



CoverCrypt

Full Documentation

Version 1.0

Date: 15/07/2022



Table Of Contents

1	Introduction	2
2	Example - hierarchical axes	3
3	CoverCrypt (new version)	4
4	Security Properties	6
5	Updates	8
5.1	Adding Users and Rights	8
5.2	Revocation	8
6	Comparision with ABE	9
6.1	Number of attributes	9
6.2	Size of ciphertexts	9
7	Parameters	10



1 Introduction

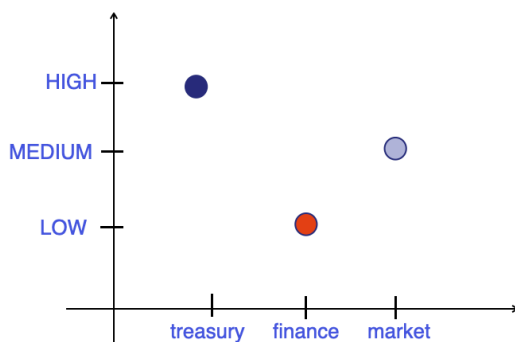
The scheme presented in this document is the new and current version of CoverCrypt whose implementation has not been yet delivered. The solution has been designed with the help of David Pointcheval who is an advisor of Cosmian. We will soon provide a RUST implementation of the scheme with benchmarks.

CoverCrypt is a multi-user encryption solution which provides access rights to users with respect to an access policy. It has been proposed as a more efficient alternative to Key-Policy Attribute encryption scheme where the policy over attributes can be expressed as a union of users' rights. In this document and for CoverCrypt, we will use sets in place of attributes. We will explain the correspondance in the first section.

This documentation specifies the procedure for: - managing users keys and access rights, - encrypting messages with respect to target rights, - decrypting the message when users' are granted rights for decryption.

2 Example - hierarchical axes

The figure below illustrates a hierarchical policy where domains: treasury, finance, market; and security level in increasing order: HIGH, MEDIUM and LOW are on x- and y- axes respectively.



- S_1, S_2, S_3 with $S_1 \subset S_2 \subset S_3$ are associated to finance with levels LOW, MEDIUM and HIGH respectively;
- S_4, S_5, S_6 with $S_4 \subset S_5 \subset S_6$ are associated to treasury with levels LOW, MEDIUM and HIGH respectively;
- S_7, S_8, S_9 with $S_7 \subset S_8 \subset S_9$ are associated to market with level LOW, MEDIUM and HIGH respectively;

Then, during the Join procedure, users get keys according to their rights:

- A right for finance with security level LOW is associated with set S_1 . A user U joining the system with this right receives sk_1 ;
- A right for market with security level MEDIUM is associated with sets $S_7 \subset S_8$. A user U joining the system with this right receives both sk_7 and sk_8 .
- A right with treasury with security level HIGH is associated with sets $S_4 \subset S_5 \subset S_6$. A user U joining the system with this right obtains sk_4 , sk_5 , and sk_6 .



3 CoverCrypt (new version)

Let \mathbb{G} be a group of prime order q , with a generator g , in which the Computational Diffie-Hellman problem is hard. We describe below the encryption mechanism that can target specific rights for decryption, and traceability of dishonest users:

- **Setup** $((S_i)_i)$: it generates the master public key mpk and the master secret key msk as follows:

- it samples random $u, v, s \xleftarrow{\$} \mathbb{Z}_q$ and sets

$$U \leftarrow g^u \quad V \leftarrow g^v \quad H \leftarrow g^s$$

- For each set $S_i \in \mathcal{S}$, where $\mathcal{S} = (S_i)_i$, it chooses a random $x_i \xleftarrow{\$} \mathbb{Z}_q$ and sets $H_i \leftarrow H^{x_i}$.

Let $\text{msk} \leftarrow (u, v, s, (x_i)_i)$ and $\text{mpk} \leftarrow (\mathbb{G}, g, U, V, H, (H_i)_i)$. And we assume that mpk is known to everybody.

- **Join** (msk, j, A_j) : it takes as input the master secret key msk , a user identifier j , and the set A_j of indices i so that user j belongs to S_i , and provides its secret key SK_j .

For the tracing, one first chooses random scalars (a_j, b_j) such that

$$a_j \cdot u + b_j \cdot v = s$$

Then $\text{SK}_j \leftarrow (a_j, b_j, (x_i)_{i \in A_j})$ is provided to user j .

- **Enc** (K, B) : it takes as input a bitstring $K \in \{0, 1\}^n$ to encrypt to all the users belonging to S_i , for $i \in B$, and outputs the encryption of K .
 - it samples a random $r \xleftarrow{\$} \mathbb{Z}_q$;
 - it sets $C \leftarrow U^r$ and $D \leftarrow V^r$;
 - for every $i \in B$, it generates $K_i \leftarrow H_i^r$.

The ciphertext thus consists of $(C, D, (E_i = \mathcal{H}(K_i) \oplus K)_{i \in B})$, where \mathcal{H} is a hash function onto $\{0, 1\}^n$.

- **Dec** $(\text{SK}_j, (C, D, (E_i = K_i \oplus K)_{i \in B}))$: it takes as input a user's secret key and a ciphertext, it outputs the encrypted key K .
 - the user first chooses an index $i \in B \cap A_j$, in both its set of rights A_j and the rights B of the ciphertext, and then uses $x_i = \text{sk}_i \in \text{SK}_j$;
 - it can compute $K_i = (C^{a_j} D^{b_j})^{x_i}$, and extract $K = E_i \oplus \mathcal{H}(K_i)$.



One can note that

$$K_i = (C^{a_j} D^{b_j})^{x_i} = (g^{ura_j + vrb_j})^{x_i} = (g^{(ua_j + vb_j)r})^{x_i} = g^{sr x_i}$$



4 Security Properties

The previous construction provides two security properties: the privacy of the encrypted key K and the traceability of users. More precisely:

Privacy: any collusion of users does not learn more than what the users could get individually and then put in common.

The security game is the following one: an adversary \mathcal{A} corrupts users and learns their secret keys SK_j for $j \in \mathcal{C}$ (the set of corrupted users), and asks for the encryption $K^{(b)}$ among $\{K^{(0)}, K^{(1)}\}$ for the set B . We say that the adversary wins the security game if it can guess b , while $B \cap (\cup_{j \in \mathcal{C}} A_j) = \emptyset$.

In the static corruption setting, we know the set \mathcal{C} before the Setup, for which we are given a Diffie-Hellman instance $(g, X = g^x, Y = g^y)$: - $U \leftarrow g^u, V \leftarrow g^v$, and $H \leftarrow g^s$, for random u, v, s ; - For $i \in \cup_{j \in \mathcal{C}} A_j$, choose a random x_i and set $H_i \leftarrow H^{x_i} = g^{sx_i}$; - For $i \notin \cup_{j \in \mathcal{C}} A_j$, choose a random y_i and set $H_i \leftarrow X^{sy_i} = g^{sxy_i}$, which virtually sets $x_i \leftarrow xy_i$;

Join-queries can be generated as in the official procedure, as for corrupted users, x_i is known. However, when encrypting $K^{(b)}$ for the set B : - $C \leftarrow Y^u = g^{yu}, D \leftarrow Y^v = g^{yv}$, which virtually sets $r \leftarrow y$; - For $i \in B$, it generates a random $E_i \xleftarrow{\$} \{0, 1\}^n$.

The adversary can only see the difference of the random E_i if it asks for $\mathcal{H}(K_i)$, for some $i \in B$, while $B \cap (\cup_{j \in \mathcal{C}} A_j) = \emptyset$: $K_i = H_i^r = g^{sxy_iy} = (g^{xy})^{sy_i}$. From all the queries to \mathcal{H} , in the random oracle model, and s and $(y_i)_i$, one can extract g^{xy} , which is the Diffie-Hellman value of (X, Y) in basis g .

Traceability: without any collusion, a user alone can be traced with a black-box tracing procedure.

In order to trace a pirate decoder that contains a key (a, b) : one generates a fake ciphertext with $C \leftarrow U^r \cdot g^{bs}$ and $D \leftarrow V^r \cdot g^{-as}$ for random $r, s \xleftarrow{\$} \mathbb{Z}_q$, and then $K_i \leftarrow H_i^r$.

User with key (a, b) will compute $C^a D^b = U^{ar} \cdot g^{abs} \cdot V^{br} \cdot g^{-abs} = (U^a \cdot V^b)^r = H^r$, and will eventually get the correct key K if it owns x_i .

Other users will compute $C^{a'} D^{b'} = U^{a'r} \cdot g^{a'bs} \cdot V^{b'r} \cdot g^{-ab's} = (U^{a'} \cdot V^{b'})^r \cdot g^{(a'b - ab')s} = H^r \cdot g^{(a'b - ab')s}$, which will unlikely be correct as $a'b - ab' \neq 0 \pmod q$.

1-Collusion Resistance: any collusion of 2 or more users can build an anonymous key.

As we have:

$$a_1 u + b_1 v = s \quad a_2 u + b_2 v = s$$

any convex combination, with $\alpha + \beta = 1$ leads to an anonymous key:

$$(\alpha a_1 + \beta a_2) u + (\alpha b_1 + \beta b_2) v = s$$



Indeed, we have

$$\alpha a_1 u + \alpha b_1 v + \beta a_2 u + \beta b_2 v = (\alpha + \beta)s$$

Hence, $(a = \alpha a_1 + \beta a_2, b = \alpha b_1 + \beta b_2)$ is an anonymous key that cannot be traced.



5 Updates

5.1 Adding Users and Rights

A new user joining the system will receive secret keys associated to its rights. These rights may evolve and the policy can be enriched over the time with more subsets S_i , and thus more keys sk_i .

5.2 Revocation

For revocation, one can use time-periods, and then a three-dimensional space, with $sk_{t,i}$. When the time-period changes, users receives the keys $sk_{t,i}$ associated to their right. The tracing part (a_j, b_j) does not need to be updated.

On the other hand, stored data does not need to be re-encrypted, but the key K must be re-encrypted under the new $sk_{t,i}$. It can be done without any private information: but we need a slight modification with $\text{Enc}(K, B)$ that now takes as input a key $K \in \mathbb{G}$, while $H_{t,i} = H^{x_{t,i}}$ depends on the time-period t - it samples a random $r \xleftarrow{\$} \mathbb{Z}_q$; - it sets $C \leftarrow U^r$ and $D \leftarrow V^r$; - for every $i \in B$, it generates $K_{t,i} \leftarrow H_{t,i}^r$ and $E_{t,i} = K \times K_{t,i}$

The ciphertext of K is thus $(C, D, (E_{t,i})_i)$, and data is encrypted with the session key $\mathcal{H}(K) \in \{0, 1\}^n$.

When updating the keys: $H_{t+1,i} \leftarrow H_{t,i} \cdot U^{\Delta_{t,i}} = H^{x_{t,i} + \Delta_{t,i} \cdot u/s}$. The ciphertext should be updated so that

$$E_{t+1,i} = K \times H_{t+1,i}^r = K \times H_{t,i}^r \cdot U^{r \cdot \Delta_{t,i}} = E_{t,i} \cdot C^{\Delta_{t,i}}$$



6 Comparision with ABE

In CoverCrypt, attributes are replaced by a set of unitary rights. For example, two attributes for finance and LOW will correspond to one set.

Compared to the ABE solution, the representation of attributes and ciphertext is more compact in CoverCrypt:

6.1 Number of attributes

In CoverCrypt, attributes are encoded as sets and a unitary set corresponds to the AND of two attributes. So, while the ciphertext is linear in the number of attributes in both cases (ABE solution and CoverCrypt), the representation of users' rights is more compact in CoverCrypt.

6.2 Size of ciphertexts

In CoverCrypt, we do not make use of pairing operations.

Let \mathbb{G} be a bilinear group of prime order and let g be a generator of \mathbb{G} . Denoting as $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ the bilinear map and denoting m and m_T the size of an element in \mathbb{G} and \mathbb{G}_T respectively: - in the ABE solution, a ciphertext is of size $\gamma \cdot m + m_T$ - in CoverCrypt, a ciphertext is of size $(\gamma + 2) \cdot m$.



7 Parameters