Dynamic Threshold Public-Key Encryption

Cécile Delerablée[1,2] and David Pointcheval[2]

[1] Orange Labs, cecile.delerablee@orange-ftgroup.com
[2] ENS-CNRS-INRIA, david.pointcheval@ens.fr

Abstract. This paper deals with *threshold public-key encryption* which allows a pool of players to decrypt a ciphertext if a given threshold of authorized players cooperate. We generalize this primitive to the dynamic setting, where any user can *dynamically* join the system, as a possible recipient; the sender can *dynamically* choose the authorized set of recipients, for each ciphertext; and the sender can *dynamically* set the threshold t for decryption capability among the authorized set. We first give a formal security model, which includes strong robustness notions, and then we propose a candidate achieving all the above dynamic properties, that is semantically secure in the standard model, under a new non-interactive assumption, that fits into the general Diffie-Hellman exponent framework on groups with a bilinear map. It furthermore compares favorably with previous proposals, *a.k.a.* threshold broadcast encryption, since this is the first threshold public-key encryption, with dynamic authorized set of recipients and dynamic threshold that provides constant-size ciphertexts.

1 Introduction

In a threshold public-key encryption (in short, TPKE) system [6, 15, 9, 18, 12, 35], the decryption key corresponding to a public key is shared among a set of n users (or servers). In such a system, a ciphertext can be decrypted only if at least t users cooperate. Below this threshold, no information about the plaintext is leaked, which is crucial in all the applications where one cannot fully trust a unique person, but possibly a pool of individuals.

Electronic voting is a classical example for such a threshold encryption primitive: only a pool of bodies is trusted not to cooperate for decrypting individual ballots, but for opening the final result only. The scrutineers cannot be individually trusted, but they globally are, since they control each other. Key-escrow is another application where the distribution of trust is a requirement, or in identity-based cryptography for the secret key extraction [6], as well as any decryption procedure requiring a judge decision.

However, one of the main limitation of standard TPKE is that authorized sets (the public keys) and the threshold t are often fixed during the setup, or at least t is fixed during the key generation phase: the threshold is intrinsic to the public key, and thus cannot be tuned at the encryption time. Additional flexibility could be useful in many applications in order to avoid the generation of multiple keys for the same purpose, but with different properties only, such as different partners in the authorized set or different thresholds.

Related Work. To this aim, different notions were proposed, like identity-based threshold encryption (decryption) [1, 26], or threshold broadcast encryption (or dynamic threshold encryption) [11, 10, 20, 9].

The scheme proposed in [11] appears to be the closest to what we are dealing with is this paper, and the most efficient. Indeed, this is the first (threshold broadcast encryption) scheme that provides ciphertexts of length smaller than $O(s)$ (actually, this is $O(s-t)$), where s is the size of the authorized set, and t is the threshold, and they are not fixed during the setup, but at the encryption time only. They proposed a scheme in the PKI scenario, and then in the identity-based case. But they let as an open problem to find a scheme with smaller ciphertexts.

Note that such a threshold encryption primitive is close to broadcast encryption [16, 27, 24, 22, 8, 14]: A ciphertext c sent to an authorized set $\mathcal{S}$, under a threshold $t = 1$, actually allows any player in $\mathcal{S}$ to individually decrypt c. However, for a larger threshold, such a primitive does not seem related to broadcast anymore, hence the name of *(dynamic) threshold public-key encryption.* A quite recent primitive, the so-called *attribute-based encryption* [23], is also related to this threshold decryption capability, according to the number of common attributes owned by the recipient. However, the first constructions were not dynamic, since the required attributes were decided at the key-generation phase (key-policy). Ciphertext-policy for attribute-based encryption is more dynamic [3], since it allows to decide about the threshold at the encryption time. However, no join-computation is required/possible for the decryption, contrarily to the usual notion of *threshold cryptography*, where a pool of players are needed to cooperate in order to perform the private computation. For practical reasons, it is preferable when the private computation needs the cooperation of several players, but in a non-interactive way. In the following, we are thus interested in *non-interactive threshold public-key encryption systems.* The latter feature of non-interactivity is also considered as an important one in [19], which deals with dynamic threshold cryptography too, but for signatures only.

Our contributions. In this paper, to capture previous notions, we propose a generalization of *threshold public-key encryption* (TPKE) to the dynamic setting, where any user can *dynamically* join the system, as a possible recipient; the sender can *dynamically* choose the authorized set of recipients, for each ciphertext; the sender can *dynamically* set the threshold t for decryption capability among the authorized set.

We first formalize this notion, and propose a security model, which deals with all the usual notions of secrecy, but also of robustness, which is important in group-oriented protocols. For our security model we start from [11]. Then, we enhance it with algorithms able to check the validity of all the objects: first a $\mathsf{ValidateCT}$ algorithm that (publicly) checks whether a ciphertext is valid with respect to the authorized set and the threshold; and a $\mathsf{ShareVerify}$ algorithm that (publicly) checks whether the players honestly computed their partial decryptions. We then present a new scheme, which is fully dynamic, and secure in the standard model. Our scheme is the first one with constant-size ciphertexts, which answers positively to the above problem, for the non-adaptive case. For the security analysis, we introduce a new assumption. Whereas it is a non-interactive assumption and thus easily falsifiable [28], it is non-standard. Since it falls under the Boneh, Boyen and Goh [5] paradigm, in the generic group model, we can have some confidence into the actual intractability, but relying on a more standard assumption remains an interesting open problem. For the robustness, for efficiency reasons, we achieve it in the random oracle model [2].

2 Definitions

This section is dedicated to the definition of the new primitive, and the security model.

2.1 Dynamic Threshold Public-Key Encryption

Our goal is to generalize the notion of *threshold public-key encryption* to the dynamic setting, where

- any user can *dynamically* join the system (the Join algorithm), as a possible recipient,
- the sender can *dynamically* choose the authorized set $\mathcal{S}$ of recipients, for each ciphertext,
- the sender can *dynamically* set the threshold t for decryption capability among the authorized set $\mathcal{S}$.

A (robust) dynamic threshold public-key encryption scheme is a tuple of algorithms $\mathcal{DTPKE} = (\mathsf{Setup}, \mathsf{Join}, \mathsf{Encrypt}, \mathsf{ValidateCT}, \mathsf{ShareDecrypt}, \mathsf{ShareVerify}, \mathsf{Combine})$ described as follows:

$\mathsf{Setup}(\lambda)$. Takes as input a security parameter λ. It outputs a set of parameters $\mathsf{Param} = (\mathsf{MK}, \mathsf{EK}, \mathsf{DK}, \mathsf{VK}, \mathsf{CK})$, where MK is the master secret key, EK is the encryption key, DK is the decryption key, VK is the verification key, and CK the combining key. The master secret key MK is kept secret by the issuer, whereas the four other keys are public parameters.

$\mathsf{Join}(\mathsf{MK}, \mathsf{ID})$. Takes as input the master secret key MK and the identity ID of a new user who wants to join the system. It outputs the user's keys $(\mathsf{usk}, \mathsf{upk}, \mathsf{uvk})$, the private key usk, for decryption; the public key upk, for encryption; and the verification key uvk. The private key usk is privately given to the user, whereas upk and uvk are widely distributed, with an authentic link to ID.

$\mathsf{Encrypt}(\mathsf{EK}, \mathcal{S}, t, M)$. Takes as input the encryption key EK, the authorized set $\mathcal{S}$ (or the public keys upk of users lying in $\mathcal{S}$), a threshold t, and a message M to be encrypted. It outputs a ciphertext.

$\mathsf{ValidateCT}(\mathsf{EK}, \mathcal{S}, t, C)$. Takes as input the encryption key EK, the authorized set $\mathcal{S}$ (or the users' public keys upk), a threshold t, and a ciphertext C. It checks whether C is a valid ciphertext with respect to EK, $\mathcal{S}$ and t.

$\mathsf{ShareDecrypt}(\mathsf{DK}, \mathsf{ID}, \mathsf{usk}, C)$. Takes as input the decryption key DK, a user ID and his private key usk, as well as a ciphertext C. It outputs a decryption share σ or $\perp$.

$\mathsf{ShareVerify}(\mathsf{VK}, \mathsf{ID}, \mathsf{uvk}, C, \sigma)$. Takes as input the verification key VK, a user ID and his verification key uvk, as well as a ciphertext C and a decryption share σ. It checks whether σ is a valid decryption share with respect to uvk. This algorihtm is crucial if robustness is required.

$\mathsf{Combine}(\mathsf{CK}, \mathcal{S}, t, C, T, \Sigma)$. Takes as input the combining key CK, a ciphertext C, a subset $T \subseteq \mathcal{S}$ of t authorized users, and $\Sigma = (\sigma_1, \ldots, \sigma_t)$ a list of t decryption shares. It outputs a cleartext M or $\perp$.

Remark 1. As already explained, for practical efficiency, we focus on non-interactive $\mathsf{ShareDecrypt}$ algorithms, and public $\mathsf{Combine}$ algorithm.

2.2 Key Encapsulation Method

For content distribution, or any encryption of a large plaintext, the by-now classical technique is the KEM-DEM methodology [34], where an ephemeral secret key is first generated, and used with an appropriate symmetric mechanism to encrypt the data. In such a case, we modify the above algorithms:

$\mathsf{Encrypt}(\mathsf{EK}, \mathcal{S}, t)$. Takes as input the encryption key EK, the authorized set $\mathcal{S}$ (or the users' public keys upk) and a threshold t. It outputs an ephemeral key K, and the key encapsulation material, called the header Hdr. The key K will be later used with the message to be encrypted with the DEM;

The header Hdr is thus the encryption of the ephemeral key, whereas the full header will denote the concatenation of the header and the authorized set $\mathcal{S}$, with the threshold t. The ciphertext will denote the concatenation of all the data: the full header and the DEM part (the data encrypted with the ephemeral key).

$\mathsf{ValidateCT}(\mathsf{EK}, \mathcal{S}, t, \mathsf{Hdr})$. Takes as input the encryption key EK and a full header $(\mathcal{S}, t, \mathsf{Hdr})$. It checks whether Hdr is a valid header with respect to EK, $\mathcal{S}$ and t.

In $\mathsf{ShareDecrypt}$ and $\mathsf{ShareVerify}$, the header Hdr only is given, instead of C;

$\mathsf{Combine}(\mathsf{CK}, \mathcal{S}, t, \mathsf{Hdr}, T, \Sigma)$. Takes as input the combining key CK, a full header $(\mathcal{S}, t, \mathsf{Hdr})$, a subset T of t authorized users in $\mathcal{S}$, and $\Sigma = (\sigma_1, \ldots, \sigma_t)$ a list of t decryption shares. It outputs the ephemeral key K or $\perp$. The key K will be later used with the ciphertext to be decrypted with the DEM;

In the following, we thus focus on this KEM-DEM methodology, and thus use the header Hdr only.

2.3 Security Model

Such a scheme must satisfy the following informal properties. They will be formalized later.

Correctness. For any header Hdr associated to an ephemeral key K during an encryption with a set $\mathcal{S}$ of registered users and a threshold t, if t users from this authorized set correctly produced the partial decryptions σ_i,
- the $\mathsf{ShareVerify}$ algorithm on any $(\mathsf{VK}, \mathsf{ID}_i, \mathsf{uvk}_i, C, \sigma_i)$ accepts;
- the $\mathsf{Combine}$ algorithm on set $\Sigma = \{\sigma_i, i = 1, \ldots, t\}$ outputs K;

Robustness. For any header Hdr associated to an ephemeral key K during an encryption with a set $\mathcal{S}$ of registered users and a threshold t, if t users (assumed to be) from this authorized set produce partial decryptions σ_i that are all accepted by the $\mathsf{ShareVerify}$ algorithm, then the $\mathsf{Combine}$ algorithm outputs K;

Privacy. For any header Hdr encrypted for a set $\mathcal{S}$ of registered users with a threshold t, any collusion that contains less than t users from this authorized set cannot learn any information about the ephemeral key.

Following [35] and [6], we can more formally define the above privacy notion, under the classical semantic-security notion [21], under various attacks [29, 31], using a game between an adversary $\mathcal{A}$ and a challenger. Both the adversary and the challenger are given as input a security parameter λ.

Setup: The challenger runs $\mathsf{Setup}(\lambda)$ to obtain the set of parameters $\mathsf{Param} = (\mathsf{MK}, \mathsf{EK}, \mathsf{DK}, \mathsf{VK}, \mathsf{CK})$. The public parameters $(\mathsf{EK}, \mathsf{DK}, \mathsf{VK}, \mathsf{CK})$ are given to the adversary.

Query phase 1: The adversary $\mathcal{A}$ adaptively issues queries:
- Join query, on input an ID: The challenger runs the Join algorithm on input $(\mathsf{MK}, \mathsf{ID})$, to create a new user in the system.
- $\mathsf{Corrupt}$ query, on input an identity ID: The challenger forwards the corresponding private key to the adversary.
- $\mathsf{ShareDecrypt}$ query, on input an ID and a header Hdr: The challenger runs the $\mathsf{ShareDecrypt}$ algorithm on Hdr, using the corresponding secret keys, and forwards the resulting partial decryption to the adversary.

Challenge: $\mathcal{A}$ outputs a target set of users $\mathcal{S}^\star$ and a threshold $t^\star$. The challenger randomly selects $b \leftarrow \{0, 1\}$ and runs algorithm $\mathsf{Encrypt}$ to obtain $(K_0, \mathsf{Hdr}^\star) = \mathsf{Encrypt}(\mathsf{EK}, \mathcal{S}^\star, t^\star)$, and randomly chooses an ephemeral key K_1. The challenger then returns $(K_b, \mathsf{Hdr}^\star)$ to $\mathcal{A}$.
There is the natural constraint that $\mathcal{S}^\star$ contains at most $t^\star - 1$ corrupted ID's.

Query phase 2: The adversary continues to adaptively issue Join, $\mathsf{Corrupt}$ and $\mathsf{ShareDecrypt}$ queries, as in phase 1, but with the constraint that the number of identities ID such that $\mathsf{Corrupt}(\mathsf{ID})$ or $\mathsf{ShareDecrypt}(\mathsf{ID}, \mathsf{Hdr}^\star)$ queries have been asked is less than $t^\star - 1$.

Guess: Finally, the adversary $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ and wins the game if $b = b'$.

The advantage of $\mathcal{A}$ is defined as $\mathsf{Adv}^{\mathsf{cca}}_{\mathcal{A}}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$. As usual, we denote by $\mathsf{Adv}^{\mathsf{cca}}_{T,n,s,t,q_C,q_D}(\lambda)$ the maximal value of $\mathsf{Adv}^{\mathsf{cca}}_{\mathcal{A}}(\lambda)$, over the adversaries $\mathcal{A}$ that run within time T, and where n, s, t, q_C and q_D are upper-bounds for the numbers of Join-queries, the size of $\mathcal{S}^\star$, $t^\star$, the number of $\mathsf{Corrupt}$ and $\mathsf{ShareDecrypt}$ queries respectively.

Non-Adaptive Adversary (NAA). We can restrict the adversary to decide before the setup which set $\mathcal{S}^\star$ as well as the threshold $t^\star$ will be sent to the challenger.

Non-Adaptive Corruption (NAC). We can also restrict the adversary to decide before the setup which identities will be corrupted.

Chosen-Plaintext Adversary (CPA). As usual, we can also prevent the adversary from issuing share decryption queries ($q_D = 0$).

Of course, the more adaptive the adversary is in the security analysis, the more secure the scheme is. But as a first step, in the following, we will focus on a basic security level:

Definition 2. (n, s, t, q_C)**-IND-NAA-NAC-CPA security** *(non-adaptive adversary, non-adaptive corruption, chosen-plaintext attacks). At initialization time, the attacker outputs the set $\mathcal{S}^\star$ of size s and a set $\mathcal{C}$ of identities that it wants to corrupt, of size q_C. The threshold $t^\star$ is set to t. Then the attacker does not have access to the ShareDecrypt-oracle.*

2.4 Extensions

Our threshold public-key encryption definition can be extended in various ways: first, in the ID-based setting, and then with improved access structures.

ID-Based Threshold Encryption. In the ID-based setting, the Join algorithm is replaced by the $\mathsf{Extract}$ algorithm that just generates the user's decryption key from the identity, in a similar way as done in [13], and let the public key and the verification key to be this identity.

More General Access Structure. To any identity ID, one can virtually associate several sub-identities $\mathsf{ID}\|1, \ldots, \mathsf{ID}\|k$, and then derive several sets of keys $(\mathsf{usk}_1, \mathsf{upk}_1, \mathsf{uvk}_1), \ldots, (\mathsf{usk}_k, \mathsf{upk}_k, \mathsf{uvk}_k)$. By including several sub-identities of the same user in an authorized set $\mathcal{S}$, one can give different weights for each user in the decryption capability.

This description is quite general and covers all the classical cases, but also quite sophisticated access structures, according to the way the private keys of the sub-identities are distributed:

– If the private keys of a given identity are all given to the same party, by including several sub-ID of the same party in the set $\mathcal{S}$ one gives a bigger weight to this party (and even the possibility for him to decrypt alone, whereas two other parties need to cooperate, etc).

– If the private keys (related to the sub-identities of ID) are distributed to distinct users, a threshold among these users will be needed to decrypt a message sent to ID (implicitly using $\mathcal{S} = (\mathsf{ID}\|1, \ldots, \mathsf{ID}\|w)$). This is the classical t-out-of-n threshold decryption scheme.

3 Computational Assumptions

Our construction will make use of groups with bilinear maps [25, 7], and a new computational assumption, that fits into the general Diffie-Hellman exponent framework proposed by Boneh, Boyen and Goh [5]. This framework does not provide a definite answer about the real intractability, but is a starting point for getting confidence.

3.1 Bilinear Maps

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be three cyclic groups of prime order p. A bilinear map $e(\cdot,\cdot)$ is a map $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ such that for any generators $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$,

– $e\left({g_1}^a, {g_2}^b\right) = e\left(g_1, g_2\right)^{ab}$ (Bilinearity)
– $e\left(g_1, g_2\right) \neq 1$ (Non-degeneracy).

A bilinear map group system is a tuple $\mathcal{B} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e(\cdot,\cdot))$, composed of objects as described above. $\mathcal{B}$ may also include a group generator. We impose all group operations as well as the bilinear map $e(\cdot,\cdot)$ to be efficiently computable, i.e. in time $\mathsf{poly}(|p|)$.

Note that our construction just makes use of an arbitrary bilinear map group system, without any particular additional property. In particular, we do not need $\mathbb{G}_1$ and $\mathbb{G}_2$ to be distinct or equal. Neither do we require the existence of an efficient isomorphism going either way between $\mathbb{G}_1$ and $\mathbb{G}_2$, as it is the case for some pairing-based systems.

3.2 The Multi-Sequence of Exponents Diffie-Hellman Assumption

As in [14], our security proof uses the general Diffie-Hellman exponent theorem due to Boneh, Boyen and Goh [5]. They indeed introduced a class of assumptions which includes a lot of (by-now familiar) assumptions, that appeared in the past with new pairing-based schemes. It includes for example DDH (in $\mathbb{G}_T$), BDH, $q-$BDHI, and $q-$BDHE assumptions. Even if group systems equipped with bilinear maps are far from being generic, an intractability result in this framework is a first step for getting some confidence in the actual intractability. In our case, we assume the intractability of the following decisional problem (ℓ, m, t)-$\mathsf{MSE\text{–}DDH}$:

Let $\mathcal{B} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e(\cdot,\cdot))$ be a bilinear map group system and let ℓ, m and t be three integers. Let g_0 be a generator of $\mathbb{G}_1$ and h_0 a generator of

$\mathbb{G}_2$. Given two random coprime polynomials f and g, of respective orders ℓ and m, with pairwise distinct roots $x_1, \ldots, x_\ell$ and $y_1, \ldots, y_m$ respectively, as well as several sequences of exponentiations

$$\begin{array}{ll} x_1, \ldots, x_\ell, & y_1, \ldots, y_m \\ g_0, g_0^{\gamma}, \ldots, g_0^{\gamma^{\ell+t-2}}, & g_0^{k\cdot\gamma\cdot f(\gamma)}, \\ g_0^{\alpha}, g_0^{\alpha\cdot\gamma}, \ldots, g_0^{\alpha\cdot\gamma^{\ell+t}}, & \\ h_0, h_0^{\gamma}, \ldots, h_0^{\gamma^{m-2}}, & \\ h_0^{\alpha}, h_0^{\alpha\cdot\gamma}, \ldots, h_0^{\alpha\cdot\gamma^{2m-1}}, & h_0^{k\cdot g(\gamma)}, \end{array}$$

and $T \in \mathbb{G}_T$, decide whether T is equal to $e(g_0, h_0)^{k\cdot f(\gamma)}$ or to some random element of $\mathbb{G}_T$.

The following statement is a corollary of Theorem 7 [5] which can be found in section 6. It provides an intractability bound in the generic model [33], but in groups equipped with pairings. We emphasize on the fact that, whereas the assumption has several parameters, it is non-interactive, and thus easily falsifiable [28].

Corollary 3 (Generic Security). *For any probabilistic algorithm $\mathcal{A}$ that totalizes of at most q_G queries to the oracles performing the group operations in $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ and the bilinear map $e(\cdot,\cdot)$,*

$$\mathsf{Adv}^{\mathsf{mse\text{-}ddh}}(\ell, m, t, \mathcal{A}) \leq \frac{(q_G + 4(\ell+t) + 6m + 4)^2 \cdot d}{2p}$$

with $d = 4(\ell + t) + 6m + 2$.

4 Our Construction

4.1 Description

In this section we present our new dynamic threshold public-key encryption ($\mathcal{DTPKE}$), with constant size ciphertexts. Basically, the encryption algorithm specifies the authorized-user set with an inclusion technique as in the broadcast encryption schemes [8, 13]. Moreover this authorized set is combined with a set of dummy users, in order to be consistent with the value of the threshold (this is a well-known technique in threshold encryption). We make use of the $\mathsf{Aggregate}$ algorithm (over $\mathbb{G}_T$) described in [14] to combine the decryption shares. The $\mathsf{Aggregate}$ algorithm simply exploits the fact that a product of inverses of coprime polynomials can be written as a sum of inverses of affine polynomials. Thus given some elements in $\mathbb{G}_T$ of the right form, one can combine the exponents using some group operations. We provide below a description of the case which interests us and refer to [14] for more details.

$\mathsf{Setup}(\lambda)$. Given the security parameter λ, a system with groups and a bilinear map $\mathcal{B} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e(\cdot,\cdot))$ is constructed such that $|p| = \lambda$. Also, two generators $g \in \mathbb{G}_1$ and $h \in \mathbb{G}_2$ are randomly selected as well as two secret values γ and $\alpha \in \mathbb{Z}_p^\star$. Finally, a set $\mathcal{D} = \{d_i\}_{i=1}^{m-1}$ of values in $\mathbb{Z}_p$ is randomly selected, where m is the maximal size of an authorized set. This corresponds to a set of dummy users, that will be used to complete a set of authorized users.

$\mathcal{B}$ constitutes the system parameters. The master secret key is defined as $\mathsf{MK} = (g, \gamma, \alpha)$. The encryption key is $\mathsf{EK} = \left(m, u, v, h^\alpha, \{h^{\alpha \cdot \gamma^i}\}_{i=1}^{2m-1}, \mathcal{D}\right)$, and the combining key is $\mathsf{CK} = \left(m, h, \{h^{\gamma^i}\}_{i=1}^{m-2}, \mathcal{D}\right)$, where $u = g^{\alpha \cdot \gamma}$, and $v = e\,(g,h)^\alpha$. In the following, we denote by $\mathcal{D}_i$ the i first elements of $\mathcal{D}$. Note that $\mathsf{DK} = \emptyset$, since no general data are needed for partial decryption. Furthermore, this version of the scheme does not provide robustness, we thus do not define VK yet. Robustness will be studied later.

$\mathsf{Join}(\mathsf{MK}, \mathsf{ID})$. Given $\mathsf{MK} = (g, \gamma, \alpha)$, and an identity ID, it randomly chooses $x \in \mathbb{Z}_p^\star$ (different from all previous ones, included dummy users data in $\mathcal{D}$), and outputs the user's keys $(\mathsf{usk}, \mathsf{upk})$ with:

$$\mathsf{upk} = x\ , \quad \mathsf{usk} = g^{\frac{1}{\gamma + x}}\ .$$

The private key usk is privately given to the user, whereas upk is widely published, in an authentic way (again, since robustness is not dealt with here, we do not set uvk yet).

$\mathsf{Encrypt}(\mathsf{EK}, \mathcal{S}, t)$. Given the encryption key EK, a set $\mathcal{S}$ of users, which is identified to $\mathcal{S} = \{\mathsf{upk}_1 = x_1, \ldots, \mathsf{upk}_s = x_s\}$ and a threshold t (with $t \leq s = |\mathcal{S}| \leq m$), $\mathsf{Encrypt}$ randomly picks $k \in \mathbb{Z}_p^\star$, and computes $\mathsf{Hdr} = (C_1, C_2)$ and K, where

$$C_1 = u^{-k}\ , \qquad C_2 = h^{k \cdot \alpha \cdot \prod_{x_i \in \mathcal{S}} (\gamma + x_i) \cdot \prod_{x \in \mathcal{D}_{m+t-s-1}} (\gamma + x)}\ , \qquad K = v^k\ .$$

$\mathsf{Encrypt}$ then outputs the full header $(\mathcal{S}, t, \mathsf{Hdr} = (C_1, C_2))$ and the secret key K, which will be used to encrypt the message. The crucial point is that $\mathsf{Encrypt}$ includes a set of $m + t - s - 1$ dummy users, in order to obtain a polynomial of degree exactly $m + t - 1$ in the exponent of h. This way, exploiting the cooperation of t authorized users together with a combining key that contains $\left(h, \{h^{\gamma^i}\}_{i=1}^{m-2}\right)$ is sufficient to decrypt a ciphertext (see the $\mathsf{Combine}$ algorithm).

$\mathsf{ValidateCT}(\mathsf{EK}, \mathcal{S}, t, \mathsf{Hdr})$. Given the encryption key EK and a full header $(\mathcal{S}, t)$ and $\mathsf{Hdr} = (C_1, C_2))$, as above, one can compute

$$C_1' = u^{-1}\ , \qquad C_2' = h^{\alpha \cdot \prod_{x \in \mathcal{S} \cup \mathcal{D}_{m+t-s-1}} (\gamma + x)}.$$

One should notice that a header $\mathsf{Hdr} = (C_1, C_2)$ is valid with respect to $\mathcal{S}$ if and only if there exists a scalar k such that $C_1 = {C_1'}^k$ and $C_2 = {C_2'}^k$. Moreover, one can note that in such a header, a correct $\mathcal{S}$ contains at least t keys of some users. As a consequence, $\mathsf{ValidateCT}$ simply checks whether $e\,(C_1, C_2') = e\,(C_1', C_2)$ and $\mathcal{S}$ is correct, or not.

$\mathsf{ShareDecrypt}(\mathsf{ID}, \mathsf{usk}, \mathsf{Hdr})$. In order to retrieve a share σ of a decryption key encapsulated in the header $\mathsf{Hdr} = (C_1, C_2)$, user with identity ID and the corresponding public key upk and private key $\mathsf{usk} = g^{\frac{1}{\gamma+x}}$ computes

$$\sigma = e\left(\mathsf{usk}, C_2\right) = e\left(g, h\right)^{\frac{k\cdot\alpha\cdot\prod_{x_i\in\mathcal{S}\cup\mathcal{D}_{m+t-s-1}}(\gamma+x_i)}{\gamma+x}} .$$

$\mathsf{Combine}(\mathsf{CK}, C, T, \Sigma)$. Given $\mathcal{S}$, t, $\mathsf{Hdr} = (C_1, C_2)$, CK, a subset T of t users ($T \subseteq \mathcal{S}$) and Σ the corresponding decryption shares, outputs

$$K = \left(e\left(C_1, h^{p_{(T,\mathcal{S})}(\gamma)}\right)\cdot \mathsf{Aggregate}(\mathbb{G}_T, \Sigma)\right)^{\frac{1}{c_{(T,\mathcal{S})}}} ,$$

with $c_{(T,\mathcal{S})}$ a constant in $\mathbb{Z}_p$ and $p_{(T,\mathcal{S})}$ a polynomial of degree $m-2$, that both allow to cancel a part corresponding to the $m-1$ decryption shares (over $m+t-1$) that are not in the input. Note that since $p_{(T,\mathcal{S})}$ is of degree $m-2$, $h^{p_{(T,\mathcal{S})}(\gamma)}$ is computable from CK. More precisely, we have:

$$\begin{aligned}
p_{(T,\mathcal{S})}(\gamma) &= \frac{1}{\gamma}\cdot\left(\prod_{x\in\mathcal{S}\cup\mathcal{D}_{m+t-s-1}-T}(\gamma+x) - c_{(T,\mathcal{S})}\right) ,\\
c_{(T,\mathcal{S})} &= \prod_{x\in\mathcal{S}\cup\mathcal{D}_{m+t-s-1}-T} x ,\\
\mathsf{Aggregate}(\mathbb{G}_T, \Sigma) &= \mathsf{Aggregate}\left(\mathbb{G}_T, \left\{e\left(g, C_2\right)^{\frac{1}{\gamma+x}}\right\}_{x\in T}\right)\\
&= e\left(g, C_2\right)^{\frac{1}{\prod_{x\in T}(\gamma+x)}}\\
&= e\left(g, h\right)^{k\cdot\alpha\cdot\prod_{x_i\in\mathcal{S}\cup\mathcal{D}_{m+t-s-1}-T}(\gamma+x_i)}
\end{aligned}$$

Correctness. Assuming C is well-formed, and Σ is correct:

$$\begin{aligned}
K' &= e\left(C_1, h^{p_{(T,\mathcal{S})}(\gamma)}\right)\cdot\mathsf{Aggregate}(\mathbb{G}_T, \Sigma)\\
&= e\left(g^{-k\cdot\alpha\cdot\gamma}, h^{p_{(T,\mathcal{S})}(\gamma)}\right)\cdot e\left(g, C_2\right)^{\frac{1}{\prod_{x\in T}(\gamma+x)}}\\
&= e\left(g, h\right)^{-k\cdot\alpha\cdot\gamma\cdot{}_{(T,\mathcal{S})}(\gamma)}\cdot e\left(g, h\right)^{k\cdot\alpha\cdot\prod_{x\in\mathcal{S}\cup\mathcal{D}_{m+t-s-1}-T}(\gamma+x)}\\
&= e\left(g, h\right)^{k\cdot\alpha\cdot c_{(T,\mathcal{S})}} = K^{c_{(T,\mathcal{S})}}.
\end{aligned}$$

Thus ${K'}^{\frac{1}{c_{(T,\mathcal{S})}}} = K$.

Efficiency. In our construction, ciphertexts remain constant (plus the authorized set $\mathcal{S}$ that contains the x_i's of the authorized users only, which is unavoidable and thus optimal). Moreover, our $\mathsf{Encrypt}$ algorithm is very efficient, since it does not need any pairing computation, whereas in [11], $3(s-t)$ pairing computations are needed, with s the size of the authorized set. Furthermore, any additional encryption for the same target set only require 3 exponentiations.

4.2 Aggregation of 1-degree terms: Aggregate

The Combine algorithm requires the computation of

$$L = e(g, C_2)^{\frac{1}{(\gamma+x_1)\cdots(\gamma+x_t)}} \in \mathbb{G}_T$$

given $\Sigma = \{\sigma_j = e(g, C_2)^{\frac{1}{\gamma+x_j}}\}_{j=1}^{t}$ where the x_j's are pairwise distinct. We recall how $\mathsf{Aggregate}(\mathbb{G}_T, \cdots)$ allows to compute L from the x_j's and the σ_j's, as described in [14].

Description. Given $x_1, \ldots, x_t$ and σ_j for $1 \leq j \leq t$, let us define for any (j, ℓ) such that $1 \leq j < \ell \leq r$,

$$L_{j,\ell} = \sigma_\ell^{\frac{1}{\prod_{\kappa=1}^{j}(\gamma+x_\kappa)}} = e(g, C_2)^{\frac{1}{(\gamma+x_\ell)} \cdot \frac{1}{\prod_{\kappa=1}^{j}(\gamma+x_\kappa)}} .$$

The Aggregate algorithm consists in computing sequentially $L_{j,\ell}$ for $j = 1, \ldots, t-1$ and $\ell = j+1, \ldots, t$ using the induction

$$L_{j,\ell} = \left(\frac{L_{j-1,j}}{L_{j-1,\ell}}\right)^{\frac{1}{x_\ell - x_j}}$$

and posing $L_{0,\ell} = \sigma_\ell$ for $\ell = 1, \ldots, t$. The algorithm finally outputs $L_t = L_{t-1,t}$.

4.3 Security Analysis

This section is devoted to the proof of the IND-NAA-NAC-CPA security level for our system, under our new MSE–DDH assumption.

Security Result. Let $\mathcal{DTPKE}$ denote our construction, described above, Section 4.1. We can state the following security result.

Theorem 4. *For any ℓ, m, t, $\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{DTPKE}}(\ell, m, t) \leq 2 \cdot \mathsf{Adv}^{\mathsf{mse-ddh}}(\ell, m, t)$.*

The rest of this section is dedicated to proving Theorem 4. To establish the semantic security of $\mathcal{DTPKE}$ against static adversaries, we assume an adversary $\mathcal{A}$ that breaks the scheme under an (ℓ, m, t)-collusion and we build an algorithm $\mathcal{R}$ that distinguishes the two distributions of the (ℓ, m, t)-MSE–DDH problem.

Both the adversary and the challenger are given as input m, the maximal size of a set of authorized users $\mathcal{S}$, ℓ the total number of Join queries that can be issued by the adversary, and a threshold t.

Algorithm $\mathcal{R}$ is given as input a group system $\mathcal{B} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e(\cdot,\cdot))$, and an (ℓ, m, t)-MSE–DDH instance in $\mathcal{B}$ (as described in Section 3.2). We thus have two coprime polynomials f and g, of respective orders ℓ and m, with their pairwise distinct roots $(x_1, \ldots, x_\ell)$ and $(x_{\ell+t}, \ldots, x_{\ell+t+m-1})$, and $\mathcal{R}$ is furthermore given

$$\begin{array}{ll}
g_0, g_0^{\gamma}, \ldots, g_0^{\gamma^{\ell+t-2}}, & g_0^{k\cdot\gamma\cdot f(\gamma)}, \\
g_0^{\alpha}, g_0^{\alpha\cdot\gamma}, \ldots, g_0^{\alpha\cdot\gamma^{\ell+t}}, & \\
h_0, h_0^{\gamma}, \ldots, h_0^{\gamma^{m-2}}, & \\
h_0^{\alpha}, h_0^{\alpha\cdot\gamma}, \ldots, h_0^{\alpha\cdot\gamma^{2m-1}}, & h_0^{k\cdot g(\gamma)},
\end{array}$$

as well as $T \in \mathbb{G}_T$ which is either equal to $e(g_0, h_0)^{k \cdot f(\gamma)}$ or to some random element of $\mathbb{G}_T$. For the sake of simplicity, we state that f and g are unitary polynomials, but this is not a mandatory requirement:

$$f(X) = \prod_{i=1}^{\ell}(X + x_i), \quad q(X) = \prod_{i=\ell+1}^{\ell+t-1}(X + x_i), \quad g(X) = \prod_{i=\ell+t}^{\ell+t+m-1}(X + x_i).$$

The polynomial f corresponds to a set of ℓ users not in the target set, that can be corrupted. The polynomial q corresponds to a set of $t-1$ users of the target set that can be corrupted. The polynomial g corresponds to the m users of the target set that cannot be corrupted. We will thus be able to simulate $\ell + t - 1$ decryption keys (corruptions), with $t-1$ of them, only, in the target set.

For $i \in [1, \ell + t - 1]$, we set

$$f_i(x) = \frac{f(x) \cdot q(\gamma)}{x + x_i},$$

which is a polynomial of degree $\ell + t - 2$.

Init: The adversary $\mathcal{A}$ outputs a set $\mathcal{S}^\star = \{\mathsf{ID}_1^\star, \ldots, \mathsf{ID}_{s^\star}^\star\}$ of identities that he wants to attack (the target authorized set), and a set $\overline{\mathcal{C}} = \{\overline{\mathsf{ID}}_1, \ldots, \overline{\mathsf{ID}}_c\}$ of identities that he wants to corrupt, with $c \leq \ell$ and $|\mathcal{S}^\star \cap \overline{\mathcal{C}}| \leq t - 1$;

Setup: To generate the system parameters, $\mathcal{R}$ formally sets $g = {g_0}^{f(\gamma)\cdot q(\gamma)}$ (but without computing it, since it does not need to publish it) and sets

$$h = h_0 \ , \qquad u = {g_0}^{\alpha\cdot\gamma\cdot f(\gamma)\cdot q(\gamma)} = g^{\alpha\cdot\gamma} \ ,$$

$$v = e(g_0, h_0)^{\alpha\cdot f(\gamma)\cdot q(\gamma)} = e(g, h)^{\alpha} \ .$$

The two latter formulae can be computed from the instance input, since $f \cdot q$ is of degree $\ell + t - 1$;

$\mathcal{R}$ then sets the set $\mathcal{D} = \{d_i\}_{i=1}^{m-1}$ corresponding to dummy users:

- $\mathcal{D}_{m+t-s^\star-1} = \{d_i\}_{i=1}^{m+t-s^\star-1}$ is a subset of $\{x_j\}_{j=\ell+t}^{\ell+t+m-1}$. This subset corresponds to the dummy users included to complete the target set in the challenge.
- $\{d_i\}_{i=m+t-s^\star}^{m-1}$ is a set of random elements in $\mathbb{Z}_p$

Finally, $\mathcal{R}$ defines the encryption key as $\mathsf{EK} = \left(m, u, v, h^{\alpha}, \{h^{\alpha\cdot\gamma^i}\}_{i=1}^{2m-1}, \mathcal{D}\right)$, and the combining key as $\mathsf{CK} = \left(h, \{h^{\gamma^i}\}_{i=1}^{m-2}, \mathcal{D}\right)$. Note that $\mathcal{R}$ can by no means compute the value of g. But we do not need it.

Generation of users' keys:

- For each $\overline{\mathsf{ID}} \in \overline{\mathcal{C}}$, $\mathcal{R}$ computes and sends $(\overline{\mathsf{usk}}, \overline{\mathsf{upk}})$ to $\mathcal{A}$ with

$$\overline{\mathsf{upk}} = x_i \ ,$$
$$\overline{\mathsf{usk}} = {g_0}^{f_i(\gamma)} = g^{\frac{1}{\gamma + x_i}} \ ,$$

with the following constraint: if $\mathsf{ID} \in \mathcal{S}^\star$, then the corresponding x_i must be taken in $\{x_j\}_{j=\ell+1}^{\ell+t-1}$. Otherwise x_i must be taken in $\{x_j\}_{j=1}^{\ell}$

– For each $\mathsf{ID} \in \mathcal{S}^\star - \mathcal{S}^\star \cap \overline{\mathcal{C}}$, $\mathcal{R}$ sends $\mathsf{upk} = x_i$ to $\mathcal{A}$, with the following constraint: x_i must be taken in $\{x_j\}_{j=\ell+t}^{\ell+t+m-1} - \mathcal{D}_{m+t-s^\star-1}$.
– For each $\mathsf{ID} \notin \mathcal{S}^\star \cup \overline{\mathcal{C}}$, $\mathcal{R}$ sends $\mathsf{upk} = x$ to $\mathcal{A}$, with $x \notin \{x_j\}_{j=1}^{\ell+t+m-1}$.

$\mathcal{R}$ runs $\mathcal{A}$ on the system parameters $\mathcal{B}$ and $(\mathsf{EK}, \mathsf{CK})$, and on the target set $\mathcal{S}^\star$.

Challenge: Algorithm $\mathcal{R}$ computes $\mathsf{Encrypt}$ to obtain

$$(\mathsf{Hdr}^\star, \mathcal{S}^\star, t, K) = \mathsf{Encrypt}(\mathsf{Param}, \mathsf{EK}, \mathcal{S}^\star, t), \text{ with}$$

$$C_1 = {g_0}^{-k\cdot\gamma\cdot f(\gamma)}\ , \qquad C_2 = {h_0}^{k\cdot g(\gamma)}\ , \qquad K = T\ ,$$
$$|\mathcal{S}| = s^\star\ , \qquad \mathcal{S}^\star \subseteq \{x_i\}_{i=\ell+1}^{\ell+m+t-1} - \mathcal{D}_{m+t-s^\star-1}\ .$$

One can verify that, if we set $k' := \frac{k}{\alpha\cdot q(\gamma)}$, then

$$C_1 = u^{-k'}\ , \quad C_2 = h^{k'\cdot\alpha\cdot\prod_{x_i\in\mathcal{S}^\star}(\gamma+x_i)\cdot\prod_{x\in\mathcal{D}_{m+t-s^\star-1}}(\gamma+x)}\ .$$

Note that if $T = e\left(g_0, h_0\right)^{k\cdot f(\gamma)}$, then $K = v^{k'}$.

The challenger then randomly selects $b \leftarrow \{0,1\}$, sets $K_b = K$, and sets K_{1-b} to a random value in $\mathcal{K}$. The challenger returns $(\mathsf{Hdr}^\star, K_0, K_1)$ to $\mathcal{A}$.

Guess: Finally, the adversary $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$ and wins the game if $b = b'$.

One has

$$\begin{aligned}\mathsf{Adv}^{\mathsf{mse\text{-}ddh}}(\mathcal{R}) &= \Pr[b' = b|\mathsf{real}] - \Pr[b' = b|\mathsf{random}]\\ &= \frac{1}{2} \times (\Pr[b' = 1|b = 1 \wedge \mathsf{real}] - \Pr[b' = 1|b = 0 \wedge \mathsf{real}])\\ &- \frac{1}{2} \times (\Pr[b' = 1|b = 1 \wedge \mathsf{random}] + \Pr[b' = 1|b = 0 \wedge \mathsf{random}])\ .\end{aligned}$$

Now in the random case, the distribution of b is independent from the adversary's view wherefrom

$$\Pr[b' = 1|b = 1 \wedge \mathsf{random}] = \Pr[b' = 1|b = 0 \wedge \mathsf{random}]\ .$$

In the real case however, the distributions of all variables defined by $\mathcal{R}$ perfectly comply with the semantic security game since all simulations are perfect. Therefore

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{DTPKE}}(\mathcal{A}) = \Pr[b' = 1|b = 1 \wedge \mathsf{real}] - \Pr[b' = 1|b = 0 \wedge \mathsf{real}]\ .$$

Putting it altogether, we get the conclusion.

5 Extensions in the Random Oracle Model

We insist on the fact that the previous construction is in the standard model, without any additional non-standard setup assumption. However, some improvements can be achieved in the random oracle model [2].

5.1 Robustness

First, note that in our security model, we defined the robustness, as a very interesting feature ($\mathsf{ShareVerify}$). Such a verification seems hard to do in the standard model, in our previous scheme. It was not available in [11] either. However, in the random oracle model, we can use proofs of equality of discrete logarithms for providing it, at almost no additional cost in our scheme above:

- when decrypting (C_1, C_2), using usk, one can generate $\mathsf{usk}' = \mathsf{usk}^\delta$, for a random δ, together with $\sigma = e(\mathsf{usk}, C_2)$.
- the validity can be checked by the existence of a common value δ such that

$$e\left(\mathsf{usk}', (h^{\alpha\gamma}) \times (h^\alpha)^{\mathsf{upk}}\right) = v^\delta \quad e\left(\mathsf{usk}', C_2\right) = \sigma^\delta.$$

  The latter can be a usual Schnorr-like proof π of equality of discrete logarithms [32] (the existence of a common exponent δ), and its non-interactive version using the Fiat-Shamir paradigm [17, 30].

The verification key is thus simply $\mathsf{uvk} = \mathsf{upk} = x$, the partial decryption consists of the triple $(\sigma, \mathsf{usk}', \pi)$, and the $\mathsf{ShareVerify}$ algorithm checks the validity of π.

5.2 Identity-Based

It is also simple to get an ID-based version in the random oracle model, as in [11] and [13], by simply taking $\mathsf{upk} = x = \mathcal{H}(\mathsf{ID})$ as in [4].

6 Intractability of (ℓ, m, t)-MSE-DDH

6.1 Notations

For the sake of simplicity, we focus to the symmetric case ($\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$). Let then $\mathcal{B} = (p, \mathbb{G}, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot))$ be a bilinear map group system. Let $g_0 \in \mathbb{G}$ be a generator of $\mathbb{G}$, and set $g = e(g_0, g_0) \in \mathbb{G}_T$. Let s, n be positive integers and $P, Q \in \mathbb{F}_p[X_1, \ldots, X_n]^s$ be two s-tuples of n-variate polynomials over $\mathbb{F}_p$. Thus, P and Q are just two lists containing s multivariate polynomials each: we write $P = (p_1, p_2, \ldots, p_s)$ and $Q = (q_1, q_2, \ldots, q_s)$ and impose that $p_1 = q_1 = 1$. For any function $h : \mathbb{F}_p \to \Omega$ and vector $(x_1, \ldots, x_n) \in \mathbb{F}_p^n$, the notation $h(P(x_1, \ldots, x_n))$ stands for

$$(h(p_1(x_1, \ldots, x_n)), \ldots, h(p_s(x_1, \ldots, x_n))) \in \Omega^s.$$

We use a similar notation for the s-tuple Q. Let $f \in \mathbb{F}_p[X_1, \ldots, X_n]$. It is said that f depends on (P, Q), which we denote by $f \in \langle P, Q \rangle$, when there exists a linear decomposition

$$f = \sum_{1 \le i,j \le s} a_{i,j} \cdot p_i \cdot p_j + \sum_{1 \le i \le s} b_i \cdot q_i \ , \qquad a_{i,j}, b_i \in \mathbb{Z}_p \ .$$

Let P, Q be as above and $f \in \mathbb{F}_p[X_1, \ldots, X_n]$. The (P, Q, f)-General Diffie-Hellman Exponent problems are defined as follows.

Definition 5 $((P,Q,f)$-$\mathsf{GDHE})$**.** *Given the tuple*

$$H(x_1,\ldots,x_n) = \left({g_0}^{P(x_1,\ldots,x_n)}, g^{Q(x_1,\ldots,x_n)}\right) \in \mathbb{G}^s \times \mathbb{G}_T^s \ ,$$

compute $g^{f(x_1,\ldots,x_n)}$*.*

Definition 6 $((P,Q,f)$-$\mathsf{GDDHE})$**.** *Given* $H(x_1,\ldots,x_n) \in \mathbb{G}^s \times \mathbb{G}_T^s$ *as above and* $T \in \mathbb{G}_T$*, decide whether* $T = g^{f(x_1,\ldots,x_n)}$*.*

We refer to [5] for a proof that (P,Q,f)-GDHE and (P,Q,f)-GDDHE have generic security when $f \notin \langle P,Q \rangle$. We will prove that our construction is secure by first exhibiting the polynomials P, Q and f involved in the security proofs, and then by showing that $f \notin \langle P,Q \rangle$.

6.2 (ℓ,m,t)-MSE–DDH

In this section, we prove the intractability of distinguishing the two distributions involved in the (ℓ,m,t)-$\mathsf{MSE\text{–}DDH}$ problem (cf. Corollary 3, section 4.3). We first review some results on the General Diffie-Hellman Exponent Problem, from [5]. In order to be the most general, we assume the easiest case for the adversary: when $\mathbb{G}_1 = \mathbb{G}_2$, or at least that an isomorphism that can be easily computed in either one or both ways is available.

Theorem 7 **([5]).** *Let* $P,Q \in \mathbb{F}_p[X_1,\ldots,X_n]$ *be two* s*-tuples of* n*-variate polynomials over* $\mathbb{F}_p$ *and let* $F \in \mathbb{F}_p[X_1,\ldots,X_n]$*. Let* d_P *(resp.* d_Q, d_F*) denote the maximal degree of elements of* P *(resp. of* Q,F*) and pose* $d = \max(2d_P, d_Q, d_F)$*. If* $F \notin \langle P,Q \rangle$ *then for any generic-model adversary* $\mathcal{A}$ *totalizing at most* q_G *queries to the oracles (group operations in* $\mathbb{G},\mathbb{G}_T$ *and evaluations of* e*) which is given* $H(x_1,\ldots,x_n)$ *as input and tries to distinguish* $g^{F(x_1,\ldots,x_n)}$ *from a random value in* $\mathbb{G}_T$*, one has*

$$\mathsf{Adv}(\mathcal{A}) \leq \frac{(q_G + 2s + 2)^2 \cdot d}{2p} \ .$$

Proof (of Corollary 3). In order to conclude with Corollary 3, we need to prove that our problem lies in the scope of Theorem 7. As already said, we consider the weakest case $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$ and thus pose $h_0 = {g_0}^{\beta}$. Our problem can be reformulated as (P,Q,F)-GDHE where

$$P = \begin{pmatrix} 1,\gamma,\gamma^2,\ldots,\gamma^{\ell+t-2}, & k\cdot\gamma\cdot f(\gamma) \\ \alpha,\alpha\cdot\gamma,\alpha\cdot\gamma^2,\ldots,\alpha\cdot\gamma^{\ell+t}, & \\ \beta,\beta\cdot\gamma,\beta\cdot\gamma^2,\ldots,\beta\cdot\gamma^{m-2} & \\ \alpha\cdot\beta,\alpha\cdot\beta\cdot\gamma,\alpha\cdot\beta\cdot\gamma^2,\ldots,\alpha\cdot\beta\cdot\gamma^{2m-1}, & k\cdot\alpha\cdot\beta\cdot g(\gamma)\cdot q(\gamma) \end{pmatrix}$$

$$Q = 1$$

$$F = k\cdot\beta\cdot f(\gamma),$$

and thus $n = 4$ and $s = 2(\ell+t) + 3m + 1$. We have to show that F is independent of (P,Q), i.e. that no coefficients $\{a_{i,j}\}_{i,j=1}^{s}$ and b_1 exist such that

$F = \sum_{i,j=1}^{s} a_{i,j} p_i p_j + b_1$ where the polynomials p_i are the one listed in P above. By making all possible products of two polynomials from P which are multiples of $k \cdot \beta$, we want to prove that no linear combination among the polynomials from the list R below leads to F:

$$R = \begin{pmatrix} k \cdot \beta \cdot g(\gamma),\ k \cdot \beta \cdot \gamma \cdot g(\gamma), \ldots,\ k \cdot \beta \cdot \gamma^{\ell+t-2} \cdot g(\gamma), \\ k \cdot \beta \cdot \gamma \cdot f(\gamma),\ k \cdot \beta \cdot \gamma^2 \cdot f(\gamma), \ldots,\ k \cdot \beta \cdot \gamma^{m-1} \cdot f(\gamma) \end{pmatrix}.$$

We simplify the task to refuting a linear combination of elements of the list R' below which leads to $f(\gamma)$:

$$R' = \begin{pmatrix} g(\gamma),\ \gamma \cdot g(\gamma), \ldots,\ \gamma^{\ell+t-2} \cdot g(\gamma), \\ \gamma \cdot f(\gamma),\ \gamma^2 \cdot f(\gamma), \ldots,\ \gamma^{m-1} \cdot f(\gamma) \end{pmatrix}.$$

Any such linear combination can be written as

$$\begin{aligned} & f(\gamma) = A(\gamma) \cdot f(\gamma) + B(\gamma) \cdot g(\gamma) \\ \Leftrightarrow\ & f(\gamma) \cdot (1 - A(\gamma)) = B(\gamma) \cdot g(\gamma) \end{aligned}$$

where A and B are polynomials such that $A(0) = 0$, $\deg A \leq m - 1$ and $\deg B \leq \ell + t - 2$. Since f and g are coprime by assumption, we must have $g \mid 1 - A$. Since $\deg g = m$ and $\deg A \leq m - 1$ this implies $A = 1$, which contradicts $A(0) = 0$. □

7 Conclusion

We presented a generalization of threshold public-key encryption to the dynamic setting. We first proposed a security model and then a new scheme, which is non-interactive, fully dynamic, and which is the first one to achieve constant-size ciphertexts. However, our scheme can be viewed as a first step toward the problem, since it still presents a few restrictions: our security proof relies on a new and non-standard assumption, and does not prevent adaptive adversaries, nor chosen-ciphertext attacks. However, it applies in the standard model.

Acknowledgements

We would like to thank the anonymous referees for their fruitful comments.

The second author was supported in part by the European Commission through the IST Program under Contract IST-2002-507932 ECRYPT, and by the CELAR.

References

1. Joonsang Baek and Yuliang Zheng. Identity-based threshold decryption. In *PKC 2004*, LNCS 2947, pages 262–276. Springer, 2004.

2. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS '93*, pages 62–73. ACM Press, 1993.
3. John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334. IEEE Computer Society, 2007.
4. Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity based encryption without random oracles. In *EUROCRYPT 2004*, LNCS 3027, pages 223–238. Springer, 2004.
5. Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In *EUROCRYPT 2005*, LNCS 3494, pages 440–456. Springer, 2005.
6. Dan Boneh, Xavier Boyen, and Shai Halevi. Chosen ciphertext secure public key threshold encryption without random oracles. In *CT-RSA 2006*, LNCS 3860, pages 226–243. Springer, 2006.
7. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In *CRYPTO 2001*, LNCS 2139, pages 213–229. Springer, 2001.
8. Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *CRYPTO 2005*, LNCS 3621, pages 258–275. Springer, 2005.
9. Ran Canetti and Shafi Goldwasser. An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In *EUROCRYPT '99*, LNCS 1592, pages 90–106. Springer, 1999.
10. Zhenchuan Chai, Zhenfu Cao, , and Yuan Zhou. Efficient id-based broadcast threshold decryption in ad hoc network. In *IMSCCS (2)*, pages 148–154. IEEE Computer Society, 2006.
11. Vanesa Daza, Javier Herranz, Paz Morillo, and Carla Ràfols. CCA2-secure threshold broadcast encryption with shorter ciphertexts. In *ProvSec 2007*, LNCS 4784, pages 35–50. Springer, 2007.
12. Alfredo De Santis, Yvo Desmedt, Yair Frankel, and Moti Yung. How to share a function securely. In *26th ACM STOC*, pages 522–533. ACM Press, 1994.
13. Cécile Delerablée. Identity-based broadcast encryption with constant size ciphertexts and private keys. In *ASIACRYPT 2007*, LNCS 4833, pages 200–215. Springer, 2007.
14. Cécile Delerablée, Pascal Paillier, and David Pointcheval. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In *Pairing 2007*, LNCS 4575, pages 39–59. Springer, 2007.
15. Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In *CRYPTO '89*, LNCS 435, pages 307–315. Springer, 1990.
16. Amos Fiat and Moni Naor. Broadcast encryption. In *CRYPTO '93*, LNCS 773, pages 480–491. Springer, 1994.
17. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO '86*, LNCS 263, pages 186–194. Springer, 1987.
18. Yair Frankel. A practical protocol for large group oriented networks. In *EUROCRYPT '89*, LNCS 434, pages 56–61. Springer, 1990.
19. Rosario Gennaro, Shai Halevi, Hugo Krawczyk, and Tal Rabin. Threshold RSA for dynamic and ad-hoc groups. In *EUROCRYPT 2008*, LNCS 4965, pages 88–107. Springer, 2008.
20. Hossein Ghodosi, Josef Pieprzyk, and Rei Safavi-Naini. Dynamic threshold cryptosystems: A new scheme in group oriented cryptography. In *PRAGOCRYPT '96*, pages 370–379, 1996.

21. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
22. Michael T. Goodrich, Jonathan Z. Sun, and Roberto Tamassia. Efficient tree-based revocation in groups of low-state devices. In *CRYPTO 2004*, LNCS 3152, pages 511–527. Springer, 2004.
23. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM CCS 2006*, pages 89–98. ACM Press, 2006. Available as Cryptology ePrint Archive Report 2006/309.
24. Dani Halevy and Adi Shamir. The LSD broadcast encryption scheme. In *CRYPTO 2002*, LNCS 2442, pages 47–60. Springer, 2002.
25. Antoine Joux. A one-round protocol for tripartite diffie-hellman. In *ANTS*, pages 385–394, 2000.
26. Benoît Libert and Jean-Jacques Quisquater. Efficient revocation and threshold pairing based cryptosystems. In *22nd ACM PODC*, pages 163–171. ACM Press, 2003.
27. Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In *CRYPTO 2001*, LNCS 2139, pages 41–62. Springer, 2001.
28. Moni Naor. On cryptographic assumptions and challenges (invited talk). In *CRYPTO 2003*, LNCS 2729, pages 96–109. Springer, 2003.
29. Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*. ACM Press, 1990.
30. David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000.
31. Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *CRYPTO '91*, LNCS 576, pages 433–444. Springer, 1992.
32. Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In *CRYPTO '89*, LNCS 435, pages 239–252. Springer, 1990.
33. Victor Shoup. Lower bounds for discrete logarithms and related problems. In *EUROCRYPT '97*, LNCS 1233, pages 256–266. Springer, 1997.
34. Victor Shoup. ISO 18033-2: An emerging standard for public-key encryption. December 2004. Final Committee Draft.
35. Victor Shoup and Rosario Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. In *EUROCRYPT '98*, LNCS 1403, pages 1–16. Springer, 1998.