbb{Z}_q^*$ and compute $C_1 = sP$.
5. Compute $C_2 = m \cdot e(P_1, P_2)^s$.
6. Compute $C_3 = s\,[h(VK)P_1 + Q]$.
7. For each $R_j \in \tilde{\mathcal{P}}$, choose $r_j \in \mathbb{Z}_q$ at random and compute $\kappa_j = \frac{e(C_3, r_j P)}{e(PK_j, sP_1) \cdot e(C_1, r_j[h(VK)P_1 + Q])}$.
8. Define $C' = (\mathcal{P}, t, \tilde{\mathcal{P}}, C_1, C_2, C_3, \{\kappa_j\}_{R_j \in \tilde{\mathcal{P}}})$.
9. Run $\Sigma.\text{Sign}(SK, C') \rightarrow \sigma$.
10. Define the final ciphertext as $C = (VK, C', \sigma)$.

Note that, excluding the description of the sets $\mathcal{P}$ and $\tilde{\mathcal{P}}$, which can have different lengths depending on the case, a ciphertext C contains $\mathcal{O}(n-t)$ elements.

Partial Decryption, EG_TBE.PartDec. Given a ciphertext $C = (VK, C', \sigma)$, any receiver $R_i \in \mathcal{P}$ first runs $\Sigma.\text{Verify}(VK, C', \sigma)$. If the result is 0 (invalid ciphertext), then the special symbol $\perp$ is output. Otherwise (valid ciphertext) let $C' = (\mathcal{P}, t, \tilde{\mathcal{P}}, C_1, C_2, C_3, \{\kappa_j\}_{R_j \in \tilde{\mathcal{P}}})$. Receiver R_i chooses $r_i \in \mathbb{Z}_q$ at random; the partial decryption that is broadcast by R_i is

$$\kappa_i = \frac{e(C_3, r_i P)}{e(C_1, SK_i + r_i[h(VK)P_1 + Q])}.$$

Final Decryption, EG_TBE.Dec. Given a valid ciphertext $C = (VK, C', \sigma)$, with $C' = (\mathcal{P}, t, \tilde{\mathcal{P}}, C_1, C_2, C_3, \{\kappa_j\}_{R_j \in \tilde{\mathcal{P}}})$, and a set of t partial decryptions κ_i, corresponding to a subset $A \subset \mathcal{P}$ with $|A| = t$, a combiner algorithm considers the whole set of partial decryptions in $B = A \cup \tilde{\mathcal{P}}$ and then computes

$$\kappa = \prod_{R_i \in B} \kappa_i^{\lambda_{i0}^B} = \frac{e(C_3, P)^{\sum\limits_{R_i \in B} \lambda_{i0}^B r_i}}{e(P_1, \sum\limits_{R_i \in B} \lambda_{i0}^B PK_i)^s \cdot e(C_1, h(VK)P_1 + Q)^{\sum\limits_{R_i \in B} \lambda_{i0}^B r_i}} =$$

$$= \frac{e(h(VK)P_1 + Q, P)^{s \sum_{R_i \in B} \lambda_{i0}^B r_i}}{e(P_1, P_2)^s \cdot e(P, h(VK)P_1 + Q)^{s \sum_{R_i \in B} \lambda_{i0}^B r_i}} = \frac{1}{e(P_1, P_2)^s}.$$

The plaintext m is recovered by computing $m = C_2 \cdot \kappa$.

4.2 Security Analysis

Theorem 1. *Suppose Σ is ε_Σ-secure, which means that any polynomial-time forger against Σ has a success probability bounded by ε_Σ.*

If there exists a CCA2 attack $\mathcal{A}$ against the proposed TBE scheme, with advantage ε and corrupting at most q_c users, then the Decisional Bilinear Diffie-Hellman (DBDH) problem can be solved with advantage $\varepsilon' \geq \frac{\varepsilon(1-\varepsilon_\Sigma)}{6(q_c+1)}$.

Proof. Let $\mathcal{D} = (P, aP, bP, cP, T)$ be an instance of the DBDH problem (which includes the description of $\mathbb{G}_1 = \langle P \rangle, \mathbb{G}_2, e$). The goal of a solver $\mathcal{S}$ is to decide if $\mathcal{D} \in \mathcal{D}_{BDH}$ (that is, if $T = e(P, P)^{abc}$), or if $\mathcal{D} \in \mathcal{D}_{rand}$ (that is, if T is a random value in $\mathbb{G}_2$). In the first case, the output of a solver is the bit $d = 1$, whereas in the second case, the output is the bit $d = 0$. We are going to construct such a solver $\mathcal{S}$ for this problem. $\mathcal{S}$ first runs $\Sigma.KG(1^k) \to (SK^*, VK^*)$. $\mathcal{S}$ chooses two suitable hash functions $h, g : \{0,1\}^* \to \mathbb{Z}_q$. The solver chooses at random $\eta \in \mathbb{Z}_q$ and defines $P_1 = aP$ and $Q = -h(VK^*)P_1 + \eta P$. At this point, $\mathcal{S}$ initializes the hypothetical attacker $\mathcal{A}$ against the TBE scheme, with input $\texttt{params} = (q, \mathbb{G}_1, \mathbb{G}_2, P, e, h, g, P_1, Q)$.

Each time $\mathcal{A}$ asks for the creation of a new user R_i, the solver $\mathcal{S}$ chooses at random $\gamma_i \in \mathbb{Z}_q^*$. Let $\mu \in (0,1)$ be a real number to be determined later. With probability μ, the value $c_i = 0$ is chosen, and then $\mathcal{S}$ defines $PK_i = \gamma_i P$ (in this case, $SK_i = \gamma_i P_1$ is known to $\mathcal{S}$). On the other hand, with probability $1 - \mu$, the value $c_i = 1$ is chosen, and $\mathcal{S}$ defines $PK_i = \gamma_i(bP)$ (in this case, $\mathcal{S}$ does not know the value of SK_i). The public key PK_i is sent back to $\mathcal{A}$. These values are stored in a table.

$\mathcal{A}$ is allowed to corrupt some users. If $\mathcal{A}$ sends a corruption query for user R_i, the solver $\mathcal{S}$ looks for c_i in the table. If $c_i = 0$, then the value $SK_i = \gamma_i P_1$ is sent to $\mathcal{A}$. Otherwise, if $c_i = 1$, the solver $\mathcal{S}$ aborts and outputs a random bit $d \in \{0,1\}$. If the number of corruption queries from $\mathcal{A}$ is q_c, then the probability that $\mathcal{S}$ does not abort in this phase is μ^{q_c}.

The CCA2 attacker $\mathcal{A}$ can make decryption queries for ciphertexts $C = (VK, C', \sigma)$ of his choice. The solver $\mathcal{S}$ acts as follows:

- If $VK = VK^*$, then $\mathcal{S}$ runs Σ.Verify(VK, C', σ). If the output is 0, $\mathcal{S}$ replies with $\perp$. If the output is 1 (valid signature), then $\mathcal{S}$ aborts and outputs a random bit $d \in \{0,1\}$.

- If $VK \neq VK^*$ and Σ.Verify(VK, C', σ) outputs 0, then $\mathcal{S}$ replies with $\perp$.

- If $VK \neq VK^*$ and Σ.Verify(VK, C', σ) outputs 1, then we have $C' = (\mathcal{P}, t, \tilde{\mathcal{P}}, C_1, C_2, C_3, \{\kappa_j\}_{R_j \in \tilde{\mathcal{P}}})$ and $h(VK) \neq h(VK^*)$ (otherwise, $\mathcal{A}$ would have found a collision on the hash function h, which is considered to be computationally infeasible). The solver $\mathcal{S}$ must simulate the partial decryption values κ_i for $R_i \in \mathcal{P}$. If $c_i = 0$, then $\mathcal{S}$ knows SK_i and κ_i can be computed as in the description of the protocol. Otherwise, $\mathcal{S}$ chooses $r_i \in \mathbb{Z}_q$ at random and computes

$$\kappa_i = \frac{e\left(C_3, r_i P - \frac{\gamma_i}{h(VK)-h(VK^*)}(bP)\right)}{e\left(C_1, \frac{-\gamma_i \eta}{h(VK)-h(VK^*)}(bP) + r_i[(h(VK) - h(VK^*))P_1 + \eta P]\right)}.$$

It is not difficult to see that this value κ_i is a correct partial decryption value for R_i, computed with the implicit random value $\tilde{r}_i = r_i - \frac{b\gamma_i}{h(VK)-h(VK^*)}$.

When all the values κ_i are computed, $\mathcal{S}$ can recover κ and the plaintext $m = C_2 \cdot \kappa$. The solver $\mathcal{S}$ sends m and $\{\kappa_i\}_{R_i \in \mathcal{P}}$ to $\mathcal{A}$.

At some point, $\mathcal{A}$ broadcasts a set $\mathcal{P} = \{R_1, \dots, R_n\}$, a threshold t such that $1 \le t \le n$, and two messages $m_0, m_1 \in \mathbb{G}_2$, such that the number of corrupted users in $\mathcal{P}$ is less than t. This means that at least one user $R_u \in \mathcal{P}$ has not been corrupted by $\mathcal{A}$. With probability $1 - \mu$, we have $c_u = 1$ and so $PK_u = \gamma_u(bP)$. In general, we define $\mathcal{P}_0 = \{R_i \in \mathcal{P} : c_i = 0\}$ and $\mathcal{P}_1 = \{R_\ell \in \mathcal{P} : c_\ell = 1\}$. As we have just said, $\mathcal{P}_1$ is not empty with probability at least $1 - \mu$. If this is not the case, $\mathcal{S}$ aborts and outputs a random bit $d \in \{0, 1\}$.

For the challenge ciphertext to be given to $\mathcal{A}$, the solver $\mathcal{S}$ defines $C_1 = cP$ (note that c is unknown to $\mathcal{S}$) and takes VK^* for the verification key. Note that, because of the way in which Q is defined, we have $C_3 = \eta C_1$. To compute C_2, the value $e(P_1, P_2)^c$ must be multiplied with the plaintext. If we define $P_2 = \sum\limits_{R_i \in \mathcal{P}} \lambda_{i0}^{\mathcal{P}} PK_i$, and recalling that $P_1 = aP$, we have

$$\begin{aligned} e(P_1, P_2)^c &= e\big(aP, \sum_{R_i \in \mathcal{P}_0} \lambda_{i0}^{\mathcal{P}} \gamma_i P\big)^c \cdot e\big(aP, \sum_{R_\ell \in \mathcal{P}_1} \lambda_{\ell 0}^{\mathcal{P}} \gamma_\ell (bP)\big)^c = \\ &= e(aP, cP)^{\sum_{R_i \in \mathcal{P}_0} \lambda_{i0}^{\mathcal{P}} \gamma_i} \cdot e(P, P)^{abc \sum_{R_\ell \in \mathcal{P}_1} \lambda_{\ell 0}^{\mathcal{P}} \gamma_\ell}. \end{aligned}$$

$\mathcal{S}$ chooses a set $\tilde{\mathcal{P}}$ of $n - t$ dummy users such that $\mathcal{P} \cap \tilde{\mathcal{P}} = \emptyset$. Analogously, defining $PK_j = \sum\limits_{R_i \in \mathcal{P}} \lambda_{ij}^{\mathcal{P}} PK_i$ for these users, we have

$$e(PK_j, cP_1) = e(cP, aP)^{\sum_{R_i \in \mathcal{P}_0} \lambda_{ij}^{\mathcal{P}} \gamma_i} \cdot e(P, P)^{abc \sum_{R_\ell \in \mathcal{P}_1} \lambda_{\ell j}^{\mathcal{P}} \gamma_\ell}.$$

$\mathcal{S}$ chooses a random bit $\beta \in \{0, 1\}$ and defines

$$C_2 = m_\beta \cdot \left(e(aP, cP)^{\sum_{R_i \in \mathcal{P}_0} \lambda_{i0}^{\mathcal{P}} \gamma_i} \cdot T^{\sum_{R_\ell \in \mathcal{P}_1} \lambda_{\ell 0}^{\mathcal{P}} \gamma_\ell} \right).$$

And, for every dummy user $R_j \in \tilde{\mathcal{P}}$, $\mathcal{S}$ chooses $r_j \in \mathbb{Z}_q$ at random and defines

$$\kappa_j = \frac{e(C_3, r_j P)}{e(cP, aP)^{\sum_{R_i \in \mathcal{P}_0} \lambda_{ij}^{\mathcal{P}} \gamma_i} \cdot T^{\sum_{R_\ell \in \mathcal{P}_1} \lambda_{\ell j}^{\mathcal{P}} \gamma_\ell} \cdot e(C_1, r_j[h(VK^*)P_1 + Q])}.$$

Note that the resulting $C'^* = (\mathcal{P}, t, \tilde{\mathcal{P}}, C_1, C_2, C_3, \{\kappa_j\}_{R_j \in \tilde{\mathcal{P}}})$ is consistent if and only if $T = e(P,P)^{abc}$. After that, $\mathcal{S}$ runs $\Sigma.\mathrm{Sign}(SK^*, C'^*) \to \sigma^*$. The final challenge ciphertext that is sent to $\mathcal{A}$ is $C^* = (VK^*, C'^*, \sigma^*)$.

At this point, $\mathcal{A}$ is allowed to make new decryption queries, which are replied by $\mathcal{S}$ exactly in the same way as before. After that, $\mathcal{A}$ outputs its guess β'. If $\beta' = \beta$, then $\mathcal{S}$ outputs $d = 1$ (meaning that it believes that $T = e(P,P)^{abc}$ and so $\mathcal{D} \in \mathcal{D}_{BDH}$). If $\beta' \neq \beta$, then $\mathcal{S}$ outputs $d = 0$ (meaning that it believes that T is a random value in $\mathbb{G}_2$ and so $\mathcal{D} \in \mathcal{D}_{rand}$).

Let us compute the success probability of $\mathcal{S}$. We assume that in the input of the DBDH problem, $\mathcal{D} \in \mathcal{D}_{BDH}$ with probability 1/2. Let us denote as ρ the probability that $\mathcal{S}$ does not abort in any phase. We have

$$\Pr[\mathcal{S} \text{ succeeds}] = \frac{1}{2}\Pr[\mathcal{S} \text{ succeeds } / \ \mathcal{D} \in \mathcal{D}_{BDH}] \ + \ \frac{1}{2}\Pr[\mathcal{S} \text{ succeeds } / \ \mathcal{D} \in \mathcal{D}_{rand}] \geq$$

$$\geq \frac{1}{2}\left[\Pr[\mathcal{S} \text{ does not abort}] \cdot (\frac{1}{2} + \varepsilon) \ + \ \Pr[\mathcal{S} \text{ aborts}] \cdot \frac{1}{2}\right] \ + \ \frac{1}{2} \cdot \frac{1}{2} \geq$$

$$\geq \frac{1}{4}\rho + \frac{1}{2}\rho\varepsilon + \frac{1}{4}(1-\rho) + \frac{1}{4} = \frac{1}{2} + \frac{\rho\varepsilon}{2}.$$

Let us denote as δ the probability that $\mathcal{A}$ makes a decryption query for a valid ciphertext $C = (VK^*, C', \sigma)$ such that $(C', \sigma) \neq (C'^*, \sigma^*)$.

Claim 1. *$\delta \leq \varepsilon_\Sigma$.*

Proof. We are going to prove that a CCA2 attacker $\mathcal{A}$ which makes a valid query $C = (VK^*, C', \sigma)$ to the decryption oracle, such that $(C', \sigma) \neq (C'^*, \sigma^*)$, with probability δ, can be used to construct a forger $\mathcal{F}$ against the one-time signature scheme Σ, with success probability δ.

When $\mathcal{F}$ receives VK^* as input, he generates `params` and initializes $\mathcal{A}$. Each time $\mathcal{A}$ asks for the creation of a new user R_i, the forger $\mathcal{F}$ generates the secret key and public key (SK_i, PK_i) for R_i. Later, $\mathcal{F}$ can answer all the corruption and decryption queries made by $\mathcal{A}$, because $\mathcal{F}$ knows all the secret keys. For the challenge ciphertext, $\mathcal{F}$ chooses VK^* as the verification key, and uses his only query to his signing oracle to obtain $\Sigma.\mathrm{Sign}(SK^*, C'^*) \to \sigma^*$.

If at some point $\mathcal{A}$ makes a decryption query for a valid ciphertext $C = (VK^*, C', \sigma)$ verifying $(C', \sigma) \neq (C'^*, \sigma^*)$, then $\mathcal{F}$ aborts and outputs (C', σ) as his valid forgery. □

The probability that $\mathcal{S}$ does not abort at any point is $\rho \geq \mu^{q_c}(1-\mu)(1-\delta)$. This value is maximized when $\mu = \frac{q_c}{q_c+1}$, which leads to

$$\rho \geq \left(\frac{1}{1+\frac{1}{q_c}}\right)^{q_c} \cdot \frac{1}{q_c+1} \cdot (1-\delta) \geq \frac{1}{e} \cdot \frac{1-\delta}{q_c+1}.$$

Therefore, the advantage of $\mathcal{S}$ in solving the DBDH problem is

$$\varepsilon' \geq \frac{\rho\varepsilon}{2} \geq \frac{\varepsilon(1-\delta)}{2(q_c+1)e} \geq \frac{\varepsilon(1-\delta)}{6(q_c+1)} \geq \frac{\varepsilon(1-\varepsilon_\Sigma)}{6(q_c+1)}.$$

□

5 The Identity-Based Case

The generic transformation of Canetti, Halevi and Katz [8], that we have used as building tool for the construction of our TBE scheme in the previous section, works also to obtain CCA2 secure identity-based cryptosystem from a 2-level hierarchical identity-based cryptosystem with chosen plaintext selective-ID security. Therefore, we should in principle be able to construct an identity-based TBE scheme with maximum security in the standard model, by starting from the 2-level hierarchical scheme in [4]. However, this particular scheme does not seem to properly adapt to the scenario of threshold broadcast encryption.

Nevertheless, it is possible to construct a secure identity-based TBE scheme by following the same idea, but applied to a different scheme. Intuitively, what we need is the Boneh-Franklin identity-based scheme [6] for the first level of identities (i.e. the identities of the receivers) and the Boneh-Boyen identity-based scheme [4] for the second level of identities (corresponding to the verification key VK). A consequence of using the Boneh-Franklin scheme is that our scheme will achieve provable CCA2 security in the random oracle model.

Actually, our proposal of identity-based TBE scheme, which results from applying this combination, is very similar to the PKI scheme described and analyzed in Section 4. For this reason, we only sketch the main differences between them:

- The Setup phase is now run by the master entity. It is the same as in the PKI scheme, but now the element P_1 is computed as $P_1 = \gamma P$ for some random $\gamma \in \mathbb{Z}_q$ that the master entity keeps secret. An additional hash function $H : \{0,1\} \to \mathbb{G}_1$ is chosen and made public.

- The Key Generation phase of the PKI scheme is replaced with a Key Extraction protocol, run by the master entity each time a user with identity ID_i asks for his secret key. The public key of ID_i is easily (and publicly) computable as $PK_i = H(ID_i)$, whereas the corresponding secret key $SK_i = \gamma PK_i$ is computed and delivered by the master entity. Note that in both (PKI and identity-based) schemes the tuples (P, P_1, PK_i, SK_i) are Diffie-Hellman tuples. In particular, in the identity-based scheme, a user can verify that the obtained secret key is consistent, by checking if $e(P, SK_i) = e(P_1, PK_i)$.

- In the security proof, the part of the proof of Theorem 1 where the pairs of keys (PK_i, SK_i) are generated is now the part of the proof where the solver $\mathcal{S}$ answers the queries that the attacker $\mathcal{A}$ makes to the random oracle for H. But the result is the same: for some users ($c_i = 0$) the solver $\mathcal{S}$ will define $H(ID_i) = PK_i = \gamma_i P$ and so $\mathcal{S}$ will know the corresponding SK_i, whereas for other users ($c_i = 1$) the solver $\mathcal{S}$ will define $H(ID_i) = PK_i = \gamma_i(bP)$.

As a result, we obtain an identity-based TBE scheme which is CCA2 secure in the random oracle model, under the Decisional Bilinear Diffie-Hellman Assumption.

6 Conclusion

Threshold broadcast encryption (TBE) schemes differ from traditional threshold public key encryption schemes [14, 7, 5] because the group of receivers and the threshold for decryption are not decided from the beginning, but chosen (ad-hoc) by the entity who encrypts each message. This difference makes TBE schemes more suitable for some applications in real life.

In this work we have designed TBE schemes with shorter ciphertexts than previous proposals, for both PKI-based and identity-based scenarios. The schemes achieve the highest possible level of security (against chosen-ciphertext attacks) assuming that the Decisional Bilinear Diffie-Hellman problem is hard.

Many problems remain open in this area. For example, to design TBE schemes with ciphertexts' length shorter than $\mathcal{O}(n)$ (for PKI or identity-based scenarios) which do not employ bilinear pairings; or to design an identity-based TBE scheme, also with ciphertexts' length shorter than $\mathcal{O}(n)$, which achieves the maximum security in the standard model. Another interesting question to be answered is whether the bound $\mathcal{O}(n-t)$ for the ciphertexts' length can be lowered, for fully secure TBE schemes.

References

[1] M. Bellare, A. Boldyreva and S. Micali. Public-key encryption in a multi-user setting: security proofs and improvements. *Proceedings of Eurocrypt'00*, LNCS **1807**, Springer-Verlag, pp. 259–274 (2000).

[2] M. Bellare and O. Goldreich. On defining proofs of knowledge. *Proceedings of Crypto'92*, LNCS **740**, Springer-Verlag, pp. 390–420 (1992).

[3] G.R. Blakley. Safeguarding cryptographic keys. *Proceedings of the National Computer Conference, American Federation of Information*, Processing Societies Proceedings **48**, pp. 313–317 (1979).

[4] D. Boneh and X. Boyen. Efficient selective-ID secure identity-based encryption without random oracles. *Proceedings of Eurocrypt'04*, LNCS **3027**, Springer-Verlag, pp. 223–238 (2004).

[5] D. Boneh, X. Boyen and S. Halevi. Chosen ciphertext secure public key threshold encryption without random oracles. *Proceedings of CT-RSA'06*, LNCS **3860**, Springer-Verlag, pp. 226–243 (2006).

[6] D. Boneh and M.K. Franklin. Identity-based encryption from the Weil pairing. *SIAM Journal on Computing*, vol. **32** (3), pp. 586–615 (2003).

[7] R. Canetti and S. Goldwasser. An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. *Proceedings of Eurocrypt'99*, LNCS **1592**, Springer-Verlag, pp. 90–106 (1999).

[8] R. Canetti, S. Halevi and J. Katz. Chosen-ciphertext security from identity-based encryption. *Proceedings of Eurocrypt'04*, LNCS **3027**, Springer-Verlag, pp. 207–222 (2004).

[9] Z. Chai, Z. Cao and Y. Zhou. Efficient ID-based broadcast threshold decryption in ad hoc network. *Proceedings of IMSCCS'06*, Volume 2, IEEE Computer Society, pp. 148–154 (2006).

[10] A. Fiat and M. Naor. Broadcast encryption. *Proceedings of Crypto'93*, LNCS **773**, Springer-Verlag, pp. 480–491 (1994).

[11] H. Ghodosi, J. Pieprzyk and R. Safavi-Naini. Dynamic threshold cryptosystems: a new scheme in group oriented cryptography. *Proceedings of Pragocrypt'96*, CTU Publishing house, pp. 370-379 (1996).

[12] A. Shamir. How to share a secret. *Communications of the ACM*, vol. **22**, pp. 612–613 (1979).

[13] A. Shamir. Identity-based cryptosystems and signature schemes. *Proceedings of Crypto'84*, LNCS **196**, Springer-Verlag, pp. 47–53 (1984).

[14] V. Shoup and R. Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. *Journal of Cryptology*, vol. **15** (2), Springer-Verlag, pp. 75–96 (2002).