Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 192 (2008) 3–15
www.elsevier.com/locate/entcs

Ad-Hoc Threshold Broadcast Encryption with Shorter Ciphertexts

Vanesa Daza[1]

Dept. D'Enginyeria Informàtica i Matemàtiques
Universitat Rovira i Virgili
Av. Països Catalans 26, E-43007 Tarragona, Spain

Javier Herranz[2]

IIIA, Artificial Intelligence Research Institute
CSIC, Spanish National Research Council
Campus UAB s/n, E-08193 Bellaterra, Spain

Paz Morillo, Carla Ràfols[3]

Dept. Matemàtica Aplicada IV
Universitat Politècnica de Catalunya
C. Jordi Girona 1-3, E-08034 Barcelona, Spain

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.
This kind of scheme has many applications in mobile ad-hoc networks, characterized by their lack of infrastructure as well as for the high dynamism of their nodes. Threshold broadcast encryption schemes are much more appropriate for mobile ad-hoc scenarios than standard threshold public key encryption schemes, where the set of receivers and the threshold for decryption must be known in advance (and remain the same for the rest of the protocol).
Previously proposed threshold broadcast encryption schemes have ciphertexts which contain at least n group elements. In this paper, we propose a new scheme where the ciphertexts contain essentially $n - t$ group elements. The construction uses secret sharing techniques and the ElGamal public key cryptosystem as basic tools. We formally prove the security of the scheme, by reduction to the security of ElGamal cryptosystem.

Keywords: Threshold encryption, ad-hoc and dynamic groups, ElGamal cryptosystem.

[1] Email: vanesa.daza@urv.cat
[2] Email: jherranz@iiia.csic.es
[3] Email: paz@ma4.upc.edu, crafols@ma4.upc.edu

doi:10.1016/j.entcs.2008.05.002

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 set 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 decided 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. With this motivation in mind, a scheme for this situation would have the following properties:

1. There is no setup phase or fixed 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 players 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.

1.1 Application to Mobile Ad-hoc Networks

This kind of schemes are specially useful in order to guarantee security in mobile ad-hoc networks. A mobile ad-hoc network (also known as MANET) is created on the fly. It consists of a set of self-organized and mobile nodes (*e.g.* PDAs, laptops or cellular phones) without any fixed infrastructure. The topology of a MANET changes rapidly and unpredictably over time. Such dynamism can be due to new nodes joining the network, whereas at the same time others leave it or just fail because they move to a region that is not in the cover range of the network. Standard threshold public key encryption schemes - where threshold and set of receivers are decided at the setup stage- are not always an ideal solution for this very general scenario. Imagine for example that a sender wants to send a message to a group of receivers in a mobile ad-hoc network, where a considerable number of these receivers are newcomers, that is, they have just entered into the network. To use standard threshold encryption techniques, a public key for the set of receivers and a threshold for decryption must have been set up in a protocol which is either interactive among the receivers, or executed by an external entity; both situations are typically not desirable in ad-hoc networks. Each time someone wants to encrypt

a message with a different set of receivers or a different threshold, which is likely to happen quite often in a MANET, the interactive setup protocol should be run again. This does not fit in with the computational and energy constraints of an ad-hoc network.

We emphasize that, unlike in threshold public key encryption schemes, in a threshold broadcast encryption scheme the sender can decide the set of receivers in an ad-hoc way, for example depending on the set of nodes he is directly connected with. Not only this, he can also decide the threshold of receivers which will be necessary to recover the information, for example depending on the secrecy level of the specific message he wants to send.

1.2 Related work

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 at least n times the length of a standard RSA modulus, where n is the number of receivers. In [7], the authors propose a TBE scheme for identity-based scenarios; again, the ciphertexts contain at least n group elements.

1.3 Our contribution

In this paper we propose a new threshold broadcast encryption scheme where the length of the ciphertexts is essentially $n - t$, being n the number of receivers and t the threshold for the decryption. The idea is to use some techniques related to secret sharing and combine them with some suitable public key encryption scheme. In this work we use ElGamal encryption scheme [9]. We formally prove that the proposed TBE scheme has the same security level as ElGamal cryptosystem.

The rest of the work is organized as follows. In Section 2 we recall some tools (secret sharing, public key encryption) that will be necessary for the construction of our scheme. In Section 3 we give the general definitions of the protocols of a threshold broadcast encryption scheme, along with the description of the formal security model for such schemes. We propose our scheme in Section 4, and we prove that the scheme is secure, by reduction to the security of ElGamal encryption scheme. We conclude our work in Section 5.

2 Preliminaries

2.1 Threshold Secret Sharing Schemes

The idea of *secret sharing schemes* was independently introduced by Shamir [12] and Blakley [5]. A (d, N)-threshold secret sharing scheme is a method by means of which a special figure, called usually *dealer*, distributes a secret s among a set $\mathcal{P} = \{P_1, \ldots, P_N\}$ of N players. Each player P_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 P_i is publicly and uniquely associated to a field element α_i. The dealer privately sends to player P_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:

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

where

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

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 $\{f(\alpha_k)\}_{P_k \in B}$, of course.

2.2 Standard Public Key Encryption

A public key encryption scheme PKE= (PKE.KG, PKE.Enc, PKE.Dec) consists of three algorithms:

- The probabilistic key generation algorithm PKE.KG takes as input a security parameter k and returns a pair (pk, sk) consisting of a public key pk and a matching secret key sk; we denote an execution of this protocol as $(pk, sk) \leftarrow \text{PKE.KG}(1^k)$.
- The probabilistic encryption algorithm PKE.Enc takes as input a public key pk and a message m, and returns a ciphertext C; we write $C \leftarrow \text{PKE.Enc}(pk, m)$.
- The deterministic decryption algorithm PKE.Dec takes the secret key sk and a ciphertext C as input, and outputs either the corresponding message or a special

symbol $\perp$. We write $\tilde{m} \leftarrow \text{PKE.Dec}(sk, C)$.

We recall the standard notion of security for public key encryption schemes in terms of *indistinguishability*. We consider both chosen-plaintext and chosen-ciphertext attacks. For this, we consider the following game that an attacker $\mathcal{A}_{atk}$ plays against a challenger:

$$
\begin{aligned}
&(pk, sk) \leftarrow \text{PKE.KG}(1^k)\\
&(St, m_0, m_1) \leftarrow \mathcal{A}_{atk}^{\mathcal{O}_1(\cdot)}(\text{find}, pk)\\
&b \leftarrow \{0,1\} \text{ at random; } C^* \leftarrow \text{PKE.Enc}(pk, m_b)\\
&b' \leftarrow \mathcal{A}_{atk}^{\mathcal{O}_2(\cdot)}(\text{guess}, C^*, St).
\end{aligned}
$$

In both phases (find and guess) of the attack, the attacker $\mathcal{A}_{atk}$ can have access to a decryption oracle with respect to sk for ciphertexts of his choice, depending on the considered kind of attacks. Namely, if atk is a chosen plaintext attack (CPA), then there is no access at all, which we write as $\mathcal{O}_1 = \mathcal{O}_2 = \epsilon$. If atk is a partial chosen ciphertext attack (CCA1), then $\mathcal{O}_1 = \text{PKE.Dec}(sk, \cdot)$ and $\mathcal{O}_2 = \epsilon$. Finally, if atk is a full chosen ciphertext attack (CCA2), then $\mathcal{O}_1 = \mathcal{O}_2 = \text{PKE.Dec}(sk, \cdot)$. 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

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

A public key 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.

2.3 ElGamal Public Key Encryption Scheme

ElGamal public key encryption scheme, EG_PKE= (EG_PKE.KG, EG_PKE.Enc, EG_PKE.Dec), works as follows (see [9] for the original proposal):

- EG_PKE.KG takes as input a security parameter k and generates two prime numbers p and q such that q is k bit long and $q|p-1$. Then a cyclic subgroup $\mathbb{G} = \langle g \rangle$ of $\mathbb{Z}_p$ is chosen, with order q. All these values are made public. The secret key sk of the user is chosen at random in $\mathbb{Z}_q^*$, whereas the matching public key is $pk = g^{sk} \bmod p$.
- EG_PKE.Enc takes as input a public key pk and a message $m \in \mathbb{G}$; then a random value $a \in \mathbb{Z}_q^*$ is chosen, and the ciphertext $C = (r, s)$ is computed as $r = g^a \bmod p$ and $s = m \cdot pk^a \bmod p$.
- EG_PKE.Dec takes the secret key sk and a ciphertext $C = (r, s)$, and outputs $s/r^{sk} \bmod p = m$.

ElGamal cryptosystem is known to be indistinguishable under CPA attacks, assuming the hardness of the Decisional Diffie-Hellman (DDH) problem. It is also known that this scheme is not indistinguishable under CCA2 attacks, because of its homomorphic properties. Nothing has been proved about indistinguishability of ElGamal encryption scheme under CCA1 attacks.

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 $\texttt{params} \leftarrow \text{TBE.Setup}(1^k)$.
- The randomized key generation algorithm TBE.KG takes as input some public parameters `params` and returns a pair (pk, sk) consisting of a public key and a matching secret key; we denote an execution of this protocol as $(pk, sk) \leftarrow \text{TBE.KG}(\texttt{params})$.
- The randomized encryption algorithm TBE.Enc takes as input a set of public keys $\{pk_i\}_{P_i \in \mathcal{P}}$ corresponding to a set $\mathcal{P}$ of n players, 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 $C \leftarrow \text{TBE.Enc}(\mathcal{P}, \{pk_i\}_{P_i \in \mathcal{P}}, t, m)$.
- 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 player $P_i \in \mathcal{P}$. The output is a partial decryption value $\tilde{m}_i$ or a special symbol $\perp$. We denote with $\tilde{m}_i \leftarrow \text{TBE.PartDec}(C, sk_i)$ 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 $\{\tilde{m}_i\}_{P_i \in A}$ corresponding to players in some subset $A \subset \mathcal{P}$. The output is a message m or a special symbol $\perp$. We write $\tilde{m} \leftarrow \text{TBE.Dec}(C, \{\tilde{m}_i\}_{P_i \in A}, A)$.

An important parameter of such schemes is the length of the ciphertext C. When measuring this length (in our proposal, but also in previous proposals), we do 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 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 [11,7], the ciphertexts contain at least n group elements, for some mathematical group (usually large). In this paper we will propose a new scheme where the ciphertexts contain essentially $n - t$ group elements.

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, as in the definition we provide in Section

2.2. 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 involve 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 any 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^*.

One remark to be done is how to deal with the first kind of corruptions, those where the attacker registers new public keys. 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 [3]. In the game which models the security of threshold broadcast encryption schemes, the attacker will be 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 [3], this is equivalent to require the adversary to supply also the matching secret key, each time he registers a public key. This approach has already been followed in other works [2,4].

For simplicity, when analyzing our scheme we will consider only the first kind of corruption. If both kinds of corruption are considered, the scheme can be still be proved secure, but the proof becomes a bit more confusing. Taking all this into consideration, indistinguishability for threshold broadcast encryption schemes is defined by considering the following game that an attacker $\mathcal{B}_{atk}$ plays against a challenger:

$\texttt{params} \leftarrow \text{TBE.Setup}(1^k)$
For players $P_i \in \mathcal{U}_1$, run $(pk_i, sk_i) \leftarrow \text{TBE.KG}(\texttt{params})$
$\mathcal{B}_{atk}$ can register public keys at any time: for players $P_j \in \mathcal{U}_2$, he runs $(pk_j, sk_j) \leftarrow \text{TBE.KG}(\texttt{params})$ and broadcasts (P_j, pk_j, sk_j)
$(St, \mathcal{P}^*, t^*, m_0, m_1) \leftarrow \mathcal{B}_{atk}^{\mathcal{O}_1(\cdot)}(\text{find}, \texttt{params}, \{pk_i\}_{P_i \in \mathcal{U}_1})$
$b \leftarrow \{0,1\}$ at random; $C^* \leftarrow \text{TBE.Enc}(\mathcal{P}^*, \{pk_i\}_{P_i \in \mathcal{P}^*}, t^*, m_b)$
$b' \leftarrow \mathcal{B}_{atk}^{\mathcal{O}_2(\cdot)}(\text{guess}, C^*, St)$.

Of course, in order to consider meaningful and successful such an attack, we require $|\mathcal{P}^* \cap \mathcal{U}_2| < t^*$. Otherwise, $\mathcal{B}_{atk}$ knows the secret key of at least t^* players in $\mathcal{P}^*$ and can decrypt C^* by himself, obtaining m_b.

Depending on the considered kind of attacks, $\mathcal{B}_{atk}$ can also have access to a decryption oracle for tuples $(\mathcal{P}, t, C)$ of his choice. As answer, $\mathcal{B}_{atk}$ receives all the information that would be broadcast in a complete decryption process for this tuple; this includes all the partial decryption values. 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, $\mathcal{B}_{\text{CCA2}}$ is not allowed to query the oracle $\mathcal{O}_2$ with the challenge tuple $(\mathcal{P}^*, t^*, C^*)$.

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

$$\text{Adv}(\mathcal{B}_{atk}) = \left| \Pr[b' = b] - \frac{1}{2} \right|.$$

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

4 A Threshold Broadcast Encryption Scheme with $|C| \approx n - t$

Our threshold broadcast encryption scheme is based on ElGamal. We denote it as EG_TBE = (EG_TBE.Setup, EG_TBE.KG, EG_TBE.Enc, EG_TBE.PartDec, EG_TBE.Dec), where the five algorithms work as follows.

EG_TBE.Setup. Given a security parameter k, it generates two prime numbers p and q such that q is k bits long and $q|p-1$. Then a cyclic subgroup $\mathbb{G} = \langle g \rangle$ of $\mathbb{Z}_p$ is chosen, with order q. We will use a collision-resistant hash function $h : \{0,1\}^* \to \mathbb{Z}_q^*$ which assigns one different element in $\mathbb{Z}_q^*$ to each potential player, to apply Shamir's secret sharing techniques. Therefore, the output of the protocol is $\texttt{params} = (p, q, \mathbb{G}, g, h)$.

EG_TBE.KG. Each player P_i chooses at random his secret key $sk_i \in \mathbb{Z}_q^*$. The matching public key is $pk_i = g^{sk_i} \bmod p$. (From now on in the paper, we will sometimes omit the explicit $\bmod p$, since all the operations will be performed modulo p.)

EG_TBE.Enc. In this phase of the scheme, a sender wants to encrypt a message addressed to some set of receivers that he chooses ad-hoc, in such a way that the message can be recovered only if the number of receivers that cooperate to decrypt is equal to or greater than a certain threshold that is also chosen by the sender.

In order to encrypt a message $m \in \mathbb{G}$ addressed to some set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of n players, with threshold $t \leq n$ for the decryption, the idea is to set up an (n, N)-threshold secret sharing scheme, where $N = 2n - t$. Let $\lambda_{i0}^{\mathcal{P}}$ be the Lagrange coefficients defined in Section 2.1, useful to compute the value of the polynomial in the point $\alpha_0 = 0$ from the value of the polynomial in the points $\{\alpha_i\}_{P_i \in \mathcal{P}}$, where $\alpha_i = h(P_i)$. The sender must act as follows.

(i) Compute $PK = \prod_{P_i \in \mathcal{P}} pk_i^{\lambda_{i0}^{\mathcal{P}}} \bmod p$.

(ii) Choose a set $\tilde{\mathcal{P}}$ of $n - t$ (dummy) players, such that $\tilde{\mathcal{P}} \cap \mathcal{P} = \emptyset$. For each $P_j \in \tilde{\mathcal{P}}$, compute $\alpha_j = h(P_j)$ and then define $pk_j = \prod_{P_i \in \mathcal{P}} pk_i^{\lambda_{ij}^{\mathcal{P}}} \bmod p$.

(iii) Choose at random $a \in \mathbb{Z}_q^*$ and compute $r = g^a \bmod p$.

(iv) Compute $s = m \cdot PK^a \bmod p$.

(v) For each $P_j \in \tilde{\mathcal{P}}$, compute $\tilde{m}_j = pk_j^a \bmod p$.

(vi) Define the ciphertext as $C = (\mathcal{P}, t, \tilde{\mathcal{P}}, r, s, \{\tilde{m}_j\}_{P_j \in \tilde{\mathcal{P}}})$.

Note that the sender has used ElGamal encryption scheme with a public key PK such that the corresponding secret key $SK = \log_g PK$ is not known by any single player. On the other hand, any subset of n players in $\mathcal{P} \cup \tilde{\mathcal{P}}$ could recover SK from their own secret keys sk_i, by using Lagrange interpolation, if we implicitly define $sk_j = \log_g pk_j$, for all $P_j \in \tilde{\mathcal{P}}$. In the particular case of threshold decryption, since the ciphertext C already includes $n-t$ partial decryptions, only t new partial decryptions, coming from $\mathcal{P}$, will be necessary to recover the plaintext. Note also that a ciphertext C, excluding the description of the sets $\mathcal{P}$ and $\tilde{\mathcal{P}}$, contains $n-t+2$ group elemtents (in $\mathbb{Z}_p$). The description of the set $\tilde{\mathcal{P}}$ can be actually very short; for example, the sender can look for an interval of $n-t$ integers $J = \{j_0, j_0+1, \ldots, j_0+n-t-1\}$ (modulo q) such that $\alpha_i \notin J$ for all $P_i \in \mathcal{P}$, and define the set $\tilde{\mathcal{P}}$ simply as the $n-t$ dummy users P_j verifying $\alpha_j \in J$. Note that in this case, the value j_0 is enough to describe the set $\tilde{\mathcal{P}}$. The resulting ciphertexts are then shorter than those in all the previously proposed schemes, for any value of t satisfying $2 \leq t \leq n$.

EG_TBE.PartDec. Given a ciphertext $C = (\mathcal{P}, t, \tilde{\mathcal{P}}, r, s, \{\tilde{m}_j\}_{P_j \in \tilde{\mathcal{P}}})$, any player $P_i \in \mathcal{P}$ can compute his partial decryption $\tilde{m}_i = r^{sk_i} \bmod p$.

It would be possible to add robustness techniques in order to detect invalid partial decryptions. For example, each player P_i can be required to add to his partial decryption a zero-knowledge proof that $\mathrm{DiscLog}_r(\tilde{m}_i) = \mathrm{DiscLog}_g(pk_i)$. This robust variant of our scheme, that we do not consider for the sake of simplicity, can also be proved secure.

EG_TBE.Dec. Given a ciphertext $C = (\mathcal{P}, t, \tilde{\mathcal{P}}, r, s, \{\tilde{m}_j\}_{P_j \in \tilde{\mathcal{P}}})$ and a set of t partial decryptions $\tilde{m}_i$, corresponding to a subset $A \subset \mathcal{P}$ with $|A| = t$, a combiner algorithm considers the whole set of partial decryptions, from $B = A \cup \tilde{\mathcal{P}}$, and then computes

$$\kappa = \prod_{P_i \in B} \tilde{m}_i^{\lambda_{i0}^B} \bmod p = g^{a \sum\limits_{P_i \in B} \lambda_{i0}^B sk_i} = \\ = g^{aSK} = PK^a.$$

Then the plaintext m is recovered by computing $m = s/\kappa \bmod p$.

4.1 Efficiency of the Scheme

To encrypt a message for a subset $\mathcal{P}$ of n receivers, with threshold t for decryption, the sender must perform $n + n(n-t+1) + 1$ modular exponentiations. However, there are only n bases for these exponentiations (i.e. the values pk_i, for $P_i \in \mathcal{P}$) and so pre-computation techniques can be used to speed up the final computation time of this phase.

For the partial decryption of a ciphertext, any of the t receivers involved in the decryption must locally compute a modular exponentiation. Finally, the combiner

algorithm must perform n modular exponentiations to recover the plaintext from the ciphertext and the partial decryptions.

4.2 Our Scheme is as Secure as ElGamal Cryptosystem

In this section we prove that our threshold broadcast encryption scheme is essentially as secure as ElGamal public key encryption scheme. As it happens with ElGamal, our scheme is not secure under CCA2 attacks. With respect to CPA and CCA1 attacks, we are going to show that a successful attack against our scheme implies a successful attack against ElGamal.

As we remarked in Section 3.1, we consider for simplicity only the first kind of corruption, and not the corruptions where the attacker asks for the secret key of a previously registered public key. If this second type of corruption is considered, the proof is still possible (with a slightly worse reduction factor) but becomes more cumbersome: the public keys of the users are defined in two different ways, depending on a probability distribution. For one of the ways, the matching secret key is known and so the corruption queries can be correctly answered for these users. For the other way, the secret key is not known, but a hypothetical attack involving this public key can be transformed into an attack against ElGamal cryptosystem.

Theorem 4.1 *For atk =CPA,CCA1, if there exists an attacker $\mathcal{B}_{atk}$ against our threshold broadcast encryption scheme with $Adv(\mathcal{B}_{atk}) \geq \varepsilon$, then there exists an attacker $\mathcal{A}_{atk}$ against ElGamal public key encryption scheme with $Adv(\mathcal{A}_{atk}) \geq \varepsilon/4$.*

Proof. We are going to deal with the CCA1 case, obviously the CPA case can be proved in a similar (but simpler) way. Let us assume therefore the existence of a CCA1 attacker $\mathcal{B}$ against our TBE scheme, and let us construct a CCA1 attacker $\mathcal{A}$ against ElGamal. $\mathcal{A}$ receives an ElGamal public key $(p, q, \mathbb{G}, g, y)$ as initial input.

Now $\mathcal{A}$ prepares the initialization of the hypothetical attacker $\mathcal{B}$. Namely, for each player $P_i \in \mathcal{U}_1$, $\mathcal{A}$ chooses at random $\beta_i \in \mathbb{Z}_q^*$ and computes $pk_i = y^{\beta_i} \bmod p$. Finally, $\mathcal{A}$ chooses a collision-resistant hash function $h : \{0,1\}^* \to \mathbb{Z}_q^*$. The public parameters for $\mathcal{B}$ will be $\texttt{params} = (p, q, \mathbb{G}, g, h)$. The attacker $\mathcal{A}$ initializes the attack $\mathcal{B}$ by sending him $(\texttt{params}, \{pk_i\}_{P_i \in \mathcal{P}})$.

$\mathcal{B}$ can register new public keys, for players $P_j \in \mathcal{U}_2$: he must broadcast the resulting pairs (pk_j, sk_j). Now $\mathcal{B}$ can make decryption queries, for ciphertexts $C = (\mathcal{P}, t, \tilde{P}, r, s, \{\tilde{m}_j\}_{P_j \in \tilde{\mathcal{P}}})$, where $\mathcal{P}, \tilde{\mathcal{P}} \subset \mathcal{U}_1 \cup \mathcal{U}_2$. The attacker $\mathcal{A}$ has to simulate the information that $\mathcal{B}$ would obtain in the whole decryption procedure for this ciphertext (including all the partial decryption values $\tilde{m}_i$ for $P_i \in \mathcal{P}$). To do this, remember that $\mathcal{A}$ has access to his own ElGamal decryption oracle.

For each player $P_j \in \mathcal{P} \cap \mathcal{U}_2$, the value sk_j is known so the partial decryption $\tilde{m}_j = r^{sk_j} \bmod p$ can be computed and broadcast by $\mathcal{A}$.

Note that if $r = g^a \bmod p$, we can write

$$s = m \left(\prod_{P_i \in \mathcal{P}} pk_i^{\lambda_{i0}^{\mathcal{P}}} \right)^a = m \left(\prod_{P_j \in \mathcal{P} \cap \mathcal{U}_2} g^{\lambda_{j0}^{\mathcal{P}} sk_j} \right)^a \left(\prod_{P_k \in \mathcal{P} \cap \mathcal{U}_1} y^{\lambda_{k0}^{\mathcal{P}} \beta_k} \right)^a .$$

Let us denote $M = m\left(\prod_{P_j\in\mathcal{P}\cap\mathcal{U}_2} g^{\lambda^{\mathcal{P}}_{j0}sk_j}\right)^a$, then we have $s = M\left(\prod_{P_k\in\mathcal{P}\cap\mathcal{U}_1} y^{\lambda^{\mathcal{P}}_{k0}\beta_k}\right)^a$.

The goal of $\mathcal{A}$ is now to compute the values $\tilde{m}_i = r^{sk_i} = pk_i^a \bmod p$, for players $P_i \in \mathcal{P}\cap\mathcal{U}_1$, where we have $pk_i = y^{\beta_i}$.

First of all, $\mathcal{A}$ sends to his ElGamal decryption oracle the ciphertext $(r^{\sum_{P_k\in\mathcal{P}\cap\mathcal{U}_1}\lambda^{\mathcal{P}}_{k0}\beta_k}, s)$. The plaintext that $\mathcal{A}$ obtains from his oracle is exactly M.

Then, for each player $P_i \in \mathcal{P}\cap\mathcal{U}_1$, $\mathcal{A}$ sends to his ElGamal decryption oracle the ciphertext $(r^{\sum_{P_k\in\mathcal{P}\cap\mathcal{U}_1}\lambda^{\mathcal{P}}_{k0}\beta_k\ -\lambda^{\mathcal{P}}_{i0}\beta_i}, s)$. The plaintext that $\mathcal{A}$ obtains from his oracle is $My^{a\beta_i\lambda^{\mathcal{P}}_{i0}}$. Since $\mathcal{A}$ already knows the value M, he can extract from this answer the desired value $y^{a\beta_i} = pk_i^a = \tilde{m}_i \bmod p$. So $\mathcal{A}$ can perfectly simulate the answers to the decryption queries that $\mathcal{B}$ makes.

Now $\mathcal{B}$ outputs $(\mathcal{P}^*, t^*, m_0, m_1)$, where $\mathcal{P}^*$ contains $n^* \geq t^*$ players and m_0 and m_1 are two messages with the same length. Since we assume that the attack performed by $\mathcal{B}$ is meaningful, there must be at least one non-corrupted player in $\mathcal{P}^*$; that is, $\mathcal{P}^*\cap\mathcal{U}_1 \neq \emptyset$. $\mathcal{A}$ chooses $\alpha_0, \alpha_1 \in \mathbb{G}$ at random and sends the messages $\alpha_0 \cdot m_0$ and $\alpha_1 \cdot m_1$ to his ElGamal challenger, who sends back to $\mathcal{A}$ an encryption (r, s) of message $\alpha_b \cdot m_b$ for a random bit $b \in \{0, 1\}$ that $\mathcal{A}$ has to guess. Therefore, we have $(r, s) = (g^w, \alpha_b m_b y^w)$ for some $w \in \mathbb{Z}_q^*$. At this point $\mathcal{A}$ makes a first guess by choosing at random $\tilde{b} \in \{0, 1\}$, and compute the value $z = s/\alpha_{\tilde{b}} m_{\tilde{b}}$. Note that if $\tilde{b} = b$, then we have that $z = y^w$. Anyway, $\mathcal{A}$ sets $r^* = r$ and

$$s^* = m_{\tilde{b}}\left(\prod_{P_j\in\mathcal{P}^*\cap\mathcal{U}_2}(g^w)^{\lambda^{\mathcal{P}^*}_{j0}sk_j}\right)\left(\prod_{P_i\in\mathcal{P}^*\cap\mathcal{U}_1} z^{\lambda^{\mathcal{P}^*}_{i0}\beta_i}\right).$$

Now $\mathcal{A}$ can choose a set $\tilde{\mathcal{P}}^* \subset \mathcal{U}_1\cup\mathcal{U}_2$ of $n^* - t^*$ dummy players P_ℓ, such that $\tilde{\mathcal{P}}^*\cap\mathcal{P}^* = \emptyset$, and compute their partial decryption values $\tilde{m}_\ell = (r^*)^{sk_\ell} = pk_\ell^w$. For these dummy players, we have

$$pk_\ell = \prod_{P_i\in\mathcal{P}} pk_i^{\lambda^{\mathcal{P}}_{i\ell}} = \left(\prod_{P_i\in\mathcal{P}\cap\mathcal{U}_2}(g^{sk_i})^{\lambda^{\mathcal{P}}_{i\ell}}\right)\cdot\left(\prod_{P_i\in\mathcal{P}\cap\mathcal{U}_1}(y^{\beta_i})^{\lambda^{\mathcal{P}}_{i\ell}}\right).$$

Therefore, the desired partial decryptions are computed as

$$\tilde{m}_\ell = r^{\sum_{P_i\in\mathcal{P}\cap\mathcal{U}_2}\lambda^{\mathcal{P}}_{i\ell}sk_i}\cdot z^{\sum_{P_i\in\mathcal{P}\cap\mathcal{U}_1}\lambda^{\mathcal{P}}_{i\ell}\beta_i},$$

which again are correct if $\tilde{b} = b$.

Finally, $\mathcal{A}$ sends to $\mathcal{B}$ the resulting challenge ciphertext $C^* = (\mathcal{P}^*, t^*, \tilde{P}^*, r^*, s^*, \{\tilde{m}_k\}_{P_k\in\tilde{\mathcal{P}}^*})$. The attacker $\mathcal{B}$ eventually outputs a bit b'. If $b' = \tilde{b}$, then $\mathcal{A}$ outputs this bit b'; otherwise, $\mathcal{A}$ outputs a random bit.

Let us compute the success probability of $\mathcal{A}$. Note that if his first guess $\tilde{b}$ is correct, then the challenge ciphertext C^* is consistent and by hypothesis $\mathcal{B}$ will

guess the correct bit with probability $1/2 + \varepsilon$. On the other hand, if $\tilde{b} \neq b$, then we have that $z = y^w \frac{\alpha_b m_b}{\alpha_{\tilde{b}} m_{\tilde{b}}}$ is a completely random value. Therefore, the information that $\mathcal{B}$ receives in the challenge ciphertext C^* is completely independent of the bit b; for this reason, we can assume that $\mathcal{B}$'s guess b' is correct with probability $1/2$. In this case, $\mathcal{A}$ will see $b' \neq \tilde{b}$ and will choose at random his own final guess on b, succeeding with probability $1/2$. Summing up, we have

$$\Pr[\mathcal{A} \text{ wins his game}] = \Pr[\tilde{b} = b \wedge \mathcal{A} \text{ wins his game}] + \Pr[\tilde{b} \neq b \wedge \mathcal{A} \text{ wins his game}] \geq$$

$$\frac{1}{2} \cdot \left[\left(\frac{1}{2} + \varepsilon \right) + \left(\frac{1}{2} - \varepsilon \right) \cdot \frac{1}{2} \right] + \frac{1}{2} \cdot \left[\frac{1}{2} \cdot \frac{1}{2} \right] = \frac{1}{2} + \frac{\varepsilon}{4} \, .$$

□

5 Conclusion

Threshold broadcast encryption (TBE) schemes differ from traditional threshold public key encryption schemes [13,6] 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, specially regarding mobile ad-hoc networks.

In this work we have designed a TBE scheme with shorter ciphertexts than previous proposals, and whose security level is proved to be the same as in ElGamal cryptosystem (at most CCA1 security). In a subsequent work [8], we have designed a different scheme which achieves maximum security (under CCA2 attacks) with ciphertexts which have length $n - t$, as well. However, the scheme employs bilinear pairings, which makes it less efficient. An interesting open problem is to study if the bound of $n - t$ for the length of the ciphertexts can still be lowered.

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, Lecture Notes in Computer Science **1807** (2000), 259–274.

[2] M. Bellare, A. Boldyreva and J. Staddon. *Randomness reuse in multi-recipient encryption schemes*. Proceedings of PKC'03, Lecture Notes in Computer Science **2567** (2002), 85–99.

[3] M. Bellare and O. Goldreich. *On defining proofs of knowledge*. Proceedings of Crypto'92, Lecture Notes in Computer Science **740** (1992), 390–420.

[4] M. Bellare, T. Kohno and V. Shoup. *Stateful public-key cryptosystems: how to encrypt with one 160-bit exponentiation*. Proceedings of CCS'06, ACM Press (2006), 380–389.

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

[6] R. Canetti and S. Goldwasser. *An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack*. Proceedings of Eurocrypt'99, Lecture Notes in Computer Science **1592** (1999), 90–106.

[7] 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 (2006), 148–154.

[8] V. Daza, J. Herranz, P. Morillo and C. Ràfols. *CCA2-secure threshold broadcast encryption with shorter ciphertexts*. Proceedings of ProvSec'07, Lecture Notes in Computer Science **4784** (2007), 35–50.

[9] T. ElGamal. *A public key cryptosystem and a signature scheme based on discrete logarithms*. IEEE Trans. Information Theory, **31** (1985), 469-472.

[10] A. Fiat and M. Naor. *Broadcast encryption*. Proceedings of Crypto'93, Lecture Notes in Computer Science **773** (1994), 480–491.

[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 (1996), 370–379.

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

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