Dynamic Threshold Public-Key Encryption

Cécile Delerablée — Orange Labs
David Pointcheval — Ecole normale supérieure

CRYPTO 2008
August 20th, 2008

Threshold Cryptography

When one cannot fully trust a unique person, but possibly a pool of individuals, the secret operation is distributed, so that authorized subsets only can perform it

- signature
- decryption

Threshold Cryptography

The access structure (authorized subsets) is defined by a threshold:

- any group of t players can perform the secret operation
- below this threshold, no power is provided to them

Threshold Public-Key Encryption

A ciphertext can be decrypted **only if at least *t* users** cooperate. Below this threshold, no additional information about the plaintext is leaked.

Many applications:

- electronic voting (decryption of the final result only)
- key-escrow
- identity-based cryptography (secret key extraction)
- etc

Classical Technique: ElGamal

$\mathbb{G} = \langle g \rangle$ is a group of prime order p

Lagrange Interpolation (Shamir's Secret Sharing)

- $\mathcal{GM}$ generates a polynomial P of degree $t-1$ over $\mathbb{Z}_p$
- each group member $i \in \{1, \ldots n\}$ receives $\mathsf{sk}_i = P(i)$
- the group public key is $\mathsf{PK} = g^{\mathsf{sk}}$, where $\mathsf{sk} = P(0)$

t users can recover sk, less than t users have no information.

Threshold ElGamal Encryption

- one can encrypt a message $m \in \mathbb{G}$: $c_1 = g^r, c_2 = \mathsf{PK}^r \times m$
- in order to decrypt, one has to compute $a = \mathsf{PK}^r = c_1^{\mathsf{sk}}$:
  - each user i computes $a_i = c_1^{\mathsf{sk}_i}$
  - with t values, a can be "interpolated".

Limitations

At the key generation phase:

- the target group (or set) is fixed (the public key)
- the threshold t, to define the authorized subsets, is fixed

Dynamic Threshold Encryption

- any user can *dynamically* join the system as a future receiver
- the sender can *dynamically* choose the target set $\mathcal{S}$
- the sender can *dynamically* set the threshold t

Related to

- Threshold broadcast encryption

[Daza, Herranz, Morillo, Ràfols – ProvSec '07]

Ciphertext linear in $O(\mathcal{S})$

Outline

A Dynamic TPKE Scheme: Encryption/Decryption

Setup(λ). It outputs a set of parameters
PARAM $=$ (MK, EK, DK, VK, CK)
MK is the master secret key: for adding new users

Join(MK, ID). With MK and the identity ID of a new user, it outputs the user's keys (usk, upk, uvk)

Encrypt(EK, $\mathcal{S}$, t, M). With the target set $\mathcal{S}$ (the public keys upk), and the threshold t, it outputs an encryption of the message M

ShareDecrypt(DK, ID, usk, C). With his private key usk, user ID gets his decryption share σ, or $\perp$

Combine(CK, $\mathcal{S}$, t, C, T, Σ). With an authorized subset T (subset of t targeted users), and $\Sigma = (\sigma_1, \ldots, \sigma_t)$ a list of t decryption shares, it outputs a cleartext M, or $\perp$

A Dynamic TPKE Scheme (Cont'd)

Robustness is achieved by **public** verification tools:

ValidateCT(EK, $\mathcal{S}$, t, C). It checks whether C is a valid ciphertext with respect to EK, $\mathcal{S}$ and t

ShareVerify(VK, ID, uvk, C, σ). It checks whether σ is a valid decryption share with respect to uvk

KEM-DEM methodology:

- an ephemeral secret key K is first generated (KEM)
- a symmetric mechanism is used to encrypt the data (DEM)

Encrypt(EK, $\mathcal{S}$, t). With the target set $\mathcal{S}$ (the public keys upk), and a threshold t, it outputs an ephemeral key K, and the key encapsulation material **HDR**

Security Model

Correctness. Valid encryptions should be correctly checked and decrypted, legitimate decryptions should be correctly verified, and should lead to the plaintext/ephemeral key

Robustness. It t shares are correctly checked with **ShareVerify**, then the **Combine** algorithm outputs the correct key K

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

Security Model: Privacy

Setup: The challenger runs **Setup**(λ) and the public parameters (EK, DK, VK, CK) are given to the adversary.

Query phase 1: The adversary $\mathcal{A}$ adaptively issues queries:

- **Join** queries (on a new user ID)
- **Corrupt** queries (on an existing user ID) to learn private keys
- **ShareDecrypt** queries (on an ID and a header **HDR**) to learn the partial decryption

Challenge: $\mathcal{A}$ outputs a set of users $\mathcal{S}^\star$ and a threshold $t^\star$. The challenger randomly selects $b \leftarrow \{0,1\}$, and gets $(K_0, \mathbf{HDR}^\star) = \mathbf{Encrypt}(\mathrm{EK}, \mathcal{S}^\star, t^\star)$, and randomly chooses an ephemeral key K_1: it returns $(K_b, \mathbf{HDR}^\star)$ to $\mathcal{A}$.

Query phase 2: as **Query phase 1**

Guess: The adversary $\mathcal{A}$ outputs its guess b' for b

Security Levels

With the natural restrictions on the oracle queries wrt. the target set and the threshold, the advantage of $\mathcal{A}$ is defined as

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

As usual, $\mathsf{Adv}(T, n, m, t, q_C, q_D)$ denotes the maximal value over the adversaries $\mathcal{A}$ such that

- it runs within time T
- it makes at most
  - n **Join**-queries
  - q_C **Corrupt**-queries
  - q_D **ShareDecrypt**-queries
- the size of $\mathcal{S}^\star$ is upper-bounded by m
- the value of $t^\star$ is upper-bounded by t.

Security Level: the Basic one

Non-Adaptive Adversary (NAA)

We restrict the adversary to decide before the setup the set $\mathcal{S}^\star$ and the threshold $t^\star$ to be sent to the challenger

Non-Adaptive Corruption (NAC)

We restrict the adversary to decide before the setup the identities that will be corrupted

Chosen-Plaintext Adversary (CPA)

We prevent the adversary from issuing **ShareDecrypt**-queries

(n, m, t, q_C)-IND-NAA-NAC-CPA security

Non-adaptive adversary, non-adaptive corruption, and CPA

Aggregate Tool

Our **Combine** algorithm makes use of the **Aggregate** tool

[Delerablée, Paillier, and Pointcheval – Pairing '07]

It allows to compute

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

given A and $\Sigma = \{(x_j, a_j = A^{\frac{1}{\gamma+x_j}})\}_{j=1}^{t}$, but γ private, where the x_j's are pairwise distinct.

Our Construction: Setup

Setup(λ). Given a bilinear setting, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, with

- generators $g \in \mathbb{G}_1$ and $h \in \mathbb{G}_2$
- $\gamma, \alpha \xleftarrow{R} \mathbb{Z}_p^*$
- $\mathcal{D} = \{d_i\}_{i=1}^{m-1}$ of random values in $\mathbb{Z}_p$, where m is the maximal size of a target set ($\mathcal{D}$ corresponds to a set of public dummy users)
- $u = g^{\alpha\cdot\gamma}$
- $v = e(g, h)^{\alpha}$

- The master secret key: $\mathsf{MK} = (g, \gamma, \alpha)$
- The encryption key: $\mathsf{EK} = \left(m, u, v, h^{\alpha}, \{h^{\alpha\cdot\gamma^i}\}_{i=1}^{2m-1}, \mathcal{D}\right)$
- The decryption key: $\mathsf{DK} = \emptyset$
- The combining key: $\mathsf{CK} = \left(m, h, \{h^{\gamma^i}\}_{i=1}^{m-2}, \mathcal{D}\right)$

Our Construction: Join/Encrypt

Join(MK, ID). Given $\mathsf{MK} = (g, \gamma, \alpha)$, and an identity ID, it randomly chooses a **new** $x \in \mathbb{Z}_p$:

$$\mathsf{upk} = x \qquad \mathsf{usk} = g^{\frac{1}{\gamma+x}}$$

Encrypt$(\mathsf{EK}, \mathcal{S}, t)$. Given a set $\mathcal{S} = \{\mathsf{upk}_1 = x_1, \ldots, \mathsf{upk}_s = x_s\}$ and a threshold t (with $t \leq s \leq m$), **Encrypt** picks $k \xleftarrow{R} \mathbb{Z}_p^*$, and sets $\mathbf{HDR} = (C_1, C_2)$ and $K = v^k$:
$C_1 = u^{-k} \qquad C_2 = h^{k \cdot \alpha \cdot \prod_{x_i \in \mathcal{S}} (\gamma + x_i) \cdot \prod_{x \in \mathcal{D}_{m+t-s-1}} (\gamma + x)}$

- a set of $m + t - s - 1$ dummy users + a set of s authorized users $\Rightarrow$ a polynomial of degree $m + t - 1$ in the exponent of h:
- $m + t - 1 \leq 2m - 1$: can be computed from EK
- the cooperation of t authorized users will decrease the degree of the polynomial in v to degree $m - 1$: **too high degree for CK!**

Our Construction: Decryption

ShareDecrypt(ID, usk, **HDR**). Given $\mathbf{HDR} = (C_1, C_2)$ and $\mathsf{usk} = g^{\frac{1}{\gamma+x}}$

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

Combine(CK, **HDR**, T, Σ). Given a set Σ of t decryption shares:

$$K = \left(e\left(C_1, h^{p(\gamma)}\right) \cdot \mathbf{Aggregate}(v, \Sigma) \right)^{\frac{1}{c}}$$

- $c = \prod_{x \in \mathcal{S} \cup \mathcal{D}_{m+t-s-1} \setminus T} x \in \mathbb{Z}_p$
- $p(\gamma) = \frac{1}{\gamma} \cdot \left(\prod_{x \in \mathcal{S} \cup \mathcal{D}_{m+t-s-1} \setminus T} (\gamma + x) - c \right)$, a polynomial of degree $m - 2$, computable from CK

Our Construction: Decryption (Cont'd)

$$\begin{aligned} K' &= e\left(C_1, h^{p(\gamma)}\right) \cdot \textbf{Aggregate}(v, \Sigma) \\ &= e\left(g^{-k\cdot\gamma}, h^{p(\gamma)}\right) \cdot v^{k \cdot \prod_{x \in \mathcal{S} \cup \mathcal{D}_{m+t-s-1} \setminus T} (\gamma + x)} \\ &= v^{-k\cdot\gamma\cdot p(\gamma)} \cdot v^{k\cdot(\gamma\cdot p(\gamma)+c)} \\ &= v^{k\cdot c} = K^c. \end{aligned}$$

ValidateCT(EK, $\mathcal{S}$, t, **HDR**). Given **HDR** $= (C_1, C_2)$

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

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 $C_2 = {C_2'}^k$:

$$e\left(C_1, C_2'\right) \stackrel{?}{=} e\left(C_1', C_2\right)$$

Our Construction: Security Result

Theorem

$Adv(T, n, m, t, \ell, 0) \leq 2 \cdot Adv^{\mathrm{mse-ddh}}(T', \ell, m, t)$.

(ℓ, m, t)-Multi-Sequence of Exponents DDH

Let f and g be two random coprime polynomials, 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

$$\begin{array}{ll} x_1, \ldots, x_\ell, & y_1, \ldots, y_m \\ g, g^\gamma, \ldots, g^{\gamma^{\ell+t-2}}, & g^{k\cdot\gamma\cdot f(\gamma)}, \\ g^\alpha, g^{\alpha\cdot\gamma}, \ldots, g^{\alpha\cdot\gamma^{\ell+t}}, & \\ h, h^\gamma, \ldots, h^{\gamma^{m-2}}, & \\ h^\alpha, h^{\alpha\cdot\gamma}, \ldots, h^{\alpha\cdot\gamma^{2m-1}}, & h^{k\cdot g(\gamma)}, \text{ and } T \in \mathbb{G}_T, \end{array}$$

decide whether T is equal to $e(g,h)^{k\cdot f(\gamma)}$ or not

Our Construction: Security Result

Lemma (Generic Security [Boneh, Boyen, Goh – Eurocrypt '05]**)**

For any probabilistic algorithm $\mathcal{A}$ that makes at most q queries to the group oracles, with $d = 4(\ell + t) + 6m + 2$

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

Theorem (Generic Security)

Our construction is secure

- *against non-adaptive and generic adversaries*
- *under non-adaptive corruption and chosen-plaintext attacks*

Our Construction: Efficiency

Ciphertext Size

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

The header has a constant size: two group elements

Decryption

Given **HDR** $= (C_1, C_2)$ and usk $= g^{\frac{1}{\gamma + x}}$, $\sigma = e(\mathrm{usk}, C_2)$.

The user decryption is quite efficient: one pairing

Non-Interactive Combination

$$K = \left(e\left(C_1, h^{p(\gamma)}\right) \cdot \mathbf{Aggregate}(v, \Sigma) \right)^{\frac{1}{c}}$$

The combination step does not need any interaction

Extensions: Random Oracle Model

All the previous properties are achieved in the standard model (under the MSE$-$DDH assumption)

Robustness

Easily achieved in the random oracle model, using Schnorr-like proof of equality of discrete logarithms

Identity-Based

It is simple to get an ID-based version in the random oracle model, by simply taking upk $= x = \mathcal{H}(\mathrm{ID})$

Conclusion

- Security model for (dynamic) threshold public-key encryption (a.k.a. threshold broadcast encryption)
- Efficient and provably secure candidate
  the first with constant-size header

But still a lot of work on this topic:

- Use of a new non-standard assumption
- Secure against restricted adversaries only:
  - Chosen-plaintext attacks
  - Non-adaptive adversaries