Time-Based Direct Revocable Ciphertext-Policy Attribute-Based Encryption with Short Revocation List*

Joseph K. Liu[1], Tsz Hon Yuen[2], Peng Zhang[3], Kaitai Liang[4]

[1]Faculty of Information Technology, Monash University, Australia
Email: `joseph.liu@monash.edu`
[2] Huawei, Singapore
[3] Shenzhen University, China
[4] University of Surrey, UK

Abstract. In this paper, we propose an efficient revocable Ciphertext-Policy Attribute-Based Encryption (CP-ABE) scheme. We base on the direct revocation approach, by embedding the revocation list into ciphertext. However, since the revocation list will grow longer as time goes by, we further leverage this by proposing a secret key time validation technique so that users will have their keys expired on a date and the revocation list only needs to include those user keys revoked before their intended expired date (e.g. those user keys which have been stolen before expiry). These keys can be removed from the revocation list after their expiry date in order to keep the revocation list short, as these keys can no longer be used to decrypt ciphertext generated after their expiry time. This technique is derived from Hierarchical Identity-based Encryption (HIBE) mechanism and thus time periods are in hierarchical structure: year, month, day. Users with validity of the whole year can decrypt any ciphertext associated with time period of any month or any day within the year. By using this technique, the size of public parameters and user secret key can be greatly reduced. A bonus advantage of this technique is the support of discontinuity of user validity (e.g. taking no-paid leave).

1 Introduction

Attribute-Based Encryption (ABE) is a generalization of Identity-Based Encryption (IBE) [13,7,6,15,14,49]. It provides flexibility of data sharing for system users in the sense that a data encryptor is allowed to specify some descriptive values x for an encryption and thus, the encryption can be decrypted successfully by a secret key associated with some descriptive values y matching x. ABE has many network applications, such as cloud computing [25,36], cloud storage systems [22,23,35,51] and medical e-healthcare systems [39,24,19,8] etc. ABE can be classified into two different types: one is Key-Policy ABE (KP-ABE), and the other is Ciphertext-Policy ABE (CP-ABE). In a KP-ABE system, ciphertexts

* The short version of this paper was published in ACNS 2018.

are associated with attribute sets and secret keys are associated with access policies. On the opposite side, in a CP-ABE scheme ciphertexts are related to access policies, and attribute sets are tagged with secret keys.

Before deploying ABE into any practical scenarios, one has to solve for the user revocation problem. No organization will be happy to see that any of its revoked users can still be able to decrypt the document designated for its users.

When we talk about revocability in ABE, there are different levels of revocation:

1. **Attribute Revocation.** In this case, the user has changed his/her attributes. For example, the original attributes for Alice are {`Physics`, `Student`, `University A`}. Now she has changed to {`Chemistry, Student`, `University A`}. Therefore, the original `Physics` attribute should be revoked (but not `Student` or `University A`).
2. **User Revocation.** In this case, the user has left the organization. That is, all attributes have to be revoked. In the previous example of Attribute Revocation, Alice has left University A and therefore the user Alice should be revoked.
3. **Key Revocation.** In this case, the secret key of the user is revoked. This is the most generalized level of revocation. This can be happened in different scenarios. For example, the user has left the organization (case (2)). Or if the user has lost his secret key (and got a replacement key), the old one has to be revoked. It can also cover case (1): If Alice changes her attribute from `Physics` to `Chemistry`, her old key is revoked while she has been issued with a new key associated with her new attribute.

   Usually in this case the key is also associated with an identity or a serial number, which is used in the revocation process by the authority. The encryptor does not need to know this identity or number. In the rest of this paper, we refer to *revocation* as this case (key revocation).

1.1 Different Approaches for Revocable ABE

There are several approaches to tackle the problem of revocability in ABE:

1. **Key Update for Non-Revoked Users**. This is also called the *Indirect Approach*. In this approach, every user has a secret key with a state. The authority will execute a key update algorithm for every non-revoked users. The keys for revoked users will not be updated. Upon the update, a new state will be issued to the secret key. Ciphertext is generated according to the current state. Therefore those revoked users who only have the secret keys of previous state(s) cannot decrypt the ciphertext which is associated with the new state.

   If the ciphertext is stored on cloud, those revoked users however are still able to decrypt the old ciphertext (generated with previous states). In order to hinder this vulnerability, a ciphertext update algorithm can be executed so that all old ciphertexts will be modified to associate with the current state. In this way, revoked users (whose secret keys are associated with an old state) can no longer decrypt any old ciphertext from the cloud as it has been updated to the current state which is only decryptable by a secret key with the new state.

Nevertheless we only regard the ciphertext update as an optional feature, as we have no way to prevent a revoked user from downloading the ciphertext (and thus decrypting the ciphertext) before he has been revoked.

The main issue of the key update approach is the inability of instant user revocation. Suppose Alice is revoked now and the next key update algorithm is scheduled at the 1st of coming month. Alice is still able to decrypt any newly generated ciphertext from now to the end of this month. (Even if the cloud is equipped with ciphertext update, Alice cannot decrypt only those old ciphertext but still can decrypt those newly generated one.) One may try to argue that the authority may execute the key update algorithm once there is any revoked user. In the point of view of security, this action can block the loophole of inability instant user revocation. Nevertheless, it is definitely not practical especially if there are a large number of non-revoked users. In a large organization, there may be a new revoked user in every hour or even every minute. It is impossible to ask every non-revoked user to update their key every minute! If we schedule a longer key update period, the instant user revocation problem will be worse then.

2. **Embedding Revocation List into Ciphertext**. This is also called the *Direct Approach*. In this approach, there is a public revocation list. The encryptor embeds the latest revocation list into the ciphertext so that only those users not in the revocation list and satisfying the predicate (attributes-policy matching) can decrypt the ciphertext. This approach can provide instant user revocation (and thus solve the problem from the indirect approach). No key update is required in this approach.

   However, there is another practical problem with this approach. The revocation list will grow longer as time goes by. If the organization is large, the revocation list will become an efficiency bottleneck for the encryption and decryption as it will continue to grow all the time. There is no way to reduce or delete the revocation list, unless the revoked user re-joins the organization in the future. This is not likely to happen in most of the scenarios though.
3. **Cloud-Assisted**. Another approach is to make use of the cloud assistance. In this approach, the decryption ability is split into two halves. The first half is owned by the user while another half is owned by the cloud. The cloud needs to partially decrypt the ciphertext into an intermediate data first, which is then sent to the user for the second level decryption. If the user is revoked, the cloud refuses to execute the first level decryption. Then the revoked user cannot decrypt the ciphertext without the assistance from the cloud.

   This is the simplest way to achieve user revocation for ABE. *In spite of that, the cloud will be very busy if the number of users is large*, as the decryption of every user requires the assistance from the cloud.

More examples on each approach will be given in Section 2.

1.2 A Naïve Approach

One may immediately think of a naïve approach by combining the indirect and direct approaches together in order to possess the merits from both sides. Intuitively the simple combination is to use a key update ABE (the indirect revocable ABE) to encrypt the plaintext first into the first-level ciphertext. Then the resulting ciphertext is further encrypted using another ABE with revocation list embedded into the ciphertext (the direct revocable ABE) as the second-level. If a user is revoked before the next key update period, since his identity has been put into the revocation list embedded into the second-level ciphertext (generated by the direct revocable ABE), he cannot decrypt it. On the other side, if the revoked user's key has been expired (that is, not being updated as it has been revoked), his identity is not needed to be put into the revocation list by the direct revocable ABE as the revoked key cannot be used to decrypt the first-level ciphertext (generated by the indirect revocable ABE) even though it can still decrypt the second level ciphertext. In this way, the revocation list can be kept short while instant revocation can be achieved and thus no frequent key update is required.

This naïve approach seems working fine, if we do not consider collusion attack. Simply speaking, collusion attack in ABE refers to two different users who both cannot decrypt the ciphertext individually but they can succeed to do so if they are colluding with each other. Suppose Alice is revoked just right now (before the next key update period) and therefore her identity `Alice` is in the revocation list embedded into the second-level ciphertext. Bob is another revoked user who was revoked in the previous time period. Thus his key has not been updated in the current time period and his identity `Bob` is not in the revocation list embedded into the second-level ciphertext. It is obviously that both Alice and Bob cannot decrypt the combined ciphertext individually. However, if they are working together, they can successfully decrypt it: Bob can use his secret key to decrypt the second-level ciphertext (as his identity is not in the revocation list) and he can pass the resulting intermediate data (which has become the first-level ciphertext) to Alice. Alice can use her secret key to decrypt this first-level ciphertext as her key is the most updated one. As a result, Alice and Bob can get the final plaintext if they are working together.

We definitely have to avoid this kind of collusion attack in any circumstance.

1.3 Our Contribution

In this paper, we propose an efficient and practical revocable CP-ABE scheme. Our scheme is motivated from the (non-revocable) CP-ABE given in [37] and incorporates the merit from all existing approaches for revocation, yet we do not have the emerged trade-off and we are immune to the collusion attack mentioned above in the naive approach. Namely,

- We have a revocation list, which is used by the encryptor to be embedded into the ciphertext. On the other side, we also have a key update process for non-revoked users at a *reasonable* interval (e.g. once every two years). The revocation list only contains those revoked users whose keys are not expired yet. If their keys are expired (they are not allowed to update their keys as they are revoked), they should not be able to decrypt any ciphertext generated after their expiry date even though they are not included in the revocation list. In other words, they can be removed from the revocation list after the expiry date of their keys and thus *the revocation list can be kept short*.

- Although we still require a key update process for all non-revoked users, we do not need to execute it frequently in order to provide instant key revocation. We argue that most organizations will require their users to renew their contracts at a reasonable time interval (e.g. once every two years). It is practical to renew their secret keys at the time they are renewing their contracts. This time does not need to be synchronized. Every user may have his own time for expiry and renewal.
- We do not need the cloud in our basic system (a cloud-free system). Extension can be made to provide ciphertext update in the cloud (e.g. using the ciphertext update technique in [48,32]) so that revoked users are no longer able to decrypt any ciphertext generated in the past.
- We are immune to the collusion attack mentioned in the naïve approach. Suppose there is a set of users. Anyone in this set cannot decrypt the ciphertext individually. They cannot succeed to do so even if they are working together, or by someone who has the secret keys for all users in this set.
- We use Hierarchical Identity-based Encryption (HIBE) technique to further shorten the size of user secret key. Our time period is *hierarchical*. That is, we have *year*, *month* and *day*. A user with secret key valid for the whole year can derive the key with validity for the underlying months of that year. A user with secret key valid for the whole month can derive the key with validity for the underlying days of that month. With this technique, we can further support discontinuity of user validity, which is believed as a common scenario in the practical world (e.g. no-paid leave).

We provide a concrete construction for our proposed scheme. The size of the ciphertext only depends on the embedded policy (access structure) but NOT the revocation list, though the size of the secret key is linear with the maximum length of the revocation list and the number of attributes of the user. In the decryption, the number of pairing operations only depends on the access structure but NOT the number of users in the revocation list or the total number of users in the system. The performance of our construction enjoys a significant improvement over other similar schemes.

Table 1 gives a functional comparison between other approaches and our proposed approach.

Table 1: Features Comparison

Approach	Examples	Features				
		Instant Revoke	No Freq. Key Update	Cloud Free	Ciphertext Update	Short Revo. lt.
Key Update	[11], [2], [48], [32], [40], [41], [27], [30], [28]	×	×	✓	Optional	✓
Embedding Revocation List	[3], [2], [5], [34], [29] [9], [50], [17], [26], [18]	✓	✓	✓	Optional	×
Cloud-Assisted	[31], [21], [44], [33], [45], [16], [46]	✓	✓	×	✓	×
Our Approach		✓	✓	✓	Optional	✓

2 Related Works

There are several schemes in the literature addressing the problem of revocation in ABE. We briefly describe them using the classification in Section 1.[1]

1. **Key Update for Non-Revoked Users (Indirect) approach**. The first revocable ABE was proposed by Boldyreva et al. [11]. It is a KP-ABE. Revocation list is stored in the authority which executes key update algorithm with each non-revoked users (those users not in the revocation list) during a regular time interval. Revoked users (without having their keys updated) cannot decrypt any newly generated ciphertext. Yu et al. [48] proposed another revocable ABE in the context of CP-ABE using key update approach. In addition, they provide a mechanism to update ciphertext so that revoked users cannot decrypt the updated ciphertext in the cloud. Nevertheless, they support policies with logical AND only. A more promising construction was given by Sahai et al. [32]. They provided constructions for both KP-ABE and CP-ABE. There are key update and ciphertext update algorithms. Ciphertext is decryptable only if the encryption time $t < t'$ where t' is the key expiry time. If the user is revoked, ciphertext will be updated so that the newly revoked users cannot decrypt those old ciphertext (those ciphertext generated before the user is revoked). All other users will then run the key update algorithm except the revoked user.

   Later on, Xie et al. [40] (the full version of [41]), proposed a revocable CP-ABE. In their construction, each user has two keys. There is an individual key and also a group key. A group is defined as a set of users with the same attributes. Decryption requires to have the group key and the individual user secret key. Revocation is done by updating the keys of non-revoked users.

[1] We exclude the discussion for [47] as it is not an ABE scheme. They require every user to have a private key and public key. Public key is generated by the authority and private key is generated by the encryptor! The encryptor needs to use the private key to encrypt the message. This is not a formal ABE that we are considering and thus it is excluded in our discussion. We also exclude [42] and [43] as they are pointed out as insecure in [20].

Ciphertext update is also allowed as in [48]. Naruse et al. [27] proposed another CP-ABE with attribute update by the cloud server. The cloud re-encrypts the ciphertext and re-generates new secret key for users, who have updated their attributes. Similar approach has also been suggested in [30,28].

2. **Embedding Revocation List into Ciphertext (Direct) approach**. Another approach is called direct approach, which requires the encryptor to incorporate the revocation list in the ciphertext. In this way, users in the revocation list cannot decrypt the ciphertext even though their attributes/policy satisfy the policy/attributes associated with the ciphertext. No key update is required using this approach. Attrapadung and Imai [3] proposed a revocable ABE by incorporating the revocation information into the ciphertext. Their approach is to use broadcast encryption. That is, the encryptor needs to know the identities of those *unrevoked* users. They provide a KP-ABE constructions. Later, Attrapadung et al. [5] proposed another revocable KP-ABE. Different from the previous one, this construction only requires the encryptor to know the identities of revoked users (instead of unrevoked users). Wang et al. [34] proposed another revocable KP-ABE using broadcast encryption which requires the encryptor to know the list of unrevoked users. Nieto et al. [29] generalize the revocability technique to Predicate Encryption. They require the encryptor to embed the revocation list into the ciphertext. They use dual pairing vector space as the primitive and thus the number of pairing operations in the decryption is very large (linear with the number of users in the system). Balu et al. [9] proposed a revocable CP-ABE by incorporating the revocation list. Encryptor only needs to know the identities of revoked users. However, their model is very weak. They only allow the adversary to query secret key that does not satisfy the challenge access structure AND not in the revocation list. In other words, it does not allow adversary to query a secret key that satisfies the access structure but in the revocation list. (This models the case for revoked users.) Zhang [50] proposed another revocable CP-ABE scheme using similar approach. In addition, the scheme also supports leakage resilient. But the construction only supports attribute-level revocation (the basic level of revocation). Datta et al. [17] proposed a KP-ABE construction using similar approach using multilinear maps. Liu and Wong [26] proposed a CP-ABE construction. Again they deploy similar approach but using matrix representation for users. Thus the size of ciphertext is of size $O(\sqrt{N})$, where N is the total number of users in the system. Recently, Datta et al. [18] proposed another KP-ABE that supports revocability with this approach. This time they use subset difference technique to achieve the purpose.

   There is also a construction using either direct and indirect approach. Attrapadung and Imai [2] proposed a KP-ABE which allows the encryptor to use whether direct or indirect mode (but not both).

3. **Cloud-Assisted approach**. With the assistance of cloud, revocation becomes easier without letting the encryptor to get the revocation list, or executing any key update for non-revoked users. Earlier stage cloud-assisted scheme [31] only provides ciphertext-update so that revoked users cannot

decryp the ciphertext in the cloud. Later on, other cloud-assisted schemes require all decryption must go through the cloud as a partial process. The cloud has the control to refuse the assistance of decryption for revoked users. Without the help from the cloud, no one can decrypt the ciphertext. Hur and Noh [21] proposed a revocable CP-ABE scheme. The data owner first outsourced the data into the cloud. The cloud then re-encrypts (using a double-encryption technique) according the set of authenticated (valid) membership. The revocation is easy. The cloud just deletes the revoked user from the membership (which is a binary tree). Decryption requires the cloud to process first. It then sends the partially decrypt data to the user. Yang et al. [44] and Shi et al. [33] proposed another cloud-assisted scheme independently. In their schemes, the decryption is split into two halves. The cloud stores the first half and the user stores another half. A complete decryption requires both parts. Yang et al. [45] use similar approach. They further reduce the trust on cloud by increasing the risk of collusion with users. Recently, Cui et al. [16] reduced the trust to the cloud server by letting the cloud server to use a kind of proxy re-encryption key only.

3 Definition

3.1 Time Period

A time period can be a day, a month or a year.[2] For example, we use "2016-Jun-15" to represent a day; "2016-Dec" to represent a month etc. Our scheme can also support some special case for non-continuity. For example, if the user is going to take no-paid leave from 01 August 2016 to 29 November 2016, then we can just assign the valid period from 15 June 2016 to 31 July 2016 and from 30 November 2016 to 31 December 2016. (Assume today is 15 June 2016 and the user expiries at the end of 2016.)

A decryptable time period is a time period set by the encryptor such that only users with validity completely covered the period can decrypt. (A specific setting is to let the encrypting time to be the decryptable time period.) For example, suppose the decryptable time period is December 2016 and the validity of user secret key is only limited to 31 December 2016. This is not a complete cover and thus this secret key is not able to decrypt. On the opposite, if the decryptable time period is 01 December 2016 and the validity of user secret key is December 2016, then it is able to decrypt as it has a complete cover for the decryptable time period (provided that other conditions are also satisfied).

3.2 Definition of Revocable Ciphertext-Policy Attribute-Based Encryption

A revocable ciphertext-policy attribute based encryption scheme consists of four algorithms: Setup, KeyGen, Encrypt, Decrypt.

[2] Our scheme can further support more levels of time period, e.g. hours, minutes etc. Yet for simplicity, we limit the description to the *day* level only.

- $\mathsf{Setup}(1^\kappa, U, R, \mathtt{T})$. Take as input the security parameter κ, the number of attributes in the system U, the maximum number of revoked users in the revocation list R and the depth of the time tree $\mathtt{T}$. It outputs the public parameters PK and a master key MK. We omit the description of κ in the rest of this paper.
- $\mathsf{KeyGen}(MK, ID, S, \mathsf{T})$. Take as input the master key MK, a user's identity ID[3], a set of attributes S and a range of validity time periods T. It outputs a private key $SK_{(ID,S,\mathsf{T})}$.
- $\mathsf{Encrypt}(PK, m, \mathsf{T}_c, \mathcal{R}, \mathbb{A})$. Take as input the public parameters PK, a message m, a decryptable time period T_c, a revoked set $\mathcal{R}$ and an access structure $\mathbb{A}$ over the universe of attributes. It outputs a ciphertext CT.
- $\mathsf{Decrypt}(PK, CT, \mathcal{R}, \mathbb{A}, \mathsf{T}_c, SK_{(ID,S,\mathsf{T})})$. Take as input the public parameters PK, a ciphertext CT, along with a description of a revoked set $\mathcal{R}$, an access policy $\mathbb{A}$ and a decryptable time period T_c, and a private key $SK_{(ID,S,\mathsf{T})}$, which is a private key for the user's identity ID, attributes set S and the range of validity time periods T. If the user's identity ID is not in the revoked set $\mathcal{R}$, the set S of attributes satisfies the access structure $\mathbb{A}$ and the range of validity time periods T completely covers the decryptable time period T_c, then the algorithm will decrypt the ciphertext and return a message m. Otherwise it outputs $\perp$.

Note that we do not explicitlly define the key update algorithm as its function can be implicitly covered by the KeyGen algorithm with the new validity time period.

3.3 Security Model

We now describe a selective security model for the revocable ciphertext-policy ABE scheme. The security model is described by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. In the game, $\mathcal{A}$ needs to submit an access structure $\mathbb{A}^*$, a revocation list $\mathcal{R}^*$ and a decryptable time period T_c^* to $\mathcal{C}$ before seeing the public parameter PK. At any time $\mathcal{A}$ can query for any private keys that cannot be used to decrypt the challenge ciphertext.

- **Init**. $\mathcal{A}$ needs to submit the challenge access structure $\mathbb{A}^*$, the challenge revocation list $\mathcal{R}^*$ and the challenge decryptable time period T_c^* to the Challenger $\mathcal{C}$.
- **Setup**. $\mathcal{C}$ runs the Setup algorithm and gives the public parameters PK to the adversary.
- **Phase 1**. Adversary $\mathcal{A}$ makes repeated private key queries corresponding to the identity ID, the attribute set S and the range of validity time periods T such that for any single returned secret key $SK_{(ID,S,\mathsf{T})}$, at least one of the following conditions must be fulfilled:

[3] In practice, this can also be the serial number of a user key in order to achieve key-level revocation.

  - S does not satisfy the access structure $\mathbb{A}^*$.
  - $ID \in \mathcal{R}^*$.
  - T_c^* is not completely covered in T.
- **Challenge**. Adversary $\mathcal{A}$ submits two equal length messages m_0 and m_1. Challenger $\mathcal{C}$ flips a random coin $\beta \in \{0,1\}$ and encrypts m_β under the access structure $\mathbb{A}^*$, the revoked set $\mathcal{R}^*$ and the time T_c^*. The ciphertext CT^* is given to adversary $\mathcal{A}$.
- **Phase 2**. It is the same as in Phase 1.
- **Guess**. The adversary outputs a guess β' for β.

The advantage of adversary $\mathcal{A}$ in the above game is defined as $Adv_{\mathcal{A}} = Pr[\beta' = \beta] - 1/2$.

Definition 3.1. *A revocable ciphertext-policy attribute-based encryption scheme is secure if all polynomial time adversaries have at most a negligible advantage in the above game.*

Other mathematical background is provided in the appendix.

4 Our Scheme

4.1 Overview

Our scheme is motivated by the (non-revocable) CP-ABE scheme in [37]. We first add a revocation list in the ciphertext so that users in the revocation list cannot decrypt. We then add time validity to user secret key and a decryptable time period in the ciphertext. Users without having a complete cover of validity for the decryptable time period cannot decrypt. (Readers may refer to Section 3.1 for more details.)

Observe that the valid time always have the "AND" relation with the user attributes. Therefore, we have to attach the *time validity* tightly to the secret key in order to avoid the collusion attack. A naïve approach is to treat each time period as an attribute. In the policy, we just need to add the "AND" relation with all valid time periods (that is, time-attributes) together with the original policy. It works fine, if the number of valid time periods in the system is small. Otherwise, the size of public parameters and user secret key will be very large as they grow linear with the number of attributes (that is, number of time periods in this naïve approach). For example, if the system supports up to 10 years and the smallest unit of time period is *day*, then there will be more than 3000 time-attributes in the system!

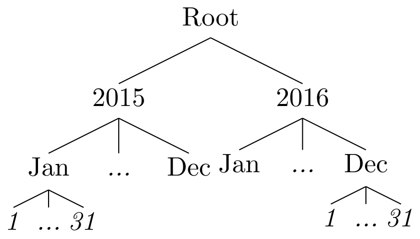
If the user is revoked before the normal expiry time, the user ID will be put into the revocation list until his expiry time has passed. Anyone whose ID is in the revocation list will not be able to decrypt any ciphertext, regardless of his attributes. The idea of the revocation approach is motived from [4,5].

Also note that we have the same restriction as the scheme in [37] using the decisional BDHE assumption (the first scheme). That is, an attribute can only be used in at most one row in the ciphertext access matrix M (the function $\rho(\cdot)$ is injective). This can be thought of as an attribute appearing in at most one node in a formula. However, this limitation can be easily mitigated by assigning a different string for each time an attribute is associated with a row in an access structure.

4.2 Technical Construction

We borrow the idea from the Boneh-Boyen-Goh Hierarchical Identity-based Encryption (HIBE) scheme [12] to apply in our time validity control. We take advantage of the fact that the validity period of a user's key or ciphertext is usually represented as some time interval (e.g. from January to December), instead of some discrete time segments (e.g. January and March and July and December). Therefore, we use a tree-based approach to further improve the efficiency for continuous time interval. The advantage is two-fold. Firstly, the size of the user secret key is reduced. Secondly, if the encryptor wants to encrypt the message for some time interval, then only the user with keys valid in the complete time interval can decrypt. At the same time, the size of the ciphertext is still independent of the length of the time interval.

We use the set-cover approach to select the minimum number of nodes that can represent all the valid time periods. A node (except the root node) in the tree represents a time period. By using HIBE, the user obtains the keys corresponding to these nodes only. Consider the following example:



The first level represents the *year*. The second level represents the *month*. The third level represents the *day*.

Suppose an employee joins the company on 29 November 2015 and his contract ends on 31 December 2016. He should obtain keys for the nodes of "2015-Nov-29", "2015-Nov-30", "2015-Dec" and "2016". For the encryptor, he can choose to encrypt a message for a specific day, for a whole month or whole year. The employee who is authorized for the whole month can decrypt the ciphertext for the whole month. On the other hand, if the ciphertext is specified for a single day only, the employee can derive the decryption key from the corresponding month or year key using the HIBE approach.

In order to simplify the description, suppose the time tree has depth $\mathtt{T}$ and each node have z children. A time period (e.g. a day, a month, a year) can be represented by a z-ary element $(\tau_1, \tau_2, \ldots, \tau_k)$ for some $k < \mathtt{T}$. Our construction is as follows.

1. $\mathsf{Setup}(U, R, \mathtt{T})$: U is the number of attributes in the system. Time is represented as a z-ary string $\{1, z\}^{\mathtt{T}-1}$. The maximum number of revoked users is $R - 1$. Choose a bilinear group $\mathbb{G}$ of prime order p with a random generator g and U random elements $h_1, \ldots, h_U, \in \mathbb{G}$. Randomly choose $\alpha, \alpha_0 \in \mathbb{Z}_p$ and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_R)^\top \in_R \mathbb{Z}_p^R$, $V_0, V_1, \ldots, V_{\mathtt{T}} \in_R \mathbb{G}$, set
$$\mathbf{F} = g^{\boldsymbol{\alpha}} = (g^{\alpha_1}, \cdots, g^{\alpha_R})^\top = (f_1, \ldots, f_R)^\top.$$
Output $PK = \Big\{g, g^{\alpha_0}, e(g, g)^\alpha, h_1, \ldots, h_U, V_0, V_1, \ldots, V_{\mathtt{T}}, \mathbf{F}\Big\}$ and $MK = \alpha$.

2. $\mathsf{KeyGen}(MK, ID, S, \mathsf{T})$: S is the set of attributes of a user with identity ID. T is the range of validity time periods for the user ID. Denote $\mathbb{T}$ as the set-cover representing T which consists of some time elements $\tau = (\tau_1, \tau_2, \ldots, \tau_{k_\tau}) \in \{1, z\}^{k_\tau}$ where $k_\tau < \mathtt{T}$ for any $\tau \in \mathbb{T}$. [4]. Randomly choose $u, t, v_\tau \in_R \mathbb{Z}_p$ for all $\tau \in \mathbb{T}$ and compute
$$D_0 = g^t, \qquad D_0' = g^u, \qquad \Big\{D_{0,\tau}^{''} = g^{v_\tau}\Big\}_{\tau\in\mathbb{T}},$$
$$\Big\{D_{1,\tau} = g^\alpha g^{\alpha_0 t} g^{\alpha_1 u}(V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau}\Big\}_{\tau\in\mathbb{T}}, \qquad \{L_{j,\tau} = V_j^{v_\tau}\}_{j=k_\tau+1,\ldots,\mathtt{T},\tau\in\mathbb{T}},$$
$$\{K_x = h_x^t\}_{x\in S}, \qquad \Big\{F_i = (f_1^{-ID^{i-1}} \cdot f_i)^u\Big\}_{i=2,\cdots,R}.$$
Output
$$SK_{(ID,S,\mathsf{T})} = \Big\{D_0, D_0', \{D_{0,\tau}^{''}, D_{1,\tau}, L_{k_\tau+1,\tau}, \ldots, L_{\mathtt{T},\tau}\}_{\tau\in\mathbb{T}}, \{K_x\}_{x\in S}, \{F_i\}_{i=2,\cdots,R}\Big\}$$
as the user secret key for the user with identity ID, attribute set S and time validity period T.

3. $\mathsf{Encrypt}(PK, m, \mathsf{T}_c, \mathcal{R}, \mathbb{A} = (M, \rho))$: $\mathcal{R} = (ID_1, \ldots, ID_r)$ is the revocation list with r revoked users and $r < R$. $m \in \mathbb{G}_T$ is the plaintext message and T_c is the decryptable time period of this ciphertext. Let $\tau_c = (\tau_1, \ldots, \tau_k) \in \{1, z\}^k$ be the z-ary representation of T_c, where $k < \mathtt{T}$. [5] Take as input an LSSS access structure $\mathbb{A} = (M, \rho)$. The function ρ associates rows of M to attributes. Let M be an $\ell \times n$ matrix. The algorithm first chooses a random vector $\mathbf{v} = (s, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$. These values will be used to share the encryption exponent s. For $i = 1$ to ℓ, it calculates $\lambda_i = \langle \mathbf{v}, M_i\rangle$, where M_i is the vector corresponding to the ith row of M. Also let
$$\mathcal{F}_{\mathcal{R}}(Z) = (Z - ID_1)\cdots(Z - ID_r) = y_1 + y_2 Z + \cdots + y_r Z^{r-1} + y_{r+1} Z^r. \quad (1)$$

[4] For example, if the user is valid from 2015-Nov-29 to 2016-Dec-31, $\mathbb{T} = \{(2015, Nov, 29), (2015, Nov, 30), (2015, Dec), (2016)\}$.

[5] Note that if $k < \mathtt{T}$, it means that only the users valid throughout a period of time $(\tau_1, \ldots, \tau_k, 1, 1, \ldots, 1)$ and $(\tau_1, \ldots, \tau_k, z, z, \ldots, z)$ can decrypt. For example, if the decryptable time period of this ciphertext is "2015-Dec" (and thus $\tau = (2015, Dec)$), only user with secret key valid for the whole December can decrypt this ciphertext.

If $r+1 < R$, the coefficients $y_{r+2}, \cdots, y_R$ are set to 0. Compute

$$C_0 = m \cdot e(g,g)^{\alpha s}, \quad C_0^{'} = g^s, \quad C_0^{''} = (f_1^{y_1} \cdots f_R^{y_R})^s, \quad C_0^{'''} = (V_0 \prod_{j=1}^{k} V_j^{\tau_j})^s,$$
$$C_1 = g^{\alpha_0 \lambda_1} h_{\rho(1)}^{-s}, \quad \ldots, \quad C_\ell = g^{\alpha_0 \lambda_\ell} h_{\rho(\ell)}^{-s}.$$

Output a ciphertext $CT = \{C_0,\ C_0^{'},\ C_0^{''},\ C_0^{'''},\ C_1,\ \ldots,\ C_\ell\}$ along with a description of $\mathsf{T}_c, (M,\rho)$ and the revoked set $\mathcal{R}$.

4. $\mathsf{Decrypt}(CT, \mathcal{R}, SK_{(ID,S,\mathsf{T})})$: First define $\mathbf{X} = (1, ID, \cdots, ID^{R-1})$ from the identity ID and $\mathbf{Y} = (y_1, \cdots, y_R)$ from the revoked set $\mathcal{R}$ (where y_i, $i = 1, \ldots, R$ are defined as in equation (1)). Note that

$$\langle \mathbf{X}, \mathbf{Y} \rangle = y_1 + y_2 ID + \ldots + y_r ID^{r-1} + y_{r+1} ID^r = \mathcal{F}_{\mathcal{R}}(ID),$$

and if $r+1 < R$, the coefficients $y_{r+2}, \cdots, y_R$ are 0. If any one of the following conditions occurs, output $\perp$:
- S does not satisfy the access structure (M, ρ).
- $ID \in \mathcal{R}$. That is, $\langle \mathbf{X}, \mathbf{Y} \rangle = \mathcal{F}_{\mathcal{R}}(ID) = 0$.
- T_c is not completely covered in T. That is, τ_c and all its prefixes are not in $\mathbb{T}$, where τ_c is the z-ary representation for T_c and $\mathbb{T}$ is the set-cover for T.

Otherwise, now we have $ID \notin \mathcal{R}$ (that is, $\langle \mathbf{X}, \mathbf{Y} \rangle \neq 0$). First compute

$$F = \prod_{i=2}^{R} F_i^{y_i} = \Big(f_1^{-\langle \mathbf{X}, \mathbf{Y} \rangle} \cdot \prod_{i=1}^{R} f_i^{y_i}\Big)^u \text{ and } \varsigma_1 = \Big(\frac{e(F, C_0')}{e(D_0', C_0'')}\Big)^{\frac{-1}{\langle \mathbf{X}, \mathbf{Y} \rangle}} = e(g,g)^{\alpha_1 s u}.$$

Further let $I \subset \{1, 2, \ldots, \ell\}$ be defined as $I = \{i : \rho(i) \in S\}$. Then, let $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret s according to M, then $\sum_{i \in I} \omega_i \lambda_i = s$. Compute

$$\varsigma_2 = \prod_{i \in I} (e(C_i, D_0) \cdot e(C_0^{'}, K_{\rho(i)}))^{\omega_i} = e(g,g)^{\alpha_0 s t}.$$

If $\tau_c = (\tau_1, \ldots, \tau_k) \in \mathbb{T}$, D_{1,τ_c} should be one of the components in the secret key. Otherwise, let its prefix $\tau_c' = (\tau_1, \ldots, \tau_{k'})$, where $k' < k$, such that $\tau_c' \in \mathbb{T}$. Then derive the key from the secret key with respect to τ_c' as follows: $D_{1,\tau_c} = D_{1,\tau_c'} \prod_{j=k'+1}^{k} L_{j,\tau_c'}^{\tau_j}$, and set $\tau_c = \tau_c'$. Finally, compute $m = \frac{C_0 \cdot \varsigma_1 \cdot \varsigma_2 \cdot e(D_{0,\tau_c}^{''}, C_0^{'''})}{e(D_{1,\tau_c}, C_0^{'})}$.

We provide the correctness and security analysis in the appendix.

4.3 Future Enhancements

There are some future enhancements that we can further improve upon the current construction:

- To lift the restriction for repeated attributes while keeping simple or standard assumption. We note that [38] provided a construction for non-revocable CP-ABE that has removed this restriction. Yet they use a non-standard assumption (decisional parallel BDHE assumption). Theoretically speaking, we can build up a system based on this scheme. However, the resulting system will also rely on the decisional parallel BDHE assumption.
- To add ciphertext update for revoked users (so that they cannot decrypt those ciphertext generated in the past). We can use the technique of proxy re-encryption to achieve this. But we believe this is not the most essential feature of a revocable ABE since revoked users can anyway decrypt the past ciphertext before they are revoked. If they have done so, it has no use to re-encrypt the ciphertext unless the system is fully cloud-based (e.g. cloud-assisted approach).

5 Performance Analysis

We first compare the efficiency of our scheme with other revocable ABE schemes. We present our comparison in Table 2. We use the following symbols in our comparison table:

Table 2: Efficiency Comparison

Scheme	PK size	SK size	Ciphertext size	Decryption Time (# of pairing)	KP/CP -ABE	Selective / Adaptive
[2]	$(\log R + U)\mathbb{G} + \mathbb{G}_T$	$((\ell+1)\log N)\mathbb{G}$	$(1 + S + \log r)\mathbb{G} + \mathbb{G}_T$	2ℓ	KP-ABE	Selective
[5]	$(R + U + 1)\mathbb{G} + \mathbb{G}_T$	$((R+1)\cdot \ell)\mathbb{G}$	$3\mathbb{G} + \mathbb{G}_T$	$2I$	KP-ABE	Selective
[34]	$(3 + 2U + R)\mathbb{G}$	$4\ell\mathbb{G}$	$(2 + 2S)\mathbb{G} + \mathbb{G}_T$	$2I$	KP-ABE	Selective
[17]	$(\log N + L + 3)\mathbb{G}$	$(1 + L + Q)\mathbb{G}$	$(2 + S)\mathbb{G} + \mathbb{G}_T$	$2L + 3Q + 3$ (multilinear)	KP-ABE	Selective
[18]	$111\mathbb{G} + \mathbb{G}_T$	$(5 + 16\ell + 16(\log^2 N + \log N)\mathbb{G}$	$(16S + 64R - 27)\mathbb{G} + \mathbb{G}_T$	16S + 37	KP-ABE	Adaptive
[29]	$32N\mathbb{G}$	$8N\mathbb{G}$	$4N\mathbb{G} + \mathbb{G}_T$	$(2 + r)N$	CP-ABE	Adaptive
[26]	$(5 + 8\sqrt{N})\mathbb{G} + \sqrt{N}\mathbb{G}_T$	$(2 + 2S + \sqrt{N})\mathbb{G}$	$(16\sqrt{N} + 3\ell)\mathbb{G} + \mathbb{G}_T$	$9 + 3I$	CP-ABE	Selective
Our	$(U + R + \mathtt{T}$ $+3)\mathbb{G} + \mathbb{G}_T$	$(S + Z + R + 1)\mathbb{G}$ best case: $Z = 2$ worst case: $Z = \frac{\mathtt{T}(\mathtt{T}+3)}{2}$	$(3 + \ell)\mathbb{G} + \mathbb{G}_T$	$4 + 2I$	CP-ABE	Selective

- R: max number of revoked users; U: max number of attributes in the system
- $\mathtt{T}$: number of time period level (depth of time tree)
- N: max number of users in the system; S: number of attributes of the user
- r: number of revoked users in the revocation list
- ℓ: number of rows of the access structure matrix
- I: number of attributes used in the decryption
- L: max number of length of input wires (exclusive for [17] only)
- Q: max number of gates (exclusive for [17] only)
- $\mathbb{G}_T$: number $\mathbb{G}_T$ elements; $\mathbb{G}$: number $\mathbb{G}$ elements

Note that we exclude the following for the comparison as they are of different features or security level with our scheme:

- Using indirect approach and cloud-assisted approach (such as those listed in Section 2), as they cannot support instant revocation or require a cloud server to assist decryption.
- Using broadcast encryption technique (such as [3,34]) as they require the encryptor to know the identity of all possible decryptors, which is not exactly an ABE but more or less similar to a broadcast encryption in nature.
- Weak security model [9]. Their model is very week. They only allow the adversary to query secret key that does not satisfy the challenge access structure AND not in the revocation list. Under this model, the collusion attack we mentioned in Section 1.2 is NOT considered as a valid attack. That means their model cannot capture such a low-level attack.
- Attribute-level revocation only [50]. They only support attribute-level revocation, instead of the more generalized user-level or key-level revocation.

From the comparison, we can see that our scheme is the most efficient CP-ABE using direct approach for revocation. In practice, $\mathtt{T}$ can be very small. For example, if we only consider *year*, *month* and *day*, $\mathtt{T} = 3$. Our space and computation complexity do not depend on N, the total number of users in the system, which is supposed to be a very large number.

6 Conclusion

In this paper, we proposed a revocable CP-ABE scheme based on the direct revoke approach. That is, the revocation list is embedded into the ciphertext so that instant revocation can be achieved. In order to shorten the revocation list, we further propose a time validity technique so that expired users cannot decrypt ciphertext associated with a decryptable time period not completely covered under their validity period. We deploy a tree-based structure and HIBE technique to construct the time validity part. The efficiency analysis also shows that our scheme is practical enough to be deployed.

Acknowledgement

This work was supported by Australian Research Council (ARC) Grant DP180102199, the National Natural Science Foundation of China (61702342), the Science and Technology Innovation Projects of Shenzhen (JCYJ20160307150216309, JCYJ20170302151321095) and Tencent "Rhinoceros Birds" -Scientific Research Foundation for Young Teachers of Shenzhen University.

References

1. N. Attrapadung, J. Herranz, F. Laguillaumie, B. Libert, E. de Panafieu, and C. Ràfols. Attribute-based encryption schemes with constant-size ciphertexts. *Theor. Comput. Sci.*, 422:15–38, 2012.

2. N. Attrapadung and H. Imai. Attribute-based encryption supporting direct/indirect revocation modes. In *IMA*, volume 5921 of *LNCS*, pages 278–300. Springer, 2009.
3. N. Attrapadung and H. Imai. Conjunctive broadcast and attribute-based encryption. In *Pairing*, volume 5671 of *LNCS*, pages 248–265. Springer, 2009.
4. N. Attrapadung and B. Libert. Functional encryption for inner product: Achieving constant-size ciphertexts with adaptive security or support for negation. In *PKC*, volume 6056 of *LNCS*, pages 384–402. Springer, 2010.
5. N. Attrapadung, B. Libert, and E. de Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In *PKC*, volume 6571 of *LNCS*, pages 90–108. Springer, 2011.
6. M. H. Au, Q. Huang, J. K. Liu, W. Susilo, D. S. Wong, and G. Yang. Traceable and retrievable identity-based encryption. In *ACNS*, volume 5037 of *LNCS*, pages 94–110. Springer, 2008.
7. M. H. Au, J. K. Liu, T. H. Yuen, and D. S. Wong. Practical hierarchical identity based encryption and signature schemes without random oracles. *IACR Cryptology ePrint Archive*, 2006:368, 2006.
8. M. H. Au, T. H. Yuen, J. K. Liu, W. Susilo, X. Huang, Y. Xiang, and Z. L. Jiang. A general framework for secure sharing of personal health records in cloud system. *J. Comput. Syst. Sci.*, 90:46–62, 2017.
9. A. Balu and K. Kuppusamy. Ciphertext-policy attribute-based encryption with user revocation support. In *QShine*, volume 115 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 696–705. Springer, 2013.
10. A. Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.
11. A. Boldyreva, V. Goyal, and V. Kumar. Identity-based encryption with efficient revocation. In *CCS*, pages 417–426. ACM, 2008.
12. D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In *EUROCRYPT*, volume 3494 of *LNCS*, pages 440–456. Springer, 2005.
13. D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, volume 2139 of *LNCS*, pages 213–229. Springer, 2001.
14. S. S. M. Chow, J. K. Liu, and J. Zhou. Identity-based online/offline key encapsulation and encryption. In *ASIACCS*, pages 52–60. ACM, 2011.
15. C. Chu, J. K. Liu, J. Zhou, F. Bao, and R. H. Deng. Practical id-based encryption for wireless sensor network. In *ASIACCS*, pages 337–340. ACM, 2010.
16. H. Cui, R. H. Deng, Y. Li, and B. Qin. Server-aided revocable attribute-based encryption. In *ESORICS Part II*, volume 9879 of *LNCS*, pages 570–587. Springer, 2016.
17. P. Datta, R. Dutta, and S. Mukhopadhyay. General circuit realizing compact revocable attribute-based encryption from multilinear maps. In *ISC*, volume 9290 of *LNCS*, pages 336–354. Springer, 2015.
18. P. Datta, R. Dutta, and S. Mukhopadhyay. Adaptively secure unrestricted attribute-based encryption with subset difference revocation in bilinear groups of prime order. In *AFRICACRYPT*, volume 9646 of *LNCS*, pages 325–345. Springer, 2016.
19. K. He, J. Weng, J. K. Liu, W. Zhou, and J. Liu. Efficient fine-grained access control for secure personal health records in cloud computing. In *NSS*, volume 9955 of *LNCS*, pages 65–79. Springer, 2016.

20. J. Hong, K. Xue, and W. Li. Comments on "DAC-MACS: Effective Data Access Control for Multiauthority Cloud Storage Systems" / Security Analysis of Attribute Revocation in Multiauthority Data Access Control for Cloud Storage Systems. *IEEE Trans. Information Forensics and Security*, 10(6):1315–1317, 2015.
21. J. Hur and D. K. Noh. Attribute-based access control with efficient revocation in data outsourcing systems. *IEEE Trans. Parallel Distrib. Syst.*, 22(7):1214–1221, 2011.
22. K. Liang, M. H. Au, J. K. Liu, W. Susilo, D. S. Wong, G. Yang, T. V. X. Phuong, and Q. Xie. A dfa-based functional proxy re-encryption scheme for secure public cloud data sharing. *IEEE Trans. Information Forensics and Security*, 9(10):1667–1680, 2014.
23. K. Liang, M. H. Au, J. K. Liu, W. Susilo, D. S. Wong, G. Yang, Y. Yu, and A. Yang. A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing. *Future Generation Comp. Syst.*, 52:95–108, 2015.
24. J. Liu, X. Huang, and J. K. Liu. Secure sharing of personal health records in cloud computing: Ciphertext-policy attribute-based signcryption. *Future Generation Comp. Syst.*, 52:67–76, 2015.
25. J. K. Liu, M. H. Au, X. Huang, R. Lu, and J. Li. Fine-grained two-factor access control for web-based cloud computing services. *IEEE Trans. Information Forensics and Security*, 11(3):484–497, 2016.
26. Z. Liu and D. S. Wong. Practical ciphertext-policy attribute-based encryption: Traitor tracing, revocation, and large universe. In *ACNS*, volume 9092 of *LNCS*, pages 127–146. Springer, 2015.
27. T. Naruse, M. Mohri, , and Y. Shiraishi. Attribute-based encryption with attribute revocation and grant function using proxy re-encryption and attribute key for updating. In *Future Information Technology*, volume 276 of *Lecture Notes in Electrical Engineering*, pages 119–125. Springer, 2014.
28. T. Naruse, M. Mohri, and Y. Shiraishi. Provably secure attribute-based encryption with attribute revocation and grant function using proxy re-encryption and attribute key for updating. *Human-centric Computing and Information Sciences*, 5(1):1–13, 2015.
29. J. M. G. Nieto, M. Manulis, and D. Sun. Fully private revocable predicate encryption. In *ACISP*, volume 7372 of *LNCS*, pages 350–363. Springer, 2012.
30. H. Qian, J. Li, Y. Zhang, and J. Han. Privacy-preserving personal health record using multi-authority attribute-based encryption with revocation. *Int. J. Inf. Sec.*, 14(6):487–497, 2015.
31. S. Ruj, A. Nayak, and I. Stojmenovic. DACC: distributed access control in clouds. In *TrustCom 2011*, pages 91–98. IEEE Computer Society, 2011.
32. A. Sahai, H. Seyalioglu, and B. Waters. Dynamic credentials and ciphertext delegation for attribute-based encryption. In *CRYPTO*, volume 7417 of *LNCS*, pages 199–217. Springer, 2012.
33. J. Shi, C. Huang, J. Wang, K. He, and J. Wang. An access control scheme with direct cloud-aided attribute revocation using version key. In *ICA3PP Part I*, volume 8630 of *LNCS*, pages 429–442. Springer, 2014.
34. P. Wang, D. Feng, and L. Zhang. Towards attribute revocation in key-policy attribute based encryption. In *CANS*, volume 7092 of *LNCS*, pages 272–291. Springer, 2011.
35. S. Wang, K. Liang, J. K. Liu, J. Chen, J. Yu, and W. Xie. Attribute-based data sharing scheme revisited in cloud computing. *IEEE Trans. Information Forensics and Security*, 11(8):1661–1673, 2016.

36. S. Wang, J. Zhou, J. K. Liu, J. Yu, J. Chen, and W. Xie. An efficient file hierarchy attribute-based encryption scheme in cloud computing. *IEEE Trans. Information Forensics and Security*, 11(6):1265–1277, 2016.
37. B. Waters. Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization. Cryptology ePrint Archive, Report 2008/290, 2008. `http://eprint.iacr.org/`.
38. B. Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *PKC*, volume 6571 of *LNCS*, pages 53–70. Springer, 2011.
39. F. Xhafa, J. Wang, X. Chen, J. K. Liu, J. Li, and P. Krause. An efficient PHR service system supporting fuzzy keyword search and fine-grained access control. *Soft Comput.*, 18(9):1795–1802, 2014.
40. X. Xie, H. Ma, J. Li, and X. Chen. An efficient ciphertext-policy attribute-based access control towards revocation in cloud computing. *J. UCS*, 19(16):2349–2367, 2013.
41. X. Xie, H. Ma, J. Li, and X. Chen. New ciphertext-policy attribute-based access control with efficient revocation. In *ICT-EurAsia*, volume 7804 of *LNCS*, pages 373–382. Springer, 2013.
42. K. Yang, X. Jia, K. Ren, and B. Zhang. DAC-MACS: effective data access control for multi-authority cloud storage systems. In *INFOCOM*, pages 2895–2903. IEEE, 2013.
43. K. Yang, X. Jia, K. Ren, B. Zhang, and R. Xie. DAC-MACS: effective data access control for multiauthority cloud storage systems. *IEEE Trans. Information Forensics and Security*, 8(11):1790–1801, 2013.
44. Y. Yang, X. Ding, H. Lu, Z. Wan, and J. Zhou. Achieving revocable fine-grained cryptographic access control over cloud data. In *ISC*, volume 7807 of *LNCS*, pages 293–308. Springer, 2015.
45. Y. Yang, J. K. Liu, K. Liang, K. R. Choo, and J. Zhou. Extended proxy-assisted approach: Achieving revocable fine-grained encryption of cloud data. In *ESORICS Part II*, volume 9327 of *LNCS*, pages 146–166. Springer, 2015.
46. Y. Yang, J. K. Liu, Z. Wei, and X. Huang. Towards revocable fine-grained encryption of cloud data: Reducing trust upon cloud. In *ACISP Part I*, volume 10342 of *LNCS*, pages 127–144. Springer, 2017.
47. J. Ye, W. Zhang, S. Wu, Y. Gao, and J. Qiu. Attribute-based fine-grained access control with user revocation. In *ICT-EurAsia*, volume 8407 of *LNCS*, pages 586–595. Springer, 2014.
48. S. Yu, C. Wang, K. Ren, and W. Lou. Attribute based data sharing with attribute revocation. In *ASIACCS*, pages 261–270. ACM, 2010.
49. T. H. Yuen, Y. Zhang, S. Yiu, and J. K. Liu. Identity-based encryption with post-challenge auxiliary inputs for secure cloud applications and sensor networks. In *ESORICS Part I*, volume 8712 of *LNCS*, pages 130–147. Springer, 2014.
50. M. Zhang. New model and construction of ABE: achieving key resilient-leakage and attribute direct-revocation. In *ACISP*, volume 8544 of *LNCS*, pages 192–208. Springer, 2014.
51. C. Zuo, J. Shao, J. K. Liu, G. Wei, and Y. Ling. Fine-grained two-factor protection mechanism for data sharing in cloud storage. *IEEE Trans. Information Forensics and Security*, 13(1):186–196, 2018.

A Mathematical Background

A.1 Vector

We use $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_n)^\top \in \mathbb{Z}_p^n$ and $g^{\boldsymbol{\alpha}} = (g^{a_1}, \cdots, g^{a_n})^\top \in \mathbb{G}^n$ to denote column vectors. For $\mathbf{a}, \mathbf{z} \in \mathbb{Z}_p^n$, $\langle \mathbf{a}, \mathbf{z} \rangle = \mathbf{a}^\top \mathbf{z} = \sum_{i=1}^n a_i z_i$ and $(g^{\mathbf{a}})^{\mathbf{z}} = g^{\langle \mathbf{a}, \mathbf{z} \rangle}$.

A.2 Bilinear Groups

$\mathcal{G}$ is an algorithm, which takes as input a security parameter κ and outputs a tuple $(p, \mathbb{G}, \mathbb{G}_T, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ are multiplicative cyclic groups with prime order p, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a map, which has the following properties:

- **Bilinearity:** $e(g^a, h^b) = e(g, h)^{ab}$ for $\forall g, h \in \mathbb{G}$ and $\forall a, b \in \mathbb{Z}_p$.
- **Non-degeneracy:** There exists $g, h \in \mathbb{G}$ such that $e(g, h) \neq 1_{\mathbb{G}}$.
- **Computability:** There exists an efficient algorithm to compute $e(g, h)$ for $\forall g, h \in \mathbb{G}$.

A.3 Access Structure

Let $\{P_1, P_2, \cdots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \cdots, P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure is a collection $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \cdots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1, P_2, \cdots, P_n\}} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

A.4 Linear Secret Sharing Schemes

We adapt our definition from those given in [10]. A secret-sharing scheme Π over a set of parties $\mathcal{P}$ is called linear (over $\mathbb{Z}_p$) if

- The shares of the parties form a vector over $\mathbb{Z}_p$.
- There exists a matrix M with ℓ rows and n columns called the share-generating matrix for Π. There exists a function ρ which maps each row of the matrix to an associated party. That is for $i = 1, \cdots, \ell$ the value $\rho(i)$ is the party associated with row i. When we consider the column vector $\mathbf{v} = (s, r_2, \ldots, r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $r_2, \cdots, r_n \in \mathbb{Z}_p$ are randomly chosen, then $M\mathbf{v}$ is the vector of ℓ shares of the secret s according to Π. The share $(M\mathbf{v})_i$ belongs to party $\rho(i)$.
  Every linear secret sharing scheme according to the above definition also enjoys the linear reconstruction property which defined as follows: Suppose that Π is an LSSS for the access structure $\mathbb{A}$. Let $S \in A$ be any authorized set, and let $I \subset \{1, 2, \cdots, \ell\}$ be defined as $I = \{i : \rho(i) \in S\}$. Then, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ such that, if $\{\lambda_i\}$ are valid shares of any secret s according to Π, then $\sum_{i \in I} \omega_i \lambda_i = s$. These constants ω_i can be found in time polynomial in the size of the share-generating matrix M. Like any secret sharing scheme, it has the property that for any unauthorized set $S \notin A$, the secret s should be information theoretically hidden from the parties in S.

A.5 Decisional q-Bilinear Diffie-Hellman Exponent (BDHE) Problem

The Decisional q-BDHE problem is defined as given

$$(g, g^s, g^a, \cdots, g^{(a^q)}, g^{(a^{(q+2)})}, \cdots, g^{(a^{2q})}, T)$$

where $s, a \in \mathbb{Z}_p$, $g \in \mathbb{G}$ and $T \in \mathbb{G}_T$, to decide if $T = e(e, g)^{sa^{q+1}}$ or if T is a random element of $\mathbb{G}_T$.

B Security Analysis

The correctness of the scheme in Section 4.2 can be seen from the following equations.

$$\begin{aligned} F &= \prod_{i=2}^{R} F_i^{y_i} \\ &= \prod_{i=2}^{R} (f_1^{-ID^{i-1}} f_i)^{y_i u} \\ &= (f_1^{-(IDy_2 + ID^2 y_3 + \cdots + ID^{R-1} y_R)} g^{\sum_{i=2}^{R} \alpha_i y_i})^u \\ &= \Big(f_1^{-\langle \boldsymbol{X}, \boldsymbol{Y} \rangle + y_1} \prod_{i=2}^{R} f_i^{y_i} \Big)^u \\ &= \Big(f_1^{-\langle \boldsymbol{X}, \boldsymbol{Y} \rangle} \cdot \prod_{i=1}^{R} f_i^{y_i} \Big)^u. \end{aligned} \tag{2}$$

$$\begin{aligned} \varsigma_1 &= \Big(\frac{e(F, C_0')}{e(D_0', C_0'')} \Big)^{\frac{-1}{\langle \boldsymbol{X}, \boldsymbol{Y} \rangle}} \\ &= \left(\frac{e\big((f_1^{-\langle \boldsymbol{X}, \boldsymbol{Y} \rangle} \cdot \prod_{i=1}^{R} f_i^{y_i})^u, g^s\big)}{e\big(g^u, (f_1^{y_1} \cdots f_R^{y_R})^s\big)} \right)^{\frac{-1}{\langle \boldsymbol{X}, \boldsymbol{Y} \rangle}} \\ &= e(g, g)^{\alpha_1 s u}. \end{aligned} \tag{3}$$

$$\begin{aligned} \varsigma_2 &= \prod_{i \in I} \Big(e(C_i, D_0) \cdot e(C_0', K_{\rho(i)}) \Big)^{\omega_i} \\ &= \prod_{i \in I} \Big(e(g^{\alpha_0 \lambda_i} h_{\rho(i)}^{-s}, g^t) \cdot e(g^s, h_{\rho(i)}^t) \Big)^{\omega_i} \\ &= \prod_{i \in I} e(g, g)^{t \alpha_0 \lambda_i \omega_i} \\ &= e(g, g)^{st\alpha_0}. \end{aligned} \tag{4}$$

$$
\begin{aligned}
m &= \frac{C_0 \cdot \varsigma_1 \cdot \varsigma_2 \cdot e(D_0^{''}, C_0^{'''})}{e(D_{1,\tau_c}, C_0^{'})} \\
&= \frac{m \cdot e(g,g)^{\alpha s} \cdot e(g,g)^{\alpha_1 s u} \cdot e(g,g)^{s t \alpha_0} \cdot e(g^{v_{\tau_c}}, (V_0 \prod_{j=1}^{k} V_j^{\tau_j})^s)}{e(g^{\alpha} g^{\alpha_0 t} g^{\alpha_1 u} (V_0 \prod_{j=1}^{k} V_j^{\tau_j})^{v_{\tau_c}}, g^s)}. \qquad (5)
\end{aligned}
$$

We prove the following theorem.

Theorem B.1. *Suppose the decisional q-BDHE assumption holds. Then no probabilistic polynomial time adversary can selectively break our system in with a challenge matrix of size $\ell^* \times n^*$, where $\ell^*, n^* \leq q$, a challenge revocation list $\mathcal{R}^*$ where $|\mathcal{R}^*| \leq q-2$ and a challenge time $\mathtt{T}_c^*$ with z-ary representation $\tau^* = (\tau_1^*, \ldots, \tau_{k^*}^*)$ for some $k^* < \mathtt{T}$ such that $\mathtt{T} \leq q$.*

Proof. Suppose there exists an adversary $\mathcal{A}$ with non-negligible advantage against our construction. Then challenger $\mathcal{C}$ can resolve the decisional q-BDHE problem with non-negligible probability.

- **Init**. Challenger $\mathcal{C}$ receives

$$\boldsymbol{y} = (g, g^s, g^a, \cdots, g^{a^q}, g^{a^{q+2}}, \cdots, g^{a^{2q}}) \in \mathbb{G}^{2q+1}, T \in \mathbb{G}_T,$$

  and decides if $T = e(g,g)^{sa^{q+1}}$ using the adversary $\mathcal{A}$. $\mathcal{A}$ first gives the challenge access structure (M^*, ρ^*) where M^* has $n^* \leq q$ columns. $\mathcal{A}$ also gives $\mathcal{C}$ the challenge time $\mathtt{T}_c^*$ with z-ary representation $\tau^* = (\tau_1^*, \ldots, \tau_{k^*}^*)$ for some $k^* \leq \mathtt{T}$ and the challenge revocation list $\mathcal{R}^*$, where $|\mathcal{R}^*| \leq q-2$. That is, the maximum number of revoked users allowed in the system $R-1$ is set to $q-2$. In other words, we have $R = q-1$.
- **Setup**. Challenger $\mathcal{C}$ chooses random $\alpha', \delta_0, \xi_0, \xi_1, \ldots, \xi_{\mathtt{T}} \in \mathbb{Z}_p$ and implicitly sets $\alpha = \alpha' + \delta_0 a^{q+1}$ by letting

$$e(g,g)^{\alpha} = e(g^a, g^{a^q})^{\delta_0} e(g,g)^{\alpha'}.$$

  Let $g^{\alpha_0} = g^a$. For each x for $1 \leq x \leq U$ begin by choosing a random value z_x. Let I denote the set of indices i, such that $\rho^*(i) = x$. $\mathcal{C}$ programs h_x as:

$$h_x = g^{z_x} g^{a M_{i,1}^*} \cdot g^{a^2 M_{i,2}^*} \cdots g^{a^{n^*} M_{i,n^*}^*}. \qquad (6)$$

  Note that if $I = \emptyset$ then we have $h_x = g^{z_x}$. Also note that the parameters are distributed randomly due to the g^{z_x} value.
  Let $|\mathcal{R}^*| = r$. Note that $r \leq q-2$. Let $\mathbf{X}_1, \cdots, \mathbf{X}_r$ be the corresponding vectors for the revoked set $\mathcal{R} = \{ID_1, \cdots, ID_r\}$. That is, $\mathbf{X}_k = (1, ID_k, \cdots, ID_k^{q-2})$ for $k \in [1, r]$.
  - For each $k \in [1, r]$, let

$$M_{\boldsymbol{X}_k} = \begin{pmatrix} -ID_k & -ID_k^2 & \cdots & -ID_k^{q-2} \\ & & I_{q-2} & \end{pmatrix}$$

where I_{q-2} is a $(q-2) \times (q-2)$ identity matrix. $\mathcal{C}$ selects $\mathbf{b}_k \in_R \mathbb{Z}_p^{q-1}$ such that

$$\mathbf{b}_k \cdot M_{\mathbf{X}_k} = \mathbf{0}. \tag{7}$$

The simplest candidate consists of the vector $\mathbf{b}_k = (1, ID_k, \cdots, ID_k^{q-2})$.

- For $k \in [r+1, q-1]$, $\mathbf{b}_k = \mathbf{0}$.

Define a $(q-1) \times (q-1)$ matrix

$$\mathrm{B} = (\mathbf{b}_1 | \cdots | \mathbf{b}_r | \mathbf{0} | \cdots | \mathbf{0})$$

where the kth column consists of $\boldsymbol{b}_k$ for $k \in [1, r]$ and the remaining $q-1-r$ columns are $\mathbf{0}$.
Challenger $\mathcal{C}$ defines $V_j = g^{\xi_j a^{q-j+1}}$ for $j \in [1, \mathtt{T}]$ and $V_0 = \prod_{j=1}^{k^*} V_j^{-\tau_j^*} g^{\xi_0}$.
Challenger $\mathcal{C}$ defines $\boldsymbol{\nu} = (\nu_1, \cdots, \nu_{q-1})^\top$ such that $\nu_i = a^{q+1-i}$ and sets $g^{\boldsymbol{\nu}} = (g^{\nu_1}, \ldots, g^{\nu_{q-1}})^\top = (g^{a^q}, \ldots, g^{a^2})^\top$. It then implicitly sets

$$\boldsymbol{\alpha} = \mathrm{B} \cdot \boldsymbol{\nu} + \boldsymbol{\delta} \tag{8}$$

by randomly choosing $\boldsymbol{\delta} \in \mathbb{Z}_p^{q-1}$. Challenger $\mathcal{C}$ defines the last public parameter

$$\mathbf{F} = g^{\mathrm{B} \cdot \boldsymbol{\nu}} \cdot g^{\boldsymbol{\delta}} = g^{\boldsymbol{\alpha}} = (g^{\alpha_1}, \cdots, g^{\alpha_R})^\top = (f_1, \ldots, f_R)^\top.$$

- **Phase 1**. Adversary $\mathcal{A}$ makes repeated private keys queries corresponding to the tuple of identity, attributes and time and identity (ID, S, T), such that at least one of the following conditions must be satisfied:

  1. the attribute set S does not satisfy the access structure $\mathbb{A}^*$;
  2. $ID \in \mathcal{R}^*$;
  3. τ^* and all its prefixes are not in $\mathbb{T}$, the set-cover of T.

  We separate into three cases.

  Case 1: S does not satisfy $\mathbb{A}^*$. Challenger $\mathcal{C}$ first chooses a random $\varphi \in \mathbb{Z}_p$. Then it finds a vector $\mathbf{w} = (w_1, \cdots, w_{n^*}) \in \mathbb{Z}_p^{n^*}$ such that $w_1 = -1$ and for all i where $\rho^*(i) \in S$ we have that $\mathbf{w} \cdot M_i^* = 0$. By our definition of an LSSS, such a vector must exist since S does not satisfy M^*. $\mathcal{C}$ implicitly defines

$$t = \varphi + \delta_0(w_1 a^q + w_2 a^{q-1} + \ldots + w_{n^*} a^{q-n^*+1}).$$

  It performs this by setting

$$D_0 = g^{\varphi} \prod_{i=1}^{n^*} \left(g^{a^{q+1-i}}\right)^{w_i \delta_0} = g^t.$$

$\mathcal{C}$ then randomly chooses $u \in \mathbb{Z}_p$ and sets $D'_0 = g^u$. Then for all $\tau = (\tau_1, \dots, \tau_{k_\tau}) \in \mathbb{T}$, $\mathcal{C}$ randomly chooses $v_\tau \in \mathbb{Z}_p$ and sets $D''_{0,\tau} = g^{v_\tau}$ and:

$$\begin{aligned}
D_{1,\tau} &= g^{\alpha'} g^{a\varphi} \prod_{i=2}^{n^*} \left(g^{a^{q+2-i}}\right)^{w_i \delta_0} g^{\alpha_1 u} (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau} \\
&= g^{\alpha'} g^{\delta_0 a^{q+1}} g^{a\varphi} g^{-\delta_0 a^{q+1}} \prod_{i=2}^{n^*} \left(g^{a^{q+2-i}}\right)^{w_i \delta_0} g^{\alpha_1 u} (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau} \\
&= g^{\alpha} g^{a\varphi} g^{w_1 \delta_0 a^{q+1}} \prod_{i=2}^{n^*} \left(g^{a^{q+2-i}}\right)^{w_i \delta_0} g^{\alpha_1 u} (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau} \quad (\because w_1 = -1) \\
&= g^{\alpha} g^{a\varphi} \prod_{i=1}^{n^*} \left(g^{a^{q+2-i}}\right)^{w_i \delta_0} g^{\alpha_1 u} (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau} \\
&= g^{\alpha} \left(g^{\varphi} \prod_{i=1}^{n^*} \left(g^{a^{q+1-i}}\right)^{w_i \delta_0} \right)^{a} g^{\alpha_1 u} (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau} \\
&= g^{\alpha} g^{at} g^{\alpha_1 u} (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau} \\
&= g^{\alpha} g^{\alpha_0 t} g^{\alpha_1 u} (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau}.
\end{aligned}$$

Now we set K_x $\forall x \in S$. First, we consider $x \in S$ for which there is no i such that $\rho^*(i) = x$. For those we can simply let $K_x = D_0^{z_x}$. Then we create keys for attributes x where x is used in the access structure. We follow the heuristic from [37].

For these keys we have to make sure no terms are of the form $g^{a^{q+1}}$ which cannot be simulated. In calculating h_x^t all terms of this form come from $M^*_{i,j} a^j w_j a^{q+1-j}$ for some j, where $\rho^*(i) = x$. However, we have that $M^*_i \mathbf{w} = 0$. Therefore everything with an exponent a^{q+1} cancels when combined.

$\mathcal{S}$ creates K_x in this case as follows. Suppose $\rho^*(i) = x$, it sets

$$K_x = D_0^{z_x} \cdot \prod_{j=1}^{n^*} \left(g^{a^j \varphi} \prod_{\mu=1, \mu \neq j}^{n^*} \left(g^{a^{q+1+j-\mu}}\right)^{w_\mu \delta_0} \right)^{M^*_{i,j}}.$$

Finally $\mathcal{C}$ computes $F_i = (f_1^{-ID^{i-1}} \cdot f_i)^u$ for $i = 2, \dots, R$ and $\{L_{j,\tau} = V_j^{v_\tau}\}_{j=k_\tau+1,\dots,\mathrm{T},\tau\in\mathbb{T}}$.

Case 2: $ID \in \mathcal{R}^*$. We follow the heuristic in [1]. In this case, let $ID_k \in \mathcal{R}^*$ be the identity of the secret key that $\mathcal{A}$ queries, where $k \in [1, r]$. Challenger $\mathcal{C}$ defines $\widetilde{u}_k = u_k - \delta_0 a^k$ for a random $u_k \in_R \mathbb{Z}_p$. Recall $\boldsymbol{\alpha}$ is defined in equation (8), the first coordinate of $\boldsymbol{\alpha}$ equals

$$\alpha_1 = \delta_1 + \sum_{j=1}^{r} \nu_j = \delta_1 + \sum_{j=1}^{r} a^{q+1-j}. \tag{9}$$

Now challenger $\mathcal{C}$ computes $D_0' = g^{u_k}(g^{a^k})^{-\delta_0} = g^{\tilde{u}_k}$. Next, $\mathcal{C}$ randomly selects $t \in \mathbb{Z}_p$ and computes $D_0 = g^t$. For all $\tau = (\tau_1, \ldots, \tau_{k_\tau}) \in \mathbb{T}$, $\mathcal{C}$ randomly selects $v_\tau \in \mathbb{Z}_p$ and computes $D_{0,\tau}^{''} = g^{v_\tau}$ and

$$\begin{aligned}
D_{1,\tau} &= g^{\alpha'} f_1^{u_k} \left(g^{a^k \delta_1} \prod_{j=1, j\neq k}^{r} g^{a^{q+1-j+k}} \right)^{-\delta_0} (g^a)^t (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau} \\
&= g^{\alpha'} g^{\alpha_1 u_k} g^{a^{q+1}\delta_0} \left(g^{\delta_1} \prod_{j=1}^{r} g^{a^{q+1-j}} \right)^{-\delta_0 a^k} (g^{\alpha_0})^t (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau} \\
&= g^{\alpha} g^{\alpha_1 u_k} (g^{\alpha_1})^{-\delta_0 a^k} g^{\alpha_0 t} (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau} \quad \text{(from equation (9))} \\
&= g^{\alpha} g^{\alpha_1 u_k - \alpha_1 \delta_0 a^k} g^{\alpha_0 t} (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau} \\
&= g^{\alpha} g^{\alpha_1 \tilde{u}_k} g^{\alpha_0 t} (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{v_\tau}.
\end{aligned}$$

We define $\mathbb{F}_k = (F_2, \ldots, F_R)^\top$ which is the secret key component for ID_k. First recall $\boldsymbol{\nu} = (\nu_1, \cdots, \nu_{q-1})^\top$ where $\nu_i = a^{q+1-i}$ and $g^{\boldsymbol{\nu}} = (g^{\nu_1}, \ldots, g^{\nu_{q-1}})^\top = (g^{a^q}, \ldots, g^{a^2})^\top$. $\mathcal{C}$ computes $\mathbb{F}_k$ as follow:

$$\begin{aligned}
\mathbb{F}_k &= g^{u_k M_{\boldsymbol{X}_k}^\top \boldsymbol{\alpha}} \cdot g^{-\delta_0 a^k M_{\boldsymbol{X}_k}^\top B \boldsymbol{\nu}} \cdot g^{-\delta_0 a^k M_{\boldsymbol{X}_k}^\top \boldsymbol{\delta}} \\
&\qquad (\mathcal{C} \text{ is able to compute } g^{a^k M_{\boldsymbol{X}_k}^\top B \boldsymbol{\nu}} \text{ because} \\
&\qquad \text{the } k\text{th column of } M_{\boldsymbol{X}_k}^\top B \textit{ is } \mathbf{0}, \text{ from equation (7))} \\
&= g^{u_k M_{\boldsymbol{X}_k}^\top \boldsymbol{\alpha}} \cdot g^{-\delta_0 a^k M_{\boldsymbol{X}_k}^\top \boldsymbol{\alpha}} \quad \text{(from equation (8))} \\
&= g^{(u_k - \delta_0 a^k) M_{\boldsymbol{X}_k}^\top \boldsymbol{\alpha}} \\
&= g^{\tilde{u}_k M_{\boldsymbol{X}_k}^\top \boldsymbol{\alpha}}.
\end{aligned}$$

That is, for $i = 2, \ldots, R$ (recall that $R = q-1$), we have

$$\begin{aligned}
F_i &= g^{\tilde{u}_k M_{\boldsymbol{X}_k, i-1}^\top \boldsymbol{\alpha}} \\
&\qquad (\text{where } M_{\boldsymbol{X}_k, i-1}^\top \text{ denotes the } (i-1)\text{th row of } M_{\boldsymbol{X}_k}^\top) \\
&= g^{\tilde{u}_k (-ID_k^{i-1} \alpha_1 + \alpha_i)} \\
&= (f_1^{-ID_k^{i-1}} \cdot f_i)^{\tilde{u}_k}.
\end{aligned}$$

Finally $\mathcal{C}$ computes $\{K_x = h_x^t\}_{x \in S}$ and $\{L_{j,\tau} = V_j^{v_\tau}\}_{j=k+1,\ldots,\mathtt{T},\tau\in\mathbb{T}}$.

Case 3: τ^* and all its prefixes are not in $\mathbb{T}$. For all $\tau = (\tau_1, \ldots, \tau_{k_\tau}) \in \mathbb{T}$, first define $\tau_{k_\tau+1} = \ldots = \tau_q = 0$ and $\tau^*_{k^*+1} = \ldots = \tau^*_q = 0$. There exists a smallest index $k' \le k^*$ such that $\tau_{k'} \neq \tau^*_{k'}$.

Challenger $\mathcal{C}$ randomly selects $v_\tau \in \mathbb{Z}_p$ and implicitly defines $\tilde{v_\tau} = \frac{\delta_0 a^{k'}}{\xi_{k'}(\tau^*_{k'} - \tau_{k'})} + v_\tau$. It performs this by setting:

$$D''_{0,\tau} = g^{\frac{\delta_0 a^{k'}}{\xi_{k'}(\tau^*_{k'} - \tau_{k'})} + v_\tau}.$$

$\mathcal{C}$ then randomly chooses $t, u \in \mathbb{Z}_p$ and sets $D_0 = g^t, D'_0 = g^u$. For all τ, compute:

$$\begin{aligned}
D_{1,\tau} &= g^{\alpha' + \alpha_0 t + \alpha_1 u + v_\tau \xi_0} (g^{a^{q-k'+1}})^{v_\tau \xi_{k'}(\tau_{k'} - \tau^*_{k'})} (g^{a^{k'}})^{\frac{\xi_0 \delta_0}{\xi_{k'}(\tau^*_{k'} - \tau_{k'})}} \\
&\quad \cdot \prod_{j=1}^{k_\tau - k' + 1} (g^{a^{q-j+1}})^{\frac{\xi_{j+k'} \tau^*_{j+k'} \delta_0}{\xi_{k'}(\tau^*_{k'} - \tau_{k'})}} \prod_{j=k'+1}^{k_\tau+1} (g^{a^{q-j+1}})^{\xi_j \tau^*_j v_\tau} \\
&= g^{\alpha'} g^{\alpha_0 t} g^{\alpha_1 u} g^{v_\tau(\xi_0 + \xi_{k'} a^{q-k'+1}(\tau_{k'} - \tau^*_{k'}))} g^{\frac{\xi_0 \delta_0 a^{k'}}{\xi_{k'}(\tau^*_{k'} - \tau_{k'})}} \\
&\quad \cdot \prod_{j=k'+1}^{k_\tau+1} g^{\frac{\xi_j a^{q-j+1+k'} \tau^*_j \delta_0}{\xi_{k'}(\tau^*_{k'} - \tau_{k'})}} \prod_{j=k'+1}^{k_\tau+1} g^{\xi_j a^{q-j+1} \tau^*_j v_\tau} \\
&= g^{\alpha' + \delta_0 a^{q+1}} g^{\alpha_0 t} g^{\alpha_1 u} g^{(\xi_0 + \xi_{k'} a^{q-k'+1}(\tau_{k'} - \tau^*_{k'}))(\frac{\delta_0 a^{k'}}{\xi_{k'}(\tau^*_{k'} - \tau_{k'})} + v_\tau)} \\
&\quad \cdot \prod_{j=k'+1}^{k_\tau+1} g^{\xi_j a^{q-j+1} \tau^*_j (\frac{\delta_0 a^{k'}}{\xi_{k'}(\tau^*_{k'} - \tau_{k'})} + v_\tau)} \\
&= g^{\alpha} g^{\alpha_0 t} g^{\alpha_1 u} g^{(\xi_0 + \xi_{k'} a^{q-k'+1}(\tau_{k'} - \tau^*_{k'}))\tilde{v_\tau}} \prod_{j=k'+1}^{k_\tau+1} g^{\xi_j a^{q-j+1} \tau^*_j \tilde{v_\tau}} \\
&= g^{\alpha} g^{\alpha_0 t} g^{\alpha_1 u} (V_0 \prod_{j=1}^{k'} V_j^{\tau_j})^{\tilde{v_\tau}} \prod_{j=k'+1}^{k_\tau+1} V_j^{\tau_j \tilde{v_\tau}} \qquad (\because \tau_j = \tau^*_j \quad \text{if } j < k') \\
&= g^{\alpha} g^{\alpha_0 t} g^{\alpha_1 u} (V_0 \prod_{j=1}^{k_\tau+1} V_j^{\tau_j})^{\tilde{v_\tau}} \\
&= g^{\alpha} g^{\alpha_0 t} g^{\alpha_1 u} (V_0 \prod_{j=1}^{k_\tau} V_j^{\tau_j})^{\tilde{v_\tau}} \qquad (\because \tau_{k_\tau+1} = 0)
\end{aligned}$$

$\mathcal{C}$ also computes $\{K_x = h_x^t\}_{x \in S}, \{F_i = (f_1^{-ID^{i-1}} \cdot f_i)^u\}_{i=2,\cdots,R}$. For $j = k_\tau + 1, \ldots, \mathtt{T}, \tau \in \mathbb{T}$, compute

$$\begin{aligned} L_{j,\tau} &= g^{\frac{\delta_0 \xi_j a^{q+1+k'-j}}{\xi_{k'}(\tau^*_{k'} - \tau_{k'})} + v_\tau \xi_j a^{q-j+1}} \\ &= (g^{\xi_j a^{q-j+1}})^{\frac{\delta_0 a^{k'}}{\xi_{k'}(\tau^*_{k'} - \tau_{k'})} + v_\tau} \\ &= V_j^{\tilde{v_\tau}}. \end{aligned}$$

- **Challenge**. Adversary $\mathcal{A}$ submits two equal length messages m_0 and m_1. Challenger $\mathcal{C}$ flips a random coin β and encrypts m_β under the access structure $\mathbb{A}^*$, revocation list $\mathcal{R}^*$ and time T_c^* with z-ary representation τ^*. It first creates

  $$\begin{aligned} C_0 &= m_\beta \cdot T^{\delta_0} \cdot e(g^s, g^{\alpha'}), \\ C_0' &= g^s. \end{aligned}$$

  For $C_0^{''}$, we follow the technique from [1]. Let $\mathcal{R}^* = (ID_1, \ldots, ID_r)$ and $\mathcal{F}_{\mathcal{R}^*}(Z) = (Z - ID_1) \cdots (Z - ID_r) = y_1 + y_2 Z + \cdots + y_r Z^{r-1} + y_{r+1} Z^r$. If $r + 1 < R$, the coefficients $y_{r+2}, \cdots, y_R$ are set to 0. Let $\mathbf{Y} = (y_1, \cdots, y_R)^\top$ satisfy $\langle \mathbf{X}_k, \mathbf{Y} \rangle = 0$ for $k \in [1, r]$. We claim that $\mathbf{Y}^\top \cdot \mathrm{B} \cdot \boldsymbol{\nu} = 0$. Therefore, it comes that $\langle \mathbf{Y}, \boldsymbol{\alpha} \rangle = \langle \mathbf{Y}, \boldsymbol{\delta} \rangle$. $\mathcal{C}$ sets

  $$C_0^{''} = (g^s)^{\langle \mathbf{Y}, \boldsymbol{\delta} \rangle}.$$

  For $C_0^{'''}$, observe that since the challenge time is $(\tau_1^*, \ldots, \tau_{k^*}^*)$, the g^{a^i} terms in V_i are cancelled out. $\mathcal{C}$ sets

  $$C_0^{'''} = (g^s)^{\xi_0}.$$

  For C_i, we follow the technique from [38]. Since the term $h^s_{\rho^*(i)}$ contains terms of the form $g^{a^j s}$ that cannot be simulated, we need to use the secret splitting technique such that these terms can be cancelled out. $\mathcal{C}$ chooses $y_2', \ldots, y_{n^*}' \in \mathbb{Z}_p$ and then share the secret using the vector

  $$\mathbf{v}^* = (s, sa + y_2', sa^2 + y_3', \ldots, sa^{n^*-1} + y_{n^*}') \in \mathbb{Z}_p^{n^*}.$$

  This allows the terms $h^{-s}_{\rho^*(i)}$ cancel out with the terms $g^{a\lambda_i}$. For $i = 1, \ldots, n^*$, $\mathcal{C}$ generates

  $$C_i = \left(\prod_{j=2}^{n^*} (g^a)^{M^*_{i,j} y_j'} \right) (g^s)^{-z_{\rho^*(i)}}.$$

  To see the simulation correctness of C_i, we first define

  $$\begin{aligned} \lambda_i^* &= \langle \mathbf{v}^*, M_i^* \rangle \\ &= s M^*_{i,1} + (sa + y_2') M^*_{i,2} + (sa^2 + y_3') M^*_{i,3} + \ldots + (sa^{n^*-1} + y_{n^*}') M^*_{i,n^*}. \end{aligned}$$

The correct distribution of C_i should be

$$\begin{aligned}
C_i &= g^{a\lambda_i^*} h_{\rho^*(i)}^{-s} \\
&= \left(g^{saM_{i,1}^*} g^{(sa+y_2')aM_{i,2}^*} g^{(sa^2+y_3')aM_{i,3}^*} \cdots g^{(sa^{n^*-1}+y_{n^*}')aM_{i,n^*}^*} \right) \\
&\quad \cdot g^{-sz_{\rho^*(i)}} \left(g^{-saM_{i,1}^*} \cdot g^{-sa^2M_{i,2}^*} \cdots g^{-sa^{n^*}M_{i,n^*}^*} \right) \qquad \text{(from equation (6))} \\
&= g^{aM_{i,2}^*y_2'} g^{aM_{i,3}^*y_3'} \cdots g^{aM_{i,n^*}^*y_{n^*}'} (g^s)^{-z_{\rho^*(i)}} \\
&= \left(\prod_{j=2}^{n^*} (g^a)^{M_{i,j}^*y_j'} \right) (g^s)^{-z_{\rho^*(i)}}.
\end{aligned}$$

- **Phase 2**. It is the same as in Phase 1.
- **Guess**. Adversary $\mathcal{A}$ will eventually output a guess β' of β. Challenger $\mathcal{C}$ then outputs 0 to guess that $T = e(g,g)^{a^{q+1}s}$ if $\beta = \beta'$; otherwise, it outputs 1 to indicate that it believes T is a random group element $\Psi \in \mathbb{G}_T$.

When T is a tuple, the challenger $\mathcal{C}$ gives a perfect simulation so we have that the advantage of the challenger $\mathcal{C}$ is the same as the advantage of the adversary $\mathcal{A}$. When T is a random group element $\Psi \in \mathbb{G}_T$, the message m_β is completely hidden from the adversary and we have $\Pr[\mathcal{C}(\mathbf{y}, T = \Psi) = 0] = 1/2$. Therefore, the challenger $\mathcal{C}$ can win the decisional q-BDHE game with non-negligible advantage. $\square$