An Efficient Ciphertext-Policy Attribute-Based Encryption Scheme with Policy Update

Changji Wang[1,*] and Yuan Yuan[2]

Abstract: Ciphertext-policy attribute-based encryption (CP-ABE) is a promising cryptographic solution to the problem for enforcing fine-grained access control over encrypted data in the cloud. However, when applying CP-ABE to data outsourcing scenarios, we have to address the challenging issue of policy updates because access control elements, such as users, attributes, and access rules may change frequently. In this paper, we propose a notion of access policy updatable ciphertext-policy attribute-based encryption (APU-CP-ABE) by combining the idea of ciphertext-policy attribute-based key encapsulation and symmetric proxy re-encryption. When an access policy update occurs, data owner is no longer required to download any data for re-encryption from the cloud, all he needs to do is generate a re-encryption key and produce a new encapsulated symmetric key, and then upload them to the cloud. The cloud server executes re-encryption without decryption. Because the re-encrypted ciphertext is encrypted under a completely new key, users cannot decrypt data even if they keep the old symmetric keys or parts of the previous ciphertext. We present an APU-CP-ABE construction based on Syalim et al.'s [Syalim, Nishide and Sakurai (2017)] improved symmetric proxy re-encryption scheme and Agrawal et al.'s [Agrawal and Chase (2017)] attribute-based message encryption scheme. It requires only 6 bilinear pairing operations for decryption, regardless of the number of attributes involved. This makes our construction particularly attractive when decryption is time-critical.

Keywords: Ciphertext-policy attribute-based encryption, key encapsulation mechanism, access structure, all-or-nothing transform, cloud computing.

1 Introduction

Cloud computing is gaining popularity as more companies decide to deploy their services and applications to the cloud. At the same time, security and privacy concerns for the data in the cloud are growing. Once users outsource their private data to the cloud, they lose the direct control of their data and have to trust the cloud service provider (CSP) reluctantly.

[1] School of Information Science and Technology, Guangdong University of Foreign Studies, Guangzhou, 510006, China.

[2] School of Mathematics and Statistics, Guangdong University of Foreign Studies, Guangzhou, 510006, China.

[*] Corresponding Author: Changji Wang. Email: wchangji@gmail.com.

Received: 30 January 2019; Accepted: 22 January 2020.

Unfortunately, CSPs are generally considered as honest-but-curious, which means the cloud will carry out its promised operations honestly, but might pry into the sensitive data led by business interest or curiosity [Sadiku, Musa and Momoh (2014)].

To protect sensitive data and prevent unauthorized access by illegal visitors, including CSPs, a straightforward solution for data owners is to encrypt data before outsourcing them to the cloud [Liu, Peng and Wang (2018)]. Traditional public key encryption or identity-based encryption are one-to-one solutions, they cannot efficiently provide data owners to selectively share their encrypted data at a fine-grained level. To address above requirement for enforcing fine-grained access control over encrypted data, Sahai et al. [Sahai and Waters (2005)] introduced the concept of attribute-based encryption (ABE), which has been identified as a potentially useful basis for fine-grained sharing of encrypted data in the cloud. There are two types of ABE schemes depending on the form of ciphertext and key: key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE). In a KP-ABE scheme [Goyal, Pandey, Sahai et al. (2006)], ciphertext is linked to a set of attributes and user's private key is tied to an access structure. In a CP-ABE scheme [Waters (2011)], ciphertext is linked to an access structure and user' private key is tied to a set of attributes. Since access policy is determined by the data owner in CP-ABE, which is conceptually closer to traditional access control methods such as role-based access control that provides fine-grained access control over encrypted data, it is more suitable for access control applications in cloud environment.

In a CP-ABE based access control system, data confidentiality protection and access control enforcement are achieved through the following hybrid encryption ways:

- Data owner encrypts the original data to obtain ciphertext c_1 by using a secure symmetric cipher (e.g., AES) with a randomly chosen symmetric key K. Then, data owner encrypts the symmetric key by using a secure CP-ABE scheme under a certain access policy to obtain ciphertext c_2.
- Data owner uploads the ciphertexts c_1 and c_2 along with the access policy to the cloud.
- A data user will be able to obtain the symmetric key K by decrypting c_2 if and only if his attributes set satisfies the access policy. Finally, the data user can obtain the original data by decrypting c_1 with the symmetric key K.

For real-world applications of CP-ABE, the issue of access policy update must be considered. In some scenarios, data owner may want to change the access policy embedded in the ciphertext because of some changes in the intended recipients of the data. In this case, some users who satisfied the old access policy in the ciphertext may no longer satisfy the new access policy, and those users should be removed from the legitimate recipients such that they can no longer decrypt the ciphertext. Suppose that data owner revokes the access permission of user Alice with an attribute set to the file *F*. On the one hand, data owner needs to update the access policy such that Alice's attribute set does not meet the new access policy. On the other hand, as access policy is related only to CP-ABE ciphertext, re-encrypting file F is necessary to ensure that Alice can no longer decrypt the data. An intuitive policy update process is described as follows.

- Data owner retrieves the ciphertext (c_1, c_2) from the cloud and decrypts them to obtain the original file.

- Data owner chooses a new symmetric key K' and produces a new ciphertext c_1' by encrypting the original file with the new symmetric key.
- Data owner generates a new access policy that the data user's attributes set does not satisfy the new access policy, and produces the ciphertext c_2' by encrypting the new symmetric key K' under the new access policy.
- Finally, data owner upload c_1' and c_2' to the cloud and delete c_1 and c_2.

Obviously, the above policy update process will incur both huge communication costs and computational burden on the data owner, especially when big data is stored in the cloud.

To improve the efficiency of access policy update, Cheng et al. [Cheng, Wang, Ma et al. (2013)] proposed a policy update scheme for CP-ABE by applying all-or-nothing transform (AONT). In their scheme, data is first split into slices such that all slices are needed to recover the original data. Then, a random slice of data is encrypted with hybrid encryption of CP-ABE and symmetric encryption so that the original data cannot be recovered unless that slice is decrypted. When an access policy update occurs, data owner decrypts the encrypted slice, and then a different slice is randomly chosen to be newly encrypted using hybrid encryption. As only a small slice needs to be decrypted and encrypted, the computational cost required for access policy update is smaller than when computing over the original data. However, Cheng et al.'s scheme still requires downloading and uploading some data slices upon policy update. More importantly, as showed by Wang et al. [Wang, Wu and Liu (2018)], Cheng et al.'s scheme only preserves all-or-nothing property for one time, if a user has obtained and kept the symmetric key before being denied access, updating access policy will not take effect because it only affects ABE ciphertext, and the symmetric key encrypting the actual data does not change.

Wang et al. [Wang, Mickens, Zeldovich et al. (2016)] proposed a hybrid encryption system in which CP-ABE is used with key homomorphic pseudo-random function so that the data can be re-encrypted by a proxy server. However, as pointed out by Garisson et al. [Garrison, Shull, Myers et al. (2016)], the key homomorphic pseudo-random function has efficiency issues with its computational cost when re-encrypting symmetric ciphertext under a new key, making it impractical for deployment in actual systems. Myers et al. [Myers and Shull (2018)] proposed a general hybrid encryption method in which only a small fraction of data is computed upon re-encryption. However, their method has a drawback where the ciphertext size and decryption time increase per re-encryption. In addition, their method does not consider the case in which data owner may want to change access policy of the ciphertext.

Syalim et al. [Syalim, Nishide and Sakurai (2011)] proposed an idea to implement a proxy re-encryption scheme for symmetric ciphers by first transforming the plaintext into a random sequence of blocks by using AONT. However, the security proof assumes that the AONT always produces random sequence, which is not applicable if the users who have access to the previous encryption keys are allowed to choose the plaintexts. Subsequently, Syalim et al. [Syalim, Nishide and Sakurai (2017)] proposed an improved version by introducing the usage of a variant of Rivest' AONT, and proved the improved proxy re-encryption scheme for symmetric ciphers to be secure under chosen plaintext attack for all types of attackers.

Recently, Yasumura et al. [Yasumura, Imabayashi and Yamana (2018)] proposed an attribute-based proxy re-encryption method in which data can be re-encrypted in the cloud without downloading any data by adopting both Waters's CP-ABE scheme and Syalim et al.'s symmetric proxy re-encryption scheme. However, they did not give formal definition and security model, nor did they consider how to apply Waters's CP-ABE encrypted symmetric key to Syalim et al.'s symmetric proxy re-encryption. In addition, Waters's CP-ABE scheme is only be proved secure in the weak selective security model under non-standard q-type assumption. More importantly, the decryption procedure in Waters's CP-ABE scheme is fairly expensive, particularly for complex access structures, because a pairing computation was needed for each attribute.

In this paper, we propose a cryptographic notion called access policy updatable ciphertext-policy attribute-based encryption (APU-CP-ABE). When an access policy update occurs, data owner is no longer required to download any data for re-encryption from the cloud, all he needs to do is generate a re-encryption key and produce a new encapsulated symmetric key, and then upload them to the cloud. The cloud will store encrypted data for users accessing and execute re-encryption algorithm by using the re-encryption key, which does not give it the ability of decrypting any ciphertexts. Data users who do not meet the new access policy can no longer decrypt the data, even if they retain old symmetric keys before their access permissions are denied, as the data is encrypted under completely new key after re-encryption. We also present an efficient and provable secure APU-CP-ABE construction by combining Agrawal et al.'s fast attribute-based message encryption scheme [Agrawal and Chase (2017)] with Syalim et al.'s improved proxy re-encryption scheme for symmetric ciphers [Syalim, Nishide and Sakurai (2017)].

The rest of the paper is organized as follows. We introduce some necessary preliminary work in Section 2. We give system architecture and syntax definition for APU-CP-ABE scheme in Section 3. We describe our APU-CP-ABE construction in Section 4. Finally, we conclude our paper in Section 5.

2 Preliminaries

Let Ω denote the universe of attributes. A collection $\Gamma \subseteq 2^{\Omega}$ is monotone if for every $\mathcal{B}$ and $\mathcal{C}$, if $\mathcal{B} \in \Gamma$ and $\mathcal{B} \subseteq \mathcal{C}$ then $\mathcal{C} \in \Gamma$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) Γ of non-empty subsets of Ω. The sets in Γ are called the authorized sets, and the sets not in Γ are called the unauthorized sets.

Access control policy is generally described in terms of Boolean formula with AND and OR gates, where each input is associated with an attribute in Ω. Boolean formulae fall into a more general class of functions called monotone span programs (MSPs) [Rouselakis and Waters (2013)]. An MSP is given by a matrix $M_{\ell\times n}$ over $\mathbb{Z}_q$ and a mapping $\rho: \{1, 2, \cdots, \ell\} \to \Omega$. Lewko and Waters describe a simple and efficient method to convert any (monotone) Boolean formula into an MSP $(M_{\ell\times n}, \rho)$ such that every row of $M_{\ell\times n}$ corresponds to an input in the Boolean formula and the number of columns is same as the number of AND gates in the Boolean formula [Lewko and Waters (2011)].

Let $\mathcal{S}$ be a set of attributes and $\mathcal{I} = \{i \,\middle|\, i \in \{1, 2, \cdots, \ell\}, \rho(i) \in \mathcal{S}\}$ be the set of rows in $M_{\ell\times n}$ that belong to $\mathcal{S}$. We say that $(M_{\ell\times n}, \rho)$ accepts $\mathcal{S}$ if there exists a linear combination of rows in $\mathcal{I}$ that gives $(1, 0, \cdots, 0)$. More formally, there should exist coefficients $\{\mu_i\}$

such that

$\sum_{i\in\mathcal{I}} \mu_i \times M_i = (1, 0, \cdots, 0)$.

An asymmetric bilinear pairing group generator is an algorithm that takes as input a security parameter κ and outputs $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \widetilde{e})$, where $\mathbb{G}_1 =< g_1 >$, $\mathbb{G}_2 =< g_2 >$, $\mathbb{G}_T$ are three cyclic groups of prime order q, and a bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ is a function $\widetilde{e}: \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with the following properties:

- Bilinearity: for all $u_1 \in \mathbb{G}_1$, $u_2 \in \mathbb{G}_2$ and any positive integer a and b, we have $\widetilde{e}\left(u_1^a,\ u_2^b\right) = \widetilde{e}\,(u_1, u_2)^{ab}$.
- Non-degeneracy: $\tilde{e}(g_1, g_2) \neq 1$.
- Computability: for all $u_1 \in \mathbb{G}_1, u_2 \in \mathbb{G}_2$, there is an efficient algorithm to compute $\tilde{e}(u_1, u_2)$.

Definition 2.1 (Decisional Linear Assumption). The decisional linear (DLIN) assumption in an asymmetric pairing group of prime order $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \widetilde{e})$ states that, given: $\mathcal{D} = \left(g_1, g_2, g_1^{a_1},\ g_1^{a_2}, g_2^{a_1}, g_2^{a_2}, g_1^{a_1 s_1}, g_1^{a_2 s_2}, g_2^{a_1 s_1}, g_2^{a_2 s_2}\right)$, it is computationally intractable for any polynomial-time adversary to distinguish the tuple $\left(\mathcal{D},\ \left(g_1^{s_1+s_2},\ g_2^{s_1+s_2}\right)\right)$ from the tuple $\left(\mathcal{D},\ (g_1^s,\ g_2^s\ )\right)$, where a_1 and a_2 are chosen from $\mathbb{Z}_q^*$, s_1, s_2 and s are chosen from $\mathbb{Z}$ at random.

The concept of All-Or-Nothing Transform (AONT) was introduced by Rivest as a mode of operation for block ciphers [Rivest (1997)]. An AONT is an un-keyed, invertible, randomized transformation, with the property that it is hard to invert unless all of the output is known. In fact, Rivest's AONT may be viewed as a $(t + 1,\ t + 1)$ threshold scheme. Data composed of t blocks is encoded into $t + 1$ different blocks so that none of the original blocks may be decoded unless all $t + 1$ encoded blocks are present, or an attacker possesses enough computing power to crack an encryption key. The key, however, is encoded with the data.

3 Access policy updatable ciphertext-policy attribute-based encryption

The system architecture and work flow of our proposed access policy updatable ciphertext-policy attribute-based encryption (APU-CP-ABE), as illustrated by Fig.1, considers a CSP stores data generated by data owners and shares them among authorized data users. A ciphertext includes two parts: a header which is encapsulated symmetric key by applying ciphertext-policy attribute-based key encapsulation mechanism and a body which is encrypted data by applying a symmetric cipher.

The following participating entities are involved in the APU-CP-ABE scheme:

- The trusted attribute authority (AA), who is responsible for generating the global public parameters, master public key and issuing attribute-based private keys for data users. AA is considered as a trusted entity in our model.
- The Cloud Service Provider (CSP), who stores and shares data among authorized users. CSP is also responsible for executing the re-encryption algorithm for symmetric cipher to obtain a new ciphertext body and replacing the old ciphertext header with the new ciphertext header under the data owner's request.

- Data owner (DO), who sets access policy, generates a ciphertext header by running symmetric key encapsulation algorithm under the access policy, and a ciphertext body by running symmetric key encryption algorithm on original data before outsourcing to the cloud. In addition, the data owner runs re-encryption key generation algorithm and sends the re-encryption key, and new ciphertext header (with new access policy) to the cloud when the access policy is updated.
- Data users (DU), who downloads the ciphertext from the cloud. Only users whose attribute-based private keys satisfy the access policy embedded in ciphertext header can successfully decrypt the symmetric key, then decrypt ciphertext body to get original data.

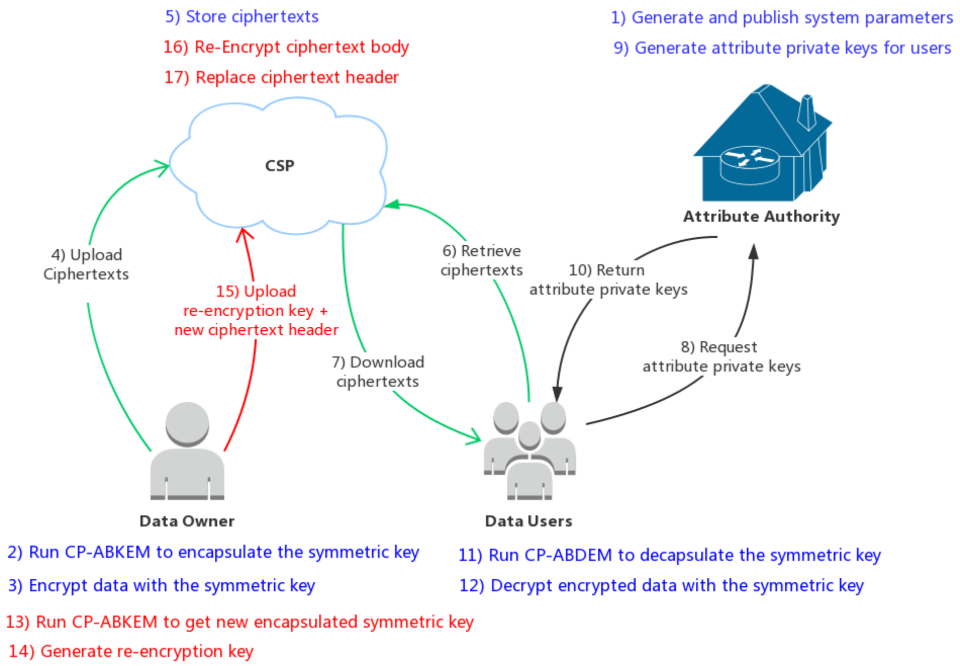
5) Store ciphertexts
16) Re-Encrypt ciphertext body
17) Replace ciphertext header
1) Generate and publish system parameters
9) Generate attribute private keys for users
CSP
Attribute Authority
4) Upload Ciphertexts
15) Upload re-encryption key + new ciphertext header
6) Retrieve ciphertexts
10) Return attribute private keys
7) Download ciphertexts
8) Request attribute private keys
Data Owner
Data Users
2) Run CP-ABKEM to encapsulate the symmetric key
3) Encrypt data with the symmetric key
13) Run CP-ABKEM to get new encapsulated symmetric key
14) Generate re-encryption key
11) Run CP-ABDEM to decapsulate the symmetric key
12) Decrypt encrypted data with the symmetric key

Figure 1: Architecture and work flow of APU-CP-ABE

An APU-CP-ABE scheme consists of the following eight polynomial-time algorithms:

- $\mathrm{Setup}(1^\kappa, \Omega)$: The setup algorithm takes as input a security parameter κ and an attribute universe description Ω. It outputs the public parameters mpk and a master secret key msk.
- $\mathrm{AttrKeyGen}(\mathrm{mpk}, \mathrm{msk}, \mathcal{S})$: The randomized attribute-based private key generation algorithm takes as input the public parameters mpk, the master secret key msk and a set of attributes $\mathcal{S}$. It outputs an attribute-based private key $sk_{\mathcal{S}}$.
- $\mathrm{Encrypt}(K, \mathrm{msg})$: The randomized symmetric encryption algorithm takes as input a symmetric key K and a message msg. It produces as output a ciphertext body c.
- $\mathrm{CPABKEM}(\mathrm{mpk}, \mathbb{A})$: The randomized key encapsulation algorithm takes as input the

public parameters mpk and an access structure $\mathbb{A}$. It outputs a symmetric key K and a ciphertext header CT.

- $\mathrm{ReKeyGen}(K_1, K_2)$: The re-encryption key generation algorithm takes as input an old symmetric key K_1 and a new symmetric key K_2. It generates re-encryption key $rk_{K_1 \to K_2}$.
- $\mathrm{ReEncrypt}(c_1,\ rk_{K_1 \to K_2})$: The re-encryption algorithm takes as input a ciphertext body $c_1 = \mathrm{Encrypt}(K_1,\ \mathrm{msg})$ encrypted on a messge msg with the symmetric key K_1 and a re-encryption key $rk_{K_1 \to K_2}$. It outputs a new ciphertext body $c_2 = \mathrm{Encrypt}(K_2,\ \mathrm{msg})$ encrypted on the message msg with the symmetric key K_2.
- $\mathrm{CPABDEM}(\mathrm{mpk}, sk_{\mathcal{S}}, CT)$: The deterministic symmetric key de-capsulation algorithm takes as input the public parameters mpk, an attribute-based private key $sk_{\mathcal{S}}$ and a ciphertext header CT. If the attributes set $\mathcal{S}$ contained in the attribute-based private key satisfies the access policy $\mathbb{A}$ in the ciphertext header, this algorithm will output a symmetric key K. Otherwise, this algorithm will output an error symbol $\perp$ indicating that the de-capsulation failed.
- $\mathrm{Decrypt}(\ K, c)$: The deterministic symmetric decryption algorithm takes as input a symmetric key K and a ciphertext body c. It produces a message msg.

Correctness when the access policy is not updated: for security parameter κ and any message msg , $\mathrm{Setup}(1^{\kappa}, \Omega) \to (\mathrm{mpk},\ \mathrm{msk})$, $\mathrm{AttrKeyGen}(\mathrm{mpk}, \mathrm{msk}, \mathcal{S}) \to sk_{\mathcal{S}}$, $\mathrm{CPABKEM}(\mathrm{mpk},\ \mathbb{A}) \to (K, CT)$, $\mathrm{Encrypt}(K,\ \mathrm{msg}) \to c$, if $\mathbb{A}(\mathcal{S}) = 1$, we have $\mathrm{Decrypt}(\mathrm{CPABDEM}(\mathrm{mpk}, sk_{\mathcal{S}}, CT), c) \to \mathrm{msg}$ with overwhelming probability.

Correctness when the access policy is updated: for security parameter κ and any message msg, $\mathrm{Setup}(1^{\kappa}, \Omega) \to (\mathrm{mpk},\ \mathrm{msk})$, $\mathrm{CPABKEM}(\mathrm{mpk},\ \mathbb{A}_1) \to (K_1, CT_1)$, $\mathrm{CPABKEM}(\mathrm{mpk},\ \mathbb{A}_2) \to (K_2, CT_2)$, $\mathrm{ReKeyGen}(K_1, K_2) \to rk_{K_1 \to K_2}$, $\mathrm{Encrypt}(K_1,\ \mathrm{msg}) \to c_1$, $\mathrm{ReEncrypt}(c_1, rk_{K_1 \to K_2}) \to c_2$, and $\mathrm{AttrKeyGen}(\mathrm{mpk}, \mathrm{msk}, \mathcal{S}) \to sk_{\mathcal{S}}$. If $\mathbb{A}_2(\mathcal{S}) = 1$, then we have $\mathrm{Decrypt}(\mathrm{CPABDEM}(\mathrm{mpk},\ sk_{\mathcal{S}},\ CT_2),\ c_2) \to \mathrm{msg}$ with overwhelming probability, namely the correctness is not affected by symmetric key rotation.

Syalim et al. [Syalim, Nishide and Sakurai (2011)] defined the security model for symmetric proxy re-encryption. For ciphertext-policy attribute-based key encapsulation mechanism, we consider the following indistinguishability against adaptive chosen plaintext attack game played between a challenger and an adversary:

- Initialization: The challenger runs the setup algorithm and provides mpk to the adversary.
- Attribute-based private key queries: The adversary can query AttrKeyGen oracle, for any attribute $\mathcal{S}$ of its choice to get $sk_{\mathcal{S}}$.
- Challenge: The adversary provides a policy $\mathbb{A}$ to the challenger, then the challenger runs $\mathrm{CPABKEM}(\mathrm{mpk},\ \mathbb{A}) \to (K, CT)$, sets $K_b \leftarrow K$ and K_{1-b} as a random key, for a random bit b. It provides (K_0, K_1, CT) to the adversary.
- Attribute-based private key queries: The adversary can again query AttrKeyGen oracle of its choice.
- Finalize: The adversary outputs its guess b' on the bit b.

The adversary's advantage in the above game is defined as $\mathrm{Adv}(1^{\kappa}) = \Pr[b' = b] - 1/2$.

Definition 1. An APU-CP-ABE scheme is said to be fully or adaptively secure if $\mathrm{Adv}(1^{\kappa})$ is negligible in the security parameter κ for all probabilistic polynomial time adversaries.

4 Our APU-CP-ABE construction

Our proposed APU-CP-ABE scheme is described as follows.

- Setup: The AA performs as follows.

1. Run the asymmetric bilinear pairing group generator algorithm that takes security parameter κ as input and outputs $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \tilde{e})$.
2. Generate hash functions $H_{ij}: \{0,\ 1\}^* \to \mathbb{G}_1$ and $G_{ij}: \{0,\ 1\}^* \to \mathbb{G}_1$ for $i \in \{1,\ 2,\ 3\}$ and $j \in \{1,\ 2\}$.
3. Compute $g_{ij} = G_{ij}(1)$ for $i \in \{1,\ 2,\ 3\}$ and $j \in \{1,\ 2\}$.
4. Choose $(r, a_1, a_2, b_1, b_2, d_1, d_2, d_3)$ from $\mathbb{Z}_q^*$ at random, then compute $g = g_1^r$, $h_1 = g_2^{a_1}$, $h_2 = g_2^{a_2}$, $T_1 = \tilde{e}(g_1, g_2)^{a_1 d_1 + d_3}$, $T_2 = \tilde{e}(g_1, g_2)^{a_2 d_2 + d_3}$.
5. Set the master public key mpk as
$(\kappa, q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \tilde{e}, g_1, g_2, g_{11}, g_{12}, g_{21}, g_{22}, g_{31}, g_{32}, h_1, h_2, T_1, T_2)$.
6. Set the master secret key msk as $(g, a_1, a_2, b_1, b_2, d_1, d_2, d_3)$.

- AttrKeyGen: The AA performs as follows.

1. Choose (r_1, r_2, σ) from $\mathbb{Z}_q^*$ at random and compute $x_1 = g_2^{b_1 r_1}$, $x_2 = g_2^{b_2 r_2}$, $x_3 = g_2^{r_1 + r_2}$, $y_1 = g_{11}^{b_1 r_1/a_1} g_{21}^{b_2 r_2/a_1} g_{31}^{r_1 + r_2/a_1} g^{\sigma/a_1 + d_1}$, $y_2 = g_{12}^{b_1 r_1/a_2} g_{22}^{b_2 r_2/a_2} g_{32}^{r_1 + r_2/a_2} g^{\sigma/a_2 + d_2}$, $y_3 = g^{d_3 - \sigma}$.
2. For a set of attributes $\mathcal{S} = \{\omega_1, \omega_2, \cdots, \omega_n\}$, choose σ_i from $\mathbb{Z}_q^*$ at random, then compute $sk_{i1} = H_{11}(\omega_i)^{b_1 r_1/a_1} H_{21}(\omega_i)^{b_2 r_2/a_1} H_{31}(\omega_i)^{r_1 + r_2/a_1} g^{\sigma_i/a_1}$, $sk_{i2} = H_{12}(\omega_i)^{b_1 r_1/a_2} H_{22}(\omega_i)^{b_2 r_2/a_2} H_{32}(\omega_i)^{r_1 + r_2/a_2} g^{\sigma_i/a_2}$ and $sk_{i3} = g^{-\sigma_i}$.
3. Set $sk_i = (sk_{i1}, sk_{i2}, sk_{i3})$ for $1 \leq \imath \leq n$.
4. Return $sk_{\mathcal{S}} = (x_1, x_2, x_3, y_1, y_2, y_3, sk_1, sk_2, \cdots, sk_n)$..

- CP-ABKEM: For a MSP $(M_{\ell \times n}, \rho)$, the data owner performs as follows.

1. Choose u_1 and u_2 from $\mathbb{Z}_q^*$ at random, then compute $z_1 = h_1^{u_1}$, $z_2 = h_2^{u_2}$, $z_3 = g_2^{u_1 + u_2}$, $K_1 = H_1(T_1^{u_1} T_2^{u_2})$, $K_2 = H_2(T_1^{u_1} T_2^{u_2})$and $K_3 = H_3(T_1^{u_1} T_2^{u_2})$.
2. For $i \in \{1, 2, \cdots, \ell\}$ and $k \in \{1, 2, 3\}$, compute
$c_{ik} = [H_{k1}(\rho(i))]^{u_1} [H_{k2}(\rho(i))]^{u_2} \prod_{j=1}^{n} [G_{k1}(j)^{u_1} G_{k2}(j)^{u_2}]^{M_{ij}}$.
3. Set $c_i = (c_{i1}, c_{i2}, c_{i3})$ for $1 \leq \imath \leq n$.
4. Set the ciphertext header $CT = (z_1, z_2, z_3, c_1, c_2, \cdots, c_\ell)$.
5. Return (CT, K_1, K_2, K_3).

- Encrypt: The data owner encrypts the message msg using the keys (K_1, K_2, K_3) to get ciphertext body c, which is exactly the same as the encryption algorithm in Syalim et al. [Syalim, Nishide and Sakurai (2017)].

- CP-ABDEM: For attribute-based private key $sk_{\mathcal{S}}$ and a ciphertext header CT, the data user performs as follows.
  1. Determine whether the attribute set $\mathcal{S}$ associated with sk_{S} satisfies the MSP $(M_{\ell\times n},\rho)$ in CT. If not, terminate and return $\perp$.
  2. Otherwise, compute constants $\{\mu_i\}_{i\in\mathcal{I}}$ that satisfy $\sum_{i\in\mathcal{I}}\mu_i M_i = (1,0,\cdots,0)$.
  3. Compute $t_1 = y_1\prod_{i\in\mathcal{I}} sk_{\rho(i)1}^{\mu_i}$, $t_2 = y_2\prod_{i\in\mathcal{I}} sk_{\rho(i)2}^{\mu_i}$, $t_3 = y_3\prod_{i\in\mathcal{I}} sk_{\rho(i)3}^{\mu_i}$, $v_1 = \prod_{i\in\mathcal{I}} c_{i1}^{\mu_i}$, $v_2 = \prod_{i\in\mathcal{I}} c_{i2}^{\mu_i}$, $v_3 = \prod_{i\in\mathcal{I}} c_{i3}^{\mu_i}$, $U = [\,\tilde{e}(t_1,z_1)\tilde{e}(t_2,z_2)\tilde{e}(t_3,z_3)]/[\tilde{e}(v_1,x_1)\tilde{e}(v_2,x_2)\tilde{e}(v_3,x_3)]$, $K_1 = H_1(U)$, $K_2 = H_2(U)$ and $K_3 = H_3(U)$.
  4. Return (K_1,K_2,K_3).
- Decrypt: The data user decrypt the ciphertext body c using the keys (K_1,K_2,K_3) to recover original data msg, which is exactly the same as the decryption algorithm in Syalim et al. [Syalim, Nishide and Sakurai (2017)].
- ReKeyGen: The data owner generates the re-encryption keys from old symmetric key (K_1,K_2,K_3) and a new symmetric key (K_1',K_2',K_3'), which is exactly the same as the re-encryption key generation algorithm in Syalim et al. [Syalim, Nishide and Sakurai (2011)].
- Re-Encrypt: The CSP re-encrypts the ciphertext body c using the re-encryption key, which is exactly the same as the re-encryption key generation algorithm in Syalim et al. [Syalim, Nishide and Sakurai (2011)].

Our proposed APU-CP-ABE scheme has three advantages over the trivial solution (i.e., the hybrid encryption of CP-ABE and AES in which the process of downloading, decrypting, encrypting, and uploading ciphertexts is required to perform re-encryption when access policy update occurs).

- First, the burden on the data owner is reduced because neither decryption nor encryption are required. The data owner only needs to generate and send re-encryption keys to the cloud.
- Second, the communication cost required for re-encryption is smaller because the data owner only needs to send re-encryption key and encapsulated symmetric key to the cloud.
- Third, data users who do not meet the new access policy can no longer decrypt the data, even if they retain old symmetric keys before their access permissions are denied, as the data is encrypted under completely new key after re-encryption.

In addition, compared to Yasumura et al.'s scheme [Yasumura, Imabayashi and Yamana (2018)] that based on Waters's CP-ABE scheme and Syalim et al.'s symmetric proxy re-encryption scheme, Waters's CP-ABE scheme is only be proved secure in the weak selective security model under non-standard q-type assumption. More importantly, the decryption procedure in Waters's CP-ABE scheme is fairly expensive, particularly for complex access structures, because a pairing computation was required for each attribute. As we all know, the decryption procedure for a CP-ABE scheme is arguably the most important one because it will be invoked by the user in most cases, and often on computationally weak devices. As a result, the decryption process in Yasumura et al.'s scheme is not efficient. In our APU-CP-ABE construction, it requires only 6 bilinear pairing operations for decryption, regardless of the number of attributes involved. This renders it particularly attractive when decryption is time-critical. Furthermore, our APU-

CP-ABE construction is proved to be adaptively secure under the DLIN assumption on asymmetric pairing groups in the random oracle model [Agrawal and Chase (2017)].

5 Conclusions

Ciphertext-policy atrribute-based encryption is a promising cryptographic tool in the cloud environment, and several CP-ABE based cryptographic cloud storage systems have been proposed in recent years. However, these systems suffer from time-consuming access policy revocation operation. In this paper, we propose a notion of access policy updatable ciphertext-policy attribute-based encryption by combining the idea of ciphertext-policy attribute-based key encapsulation and symmetric proxy re-encryption. When an access policy update occurs, data owner only needs to do generate a re-encryption key and produce a new encapsulated symmetric key, and then upload them to the cloud. This reduces the communication costs between the data owner and the cloud. We present a concrete access policy updatable ciphertext-policy attribute-based encryption construction based on Syalim et al.'s improved symmetric proxy re-encryption scheme and Agrawal et al.'s attribute-based message encryption scheme, and our construction requires only 6 bilinear pairing operations for decryption, regardless of the number of attributes involved. This makes it particularly attractive when decryption is time-critical.

Acknowledgement: This research is funded by Science and Technology Program of Guangzhou (Grant No. 201707010358).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

Agrawal, S.; Chase, M. (2017): Fame: fast attribute-based message encryption. *Proceedings of the 24th ACM Conference on Computer and Communications Security*, pp. 665-682.

Cheng, Y.; Wang, Z.; Ma, J.; Wu, J.; Mei, S. et al. (2013): Efficient revocation in ciphertext-policy attribute-based encryption based cryptographic cloud storage. *Journal of Zhejiang University SCIENCE C*, vol. 14, no. 2, pp. 85-97.

Garrison, W.; Shull, A.; Myers, S.; Lee, A. (2016): On the practicality of cryptographically enforcing dynamic access control policies in the cloud. *IEEE Symposium on Security and Privacy*, pp. 819-838.

Goyal, V.; Pandey, O.; Sahai, A.; Waters, B. (2006): Attribute-based encryption for fine-grained access control of encrypted data. *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pp. 89-98.

Lewko, A.; Waters, B. (2011): Unbounded hibe and attribute-based encryption. *Advances in Cryptology-EUROCRYPT 2011, Lecture Notes in Computer Science*, vol. 6632, pp. 547-567.

Liu, Y.; Peng, H.; Wang, J. (2018): Verifiable diversity ranking search over encrypted outsourced data. *Computers, Materials & Continua*, vol. 55, no. 1, pp. 37-57.

Myers, S.; Shull, A. (2018): Practical revocation and key rotation. *Topics in Cryptology-CT-RSA 2018, Lecture Notes in Computer Science*, vol. 10808, pp. 151-178.

Rivest, R. (1997): All-or-nothing encryption and the package transform. *International Workshop on Fast Software Encryption, Lecture Notes in Computer Science*, vol. 1267, pp. 210-218.

Rouselakis, Y.; Waters, B. (2013): Practical constructions and new proof methods for large universe attribute-based encryption. *Proceedings of the 20th ACM Conference on Computer and Communications Security*, CCS'13, pp. 463-474.

Sadiku, M.; Musa, S.; Momoh, O. (2014): Cloud computing: opportunities and challenges. *IEEE Potentials*, vol. 33, no. 1, pp. 34-36.

Sahai, A.; Waters, B. (2005): Fuzzy identity based encryption. *Advances in Cryptology-EUROCRYPT, Lecture Notes in Computer Science*, vol. 3494, pp. 457-473.

Syalim, A.; Nishide, T.; Sakurai, K. (2011): Realizing proxy re-encryption in the symmetric world. *Informatics Engineering and Information Science, Communications in Computer and Information Science*, vol. 251, pp. 259-274.

Syalim, A.; Nishide, T.; Sakurai, K. (2017): Improved proxy re-encryption scheme for symmetric key cryptography. *IEEE International Workshop on Big Data and Information Security*, pp. 105-111.

Wang, C.; Wu, J.; Liu, J. (2018): Insecurity of Cheng et al.'s efficient revocation in ciphertext-policy attribute-based encryption based cryptographic cloud storage. *2017 IEEE International Conference on Ubiquitous Computing and Communications*, pp. 1387-1393.

Wang, F.; Mickens, J.; Zeldovich, N.; Vaikuntanathan, V. (2016): Sieve: cryptographically enforced access control for user data in untrusted clouds. *Proceedings of the 13th Usenix Conference on Networked Systems Design and Implementation*, NSDI'16, pp. 611-626.

Waters, B. (2011): Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. *Public Key Cryptography-PKC 2011, Lecture Notes in Computer Science,* vol. 6571, pp. 53-70.

Yasumura, Y.; Imabayashi, H.; Yamana, H. (2018): Attribute-based proxy re-encryption method for revocation in cloud storage: reduction of communication cost at re-encryption. *IEEE International Conference on Big Data Analysis*, pp. 312-318.