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 Aggregate algorithm (over $\mathbb{G}_T$) described in [14] to combine the decryption shares. The 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.

Setup(λ). 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 $\mathsf{EK} = \left(m, u, v, h^\alpha, \{h^{\alpha.\gamma^i}\}_{i=1}^{2m-1}, \mathcal{D}\right)$, and the combining key is $\mathsf{CK} = \left(m, h, \{h^{\gamma^i}\}_{i=1}^{m-2}, \mathcal{D}\right)$, where $u = g^{\alpha.\gamma}$, 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.

Join(MK, 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 $e\left(C_1, C_2'\right) = e\left(C_1', C_2\right)$ 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:

$$p_{(T,\mathcal{S})}(\gamma) = \frac{1}{\gamma} \cdot \left(\prod_{x \in \mathcal{S} \cup \mathcal{D}_{m+t-s-1} - T} (\gamma + x) - c_{(T,\mathcal{S})} \right) ,$$

$$c_{(T,\mathcal{S})} = \prod_{x \in \mathcal{S} \cup \mathcal{D}_{m+t-s-1} - T} x ,$$

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

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

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

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 $\mathsf{Combine}$ algorithm requires the computation of

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

given $\Sigma = \{\sigma_j = e\left(g, C_2\right)^{\frac{1}{\gamma+x_j}}\}_{j=1}^{t}$ 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 $1 \leq j < \ell \leq r$,

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

The $\mathsf{Aggregate}$ 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 $L_t = L_{t-1,t}$.