Attribute-Based Encryption for Fine-Grained Access Control over Sensitive Data

by

Qiuxiang Dong

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved November 2019 by the
Graduate Supervisory Committee:

Dijiang Huang, Chair
Arunabha Sen
Adam Doupé

ARIZONA STATE UNIVERSITY

December 2019

ABSTRACT

The traditional access control system suffers from the problem of separation of data ownership and management. It poses data security issues in application scenarios such as cloud computing and blockchain where the data owners either do not trust the data storage provider or even do not know who would have access to their data once they are appended to the chain. In these scenarios, the data owner actually loses control of the data once they are uploaded to the outside storage. Encryption-before-uploading is the way to solve this issue, however traditional encryption schemes such as AES, RSA, ECC, bring about great overheads in key management on the data owner end and could not provide fine-grained access control as well.

Attribute-Based Encryption (ABE) is a cryptographic way to implement attribute-based access control, which is a fine-grained access control model, thus solving all aforementioned issues. With ABE, the data owner would encrypt the data by a self-defined access control policy before uploading the data. The access control policy is an AND-OR boolean formula over attributes. Only users with attributes that satisfy the access control policy could decrypt the ciphertext. However the existing ABE schemes do not provide some important features in practical applications, *e.g.*, user revocation and attribute expiration. Furthermore, most existing work focus on how to use ABE to protect cloud stored data, while not the blockchain applications.

The main objective of this thesis is to provide solutions to add two important features of the ABE schemes, *i.e.*, user revocation and attribute expiration, and also provide a practical trust framework for using ABE to protect blockchain data. To add the feature of user revocation, I propose to add user's hierarchical identity into the private attribute key. In this way, only users whose identity is not revoked and attributes satisfy the access control policy could decrypt the ciphertext. To add the feature of attribute expiration, I propose to add the attribute valid time period into

the private attribute key. The data would be encrypted by access control policy where all attributes have a temporal value. In this way, only users whose attributes both satisfy the access policy and at the same time these attributes do not expire, are allowed to decrypt the ciphertext. To use ABE in the blockchain applications, I propose an ABE-enabled trust framework in a very popular blockchain platform, Hyperledger Fabric. Based on the design, I implement a light-weight attribute certificate authority for attribute distribution and validation; I implement the proposed ABE schemes and provide a toolkit which supports system setup, key generation, data encryption and data decryption. All these modules were integrated into a demo system for protecting sensitive files in a blockchain application.

DEDICATION

To my family for their love and support.

ACKNOWLEDGMENTS

At first I would like to thank my advisor, Dr. Dijiang Huang, for all his advising and support during my study at ASU. He provided me unreserved help in research. The study and work experience in the lab is very important and helpful to me. I also would like to thank the committee members of my thesis, Dr. Arunabha Sen and Dr. Adam Doupé. Thanks for your valuable feedback and suggestions.

Thanks my lab mates: Dr. Chun-Jen Chung, Dr. Weijia Wang, Yuli Deng, for all the valuable discussions.

Members of our Secure Networking And Computing (SNAC) Lab provided me a lot of inspiring ideas by ways of group meetings and discussions. Thanks Abdulhakim Sabur, Dr. Adel Alshamrani, Ankur Chowdhary, Dr. Bing Li, Duo Lu, Dr. Jiayue Li, Dr. Sandeep Pisharody, Sowmya Myneni and Zhen Zeng, for all the valuable discussions and suggestions.

Thanks my academic advisor Christina Sebring and Arzuhan Kavak, for all your help, timely replies and kindness.

Thanks our friends Fred Morstatter and Rio Cavendish, for helping us settle down here. Without you guys, our life here could not be so enjoyable.

Finally, I would like to thank my parents for their unconditional love and constant support. To my sister for taking care of our parents during my study here, for her unconditional love and support all through my life. To my husband, for all the encouragements and understandings — you are both a great family member and my best friend. To our ginger fluffy roommate Frankie, for making me happy.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Chapter 1

INTRODUCTION

The traditional way to perform access control over data accessible or shared by multiple entities is to build two separate services or systems, one for data storage and the other for data access control. Both are out of data owners' control. This type of access control system suffers from the problem of separation of data ownership and management. In particular, it is the data access control system enforcing the access permission over the data, while not the data owners themselves.

However, in some application scenarios, such as cloud computing and blockchain-based application, data owners could not trust the data storage providers. In cloud computing, the storage provider could have access to the stored data at any time. For the blockchain-based applications, if the sensitive data is included in transactions which are appended to the chain, any user in the system who is allowed to invoke the query method would have access to the data. Considering cloud computing platforms are being used by more and more companies and individuals [1] and the increasing number of corporations that are investing resources in optimizing their businesses by taking use of the decentralization and transparency features offered by blockchain [2], it is of great significance to provide a fine-grained access control mechanism which enables data owners to perform fine-grained access control in a discretionary way.

Attribute-Based Encryption (ABE) [5], in particular Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [6] is the way to solve all above-mentioned problems. In particular, it is a cryptographic way to implement the Attribute-Based Access

[1] https://tinyurl.com/y5rtefd2

[2] https://tinyurl.com/y6ns865p

Control (ABAC) model. Different from traditional ways to perform access control, the data ownership and management are combined together. It is the data owners themselves to define the access control policy on the data and the access control is enforced on the data directly by the way of encrypting the data with a specified access control policy, while not by a separate access control system. In the cloud computing application scenario, the data storage provider is only responsible for storing the data but not performing the access control. Even if the data is stored on the public cloud storage and accessible by everyone, they are still secure because of the encryption. As promising as it is, multiple users might share common attributes, thus making the following user and attribute management problems very difficult in practical applications. In this thesis, I propose new schemes which solve the user and attribute management problems. Furthermore, existing work focuses more on how to utilize ABE in cloud storage, while for the most recently proposed blockchain platforms, such as Hyperledger Fabric, it is still not clear how to integrate ABE to provide discretionary fine-grained access control by data owners themselves over sensitive data.

1.1 Problem Statement

In the thesis, I study the problem of constructing ABE schemes that provide features that are of great importance in practical applications and also investigate how to use ABE schemes in blockchain platforms to provide discretionary fine-grained access control over sensitive data stored on the chain. In summary, the following questions are investigated:

- How can a data owner revoke data users in a discretionary way?
- How to assign private attribute key for attributes with temporal constrains?
- How to set up an ABE-enabled trust framework in Hyperledger Fabric?

In the following, I describe the motivation of each research question. To this end, at first I give an easy-to-understand explanation of CP-ABE schemes as shown in Figure 1.1.

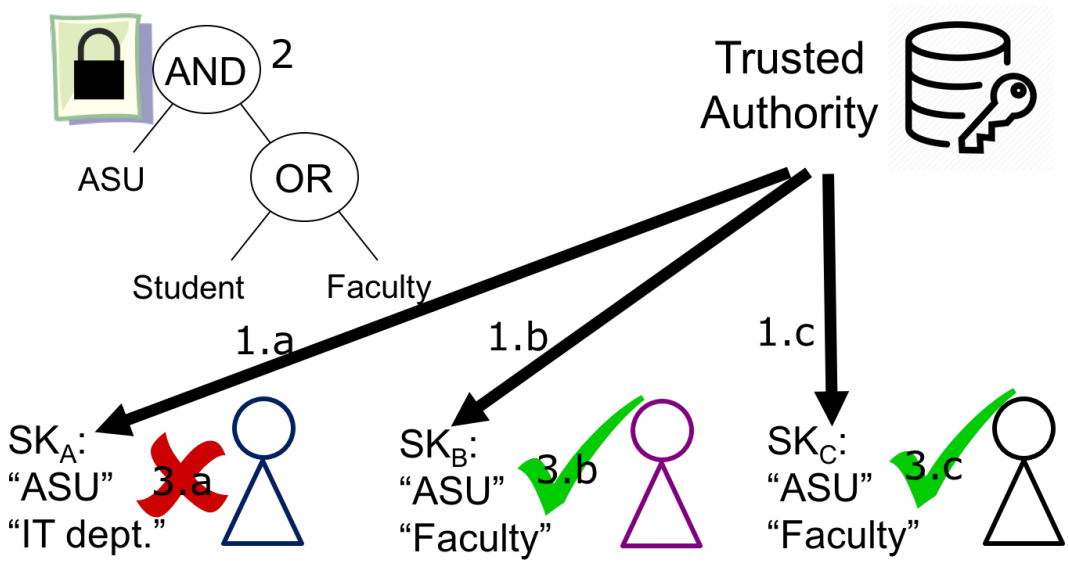

Figure 1.1: CP-ABE Example

- Step 1. 1.a: User A is assigned a private key based its attributes, *i.e.*, "ASU", "IT dept.". 1.b: User B is assigned a private key based on its attributes, *i.e.*, "ASU", "Faculty". 1.c: User C is assigned a private key based on its attributes, *i.e.*, "ASU", "Faculty".

- Step 2. Data is encrypted by the data owner under an access policy over a certain set of attributes, *e.g.*, *"ASU"* ***AND*** *("Student"* ***OR*** *"Faculty").*

- Step 3. 3.a: User A with attributes "ASU", "IT dept." fails to decrypt the ciphertext since its attributes do not satisfy the access policy. 3.b: User B

with attributes "ASU", "Faculty" succeeds to decrypt the ciphertext since its attributes satisfy the access policy. 3.c: User C succeeds in decryption in the same way as User B.

Identity Based Discretionary User Revocation

As shown in Figure 1.1, only using the existing attributes "ASU", "IT Dept.", "Student" and "Faculty", it is impossible to construct the group of authorized users {User B}. To deal with this issue, when constructing the access policy, user revocation is needed.

Previous researches define the revocation problem as attribute-based revocation [1]-[3]. The basic idea is to cease certain access privileges of users from the perspective of key generation. In particular, whenever a revocation occurs, the trusted authority generates some secret information for non-revoked users to update their private key. Since the revoked user does not have the secret updating information, the components of his/her private key corresponding to the revoked attributes will not work any more, thus achieving the goal of ceasing certain users' access privilege(s).

Although attribute-based revocation is a feasible solution to the user revocation problem in CP-ABE, it suffers the following deficiencies when applied in practice. First, the trusted authority (TA) has to be online all the time to deal with each revocation, which becomes a potential single point of failure. Second, each revocation incurs a key re-distribution procedure, which increases both the computation overheads and the communication overheads. The overheads will be especially big in the following three scenarios: the revoked users have a great number of attributes; the number of the non-revoked users sharing common attributes with the revoked user is big; user revocation happens frequently. Third, if a user is revoked with the attribute-based approach, that means he/she is a revoked user for all the data owners.

Therefore, attribute-based user revocation is suitable for revoking a compromised key. Whereas, in practice, data user revocation might be brought by other reasons, *e.g.*, conflicts of interests. Lastly, the attribute-based revocation is a stateful mechanism where once a receiver misses an update it will not be able to decrypt future messages (or this must be corrected somehow). Therefore, it is of great significance to provide a flexible user revocation scheme for the ABE-enabled access control system.

Automatic Attribute Expiration

As shown in Figure 1.1, there is an implicit assumption that all attributes assigned to users will not expire. In practice, users' attributes might only be effective for a limited time period. However existing CP-ABE schemes lack this important feature of attribute expiration. One feasible solution is to enforce this feature at the protocol level by rekeying periodically. Another way to implement attribute expiration is by leveraging attribute revocation [7, 8]. However, revocation itself is highly inefficient, usually requires rekeying of all other users, or non-monotonic access policies with negative attributes [9]. A time-based approach for attribute expiration is presented in BSW [10]. In this approach, the authority simply appends an expiration date to the attribute string.

Although the above-mentioned approaches could solve the problem. The most desirable feature of attribute expiration is that no interaction exists between users and the trusted authority when an attribute expires. Therefore, in [10], the author proposed an open problem to construct an automatic attribute expiration CP-ABE scheme. In particular, the expiration of the attribute should be embedded into the private attribute key. When decrypting, the checking of whether an attribute expires or not will be done within the crypto algorithm automatically.

Discretionary ABAC in Blockchain Applications

Data of applications built upon blockchain will be stored on the peer nodes in the blockchain network and any user who is permitted to query the data on the chain would have access privileges as well. In the most popular blockchain application, Bitcoin, all the transactions within the system are stored on all the nodes and everyone has access to the data. Considering this would pose data privacy issues, the industry proposed permissioned blockchain platforms, one of the most popular platforms is Hyperledger Fabric [3], which restricts access to the blockchain network to only pre-verified users. In particular, every member or user will be assigned an identity which is actually a public/private key pair as well as a certificate which binds the public key with the userID. However, with a single user identity, it is hard to implement fine-grained access control, especially when the chaincode authors did not know the identity of the users of the application in advance. To deal with this issue, attribute-based access control [4] is introduced into the chaincode logics. When registering a user into the system, the administrator would also assign a set of attributes to the user. With this mechanism, now the chaincode could do a more fine-grained access control, for example, only users with a particular attribute(s) could query a certain set of resources. However, this solution does not solve the issue of separation of data ownership and data management. It's the chaincode writer defining the access control policy but not the asset owners. To this end, in this thesis, I propose a new framework which combines the attribute-based access control model in Hyperldeger fabric and the attribute-based encryption to achieve discretionary attribute-based access control in Hyperledger Fabric.

[3]https://www.hyperledger.org/projects/fabric

[4]https://tinyurl.com/y6a5uyju

1.2 Contributions

To answer the research questions above, the contributions of the thesis are summarized as follows.

- I propose firstly a data owner discretionary CP-ABE user revocation mechanism. During data sharing, the owners are able to revoke any user directly without turning to TA or re-distributing private keys. I propose a new primitive DUR-CP-ABE that supports both revocation of particular users and affiliated users revocation in a batch. I present a construction of the DUR-CP-ABE scheme and prove that the construction is secure in terms of the proposed security model. I perform performance evaluation and show that the proposed DUR-CP-ABE construction is practical for real-world applications.

- I extend the identity revocable CP-ABE scheme to support federation, delegation and inter-operability in system setup and private key generation. Through federation, no single trusted authority owns the master secret key, while it is multiple trusted authorities working in a federated way to generate the public parameter and root organization's private attribute key, thus splitting the master secret key into shares owned by all of them. Federation solves the problem of single point of failure and single point of trust issue. Delegating TA's key generation capability can not only relieve the single-point failure issue, but also naturally follow the organization's hierarchical management structure. In practice, users need to interact with several different organizations that may not trust each other; a user also may own attributes belonged to different subdivisions of an organization. Inter-operability enables a user to get private keys from different organizations or subdivisions.

- I propose firstly an ABE scheme supporting automatic attribute expiration with the following salient features: Decentralization: in the proposed access control system, the data owner is able to share data based on an access control policy written over attributes issued across multiple authorities in different organizations with different roots of trust. Automatic Attribute Expiration: the proposed scheme assigns a time period of validity for each attribute during the key generation phase. When the data owner encrypts data, a current time is attached to each attribute in the access policy tree. In this way, the data owner could make a more fine-grained access control on the time dimension. Only users whose attributes are still valid when the data is encrypted and shared could decrypt the ciphertext.

- I propose an ABE-enabled trust framework for the popular blockchain platform Hyperledger Fabric. I implement the trust framework by providing lightweight attribute authorities which provide proof of users' attributes by taking use of the bilinear short signature Boneh *et al.* (2001) which is much more efficient compared with the traditional signature schemes, such as RSA, DSA, ECDSA, *etc.*. I implement an ABE toolkit for private key distribution and data encryption/decryption. I implement a file encryption/decryption toolkit to work together with ABE scheme so that data of any format could be encrypted/decrypted. I also provided a Hyperledger Fabric example application to show how ABE could be used to provide discretionary, fine-grained attribute-based access control over the sensitive data stored in the blockchain system.

1.3 Thesis Organization

Rest of the thesis is organized as follows. In Chapter 2, I describe related work of this thesis. In Chapter 3, I provide mathematical preliminaries used in this thesis. In

Chapter 4, I present an ABE scheme which supports hierarchical identity-based user revocation. In Chapter 5, I present an ABE scheme which extends the scheme Chapter 4 to support federation, delegation and inter-operability. In Chapter 6, I describe an ABE scheme which supports automatic attribute expiration. In Chapter 7, I describe how to enable discretionary attribute-based access control with ABE in Hyperledger Fabric applications. Chapter 8 concludes the thesis and describes future work.

Chapter 2

RELATED WORK

In this chapter, I will introduce some background knowledge of this thesis, including the attribute-based access control model, the history of ABE and its extension and an introduction of the blockchain technology.

2.1 Access Control

2.1.1 Definition

Access Control is defined as Harris and Maymi (2016) security features which control the way users and systems communicate and interact with other systems and resources. The purpose is to protect systems and resources from unauthorized access. The following are some important terms I will use when talking about access control Harris and Maymi (2016).

- **Access**: it is the flow of information between a subject and an object.
- **Subject**: it is an active entity that requests access to an object or the data within an object. It could be a user, a program or a process, *etc.*
- **Object**: it is a passive entity that contains information or needed functionality. It could be a database, a file, a column in a database's table, *etc.*

Take the cloud storage as the example: a user who requests to access the files stored in the cloud storage server is a subject, and the files are the objects. The flow of request and response information between the user and the files is access.

2.1.2 Access Control Model

An access control model is a framework defining how subjects access objects. Currently there are mainly three different types of access control models, *i.e.*, discretionary, mandatory, and role-based access control. In the following, I will briefly introduce each type of these access control models Harris and Maymi (2016).

- **Discretionary Access Control (DAC)**: this access control model enables the owner of the object to specify which subjects can access this object. The term "discretionary" means the control of the access is on the basis of the object's owner's discretion. Access Control List (ACL) is the most common implementation of the DAC model. Most of the popular operating systems are based on DAC models. In the properties of a file, choices that allow users to control by whom, to what degree this file could be accessed could be found. DAC is based on identities of users. The identities could be user identity or group membership identity.

- **Mandatory Access Control (MAC)**: the MAC model is more strict and structured than the DAC model. It is built on basis of a security label system. Data are classified by security clearance (*e.g.*, secret, top secret, confidential, *etc.*) and users are classified in the same way. For a user's access request, it is the user's security clearance, the object's classification and the system's security policy all together making the decision. MAC-based systems are usually used by government organizations for maintaining confidential information.

- **Role-Based Access Control (RBAC)**: By the DAC model, the access control is specified at the object level by the way of ACLs. The ACL administrator who is permitted to create and edit the ACL is responsible for translating

an organizational authorization policy into permission. With the increasing number of objects and subjects, some subjects will be granted unnecessary access to some objects. These unnecessary permissions violate the least-privilege rule and increase data leakage risks. Different from DAC where access is based on user identity, the RBAC model is based on role which is defined in terms of the operations and tasks the role is going to carry out. With RBAC, the permissions are assigned in an implicit way, *i.e.*, they are assigned to a role or group but not directly to a user, and the user just inherits those attributes.

2.2 Attribute-Based Access Control

2.2.1 Motivation

The following is a brief review of some concepts in access control Harris and Maymi (2016) below.

- **Identification**: subjects supplying identification information, *e.g.*, username, userID, account number, *etc.*.
- **Authentication**: verifying the identification information with information such as password, PIN value, fingerprint, *etc.*.
- **Authorization (Access Control)**: using the identity of the subject together with other criteria to make a determination of operations that a subject can carry out on objects, *i.e.*, "I know who you are, now what am I going to allow you to do?"

The traditional access control models described above are based on a user's identity in the following steps as presented in Figure 2.1, either directly or by the way of predefined attribute types, *e.g.*, roles/groups, assigned to the identity Hu *et al.*

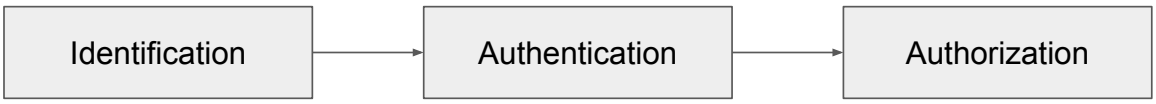

Figure 2.1: Three Steps for a Subject to Access an Object: Identification, Authentication, Authorization.

(2015). However, this type of access control models suffer from the following issues Hu *et al.* (2015).

- **Inter-organization authorization issue**: in non-ABAC inter-organizational access requests, *e.g.*, user userA from organization A would request to access organizationB's resources, the user has to get its identity pre-provisioned in organizationB and added to the requested' object's ACL before the access could be performed successfully.

- **Expressiveness issue**: the identity, roles or groups are not sufficient for expressing the access control needs in real-world applications. Take RBAC as the example, its role assignments are usually based on static organizational positions, thus presenting challenges for applications where dynamic access control decisions are required. Although with ad-hoc roles, dynamic access control could be implemented, it will lead to the issue of "role explosion".

The biggest issue of the traditional access control models is that previous knowledge of object by the subject or knowledge of the subject by the object owner must be provided Hu *et al.* (2013). To this end, an alternative approach to grant or deny user access requests on the basis of the user's attributes, object's attributes and environmental conditions were proposed. This type of access control model is named Attribute-Based Access Control (ABAC) Hu *et al.* (2013).

2.2.2 Definition

Definition 2.1. Attribute Based Access Control (ABAC): An access control method where subject requests to perform operations on objects are granted or denied based on assigned attributes of the subject, assigned attributes of the object, environment conditions, and a set of policies that are specified in terms of those attributes and conditions.

Besides the terms explained in section 2.1, in the following, terms *attribute, operation*, *environment condition* and *policy* are explained Hu *et al.* (2013).

- **Attributes**: characteristics of the subject, object, or environment conditions. Usually it is denoted by a key-value pair.
- **Operation**: executions of a function at the request of a subject upon an object. Common operations include "read", "write", "execute", "edit", *etc.*.
- **Environment Condition**: it is defined as detectable environmental characteristics. It is independent of both subject and object. It may include time, location, *etc.*.
- **Policy**: the representation of rules or relationships which enables to determine whether an access request should be permitted, given the values of the subject's attributes, object's attributes, and possibly environment conditions.

As described in Figure 2.2, step 1 shows the process of a subject sending access request to an object; step 2a, 2b, 2c, 2d shows that the ABAC access control mechanism evaluates a) policy, b) subject attributes, c) object attributes and d) environment conditions, to make a decision; step 3 shows that the subject is given access to the object if permitted.

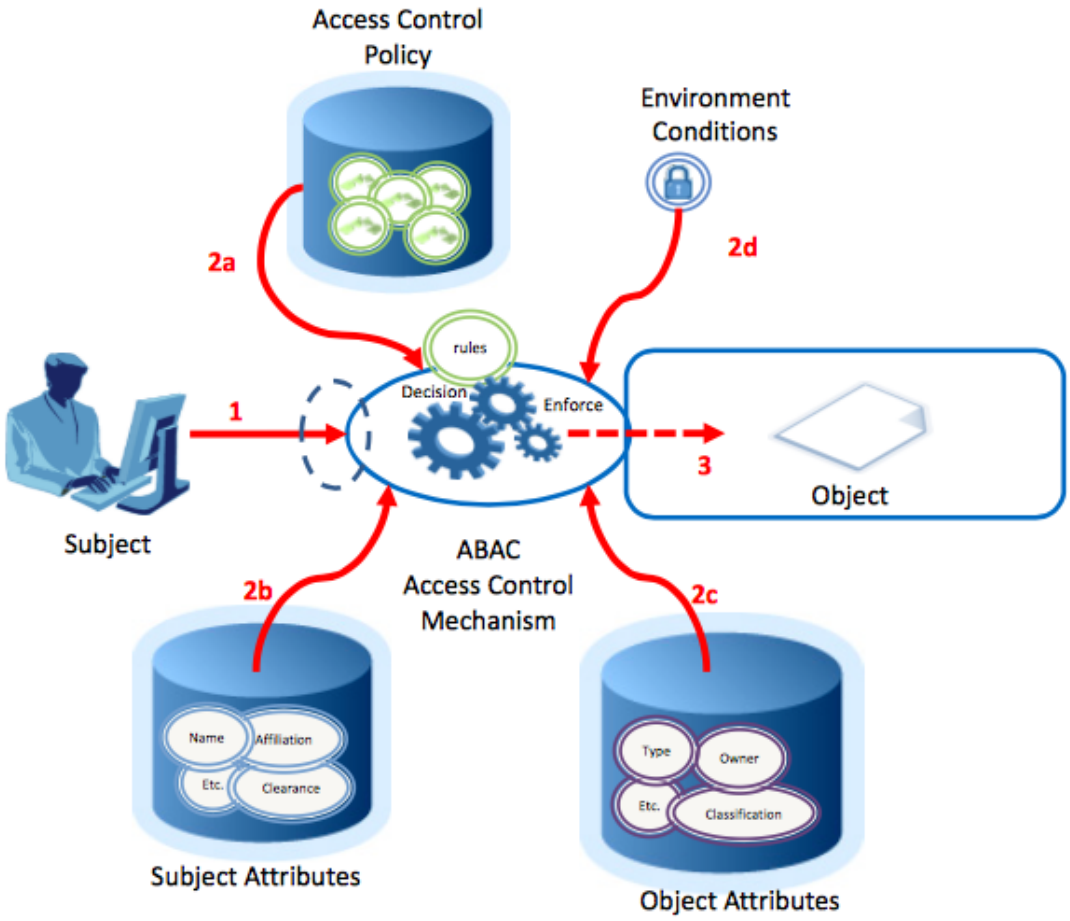

Figure 2.2: Basic ABAC Scenario Hu *et al.* (2013)

2.2.3 Benefits

In summary, the ABAC model provides the following benefits compared with the traditional access control models Hu *et al.* (2015).

- since the access control decision is made based on a higher number of inputs (subject attributes, object attributes, environment conditions, and policy), a larger set of possible combinations could be used to get a larger number of and more expressive policies.

- the access decisions vary between different requests simply based on different attribute values, while not change the policy itself.
- there is no need to get prior knowledge of the relationship between the subject and object. When there are members joining or leaving the system, it is not necessary to modify the policies.

2.3 Attribute-Based Encryption

2.3.1 Motivation

The development of Attribute-Based Encryption (ABE) is similar to how the ABAC model was proposed. The goal is to share encrypted data among multiple users so that only authorized users could decrypt the ciphertexts. Since ABE is a public key encryption scheme, in the following I review the development history starting from public key encryption.

Traditional Public Key Encryption

In public key encryption schemes, each user has a randomly generated public/private key pair. The private key is kept secret by the user and the public key is published to the public. Whenever user B sends a message to user A, he will at first get user A's public key pk_A and then encrypt the message with the encryption algorithm with inputs of the message and pk_A. Only user A who has the private key sk_A could decrypt the ciphertext. However, if a malicious user C cheated B into using her public key pk_C to encrypt the message, then C is capable to decrypt the ciphertext while B still thought he sent the message to A. Therefore, it is very important to get users' public keys in a trustworthy way. To deal with this issue, Public Key Infrastructure (PKI) was proposed. Each user will publish its public key with a certificate which includes

both the user's identity and its public key. This could be considered as a trustworthy map from a user's identity to its public key. It is computationally impossible for the malicious user C to create a certificate which maps user B's identity to pk_C.

PKI management overhead is very big. It would be better if there is a natural binding between a user's identity and its public key. To this end, Shamir (1984) proposed identity-based cryptosystems and signature schemes.

Identity-Based Encryption

The Identity-Based Encryption (IBE) is a special type of public key encryption with an extra twist. In IBE, the public key is not randomly generated, it could be a string that uniquely identifies this user. For example, it could be an email address, a social security number, telephone number, *etc.*, or their combinations.

For the use case described in the section "Traditional Public Key Encryption", when user B wants to send secret message to user A, there is no need to get the pk_A anymore, while he could just encrypt the message with user A's identity ID_A, for example, user A's email. Only user A whose identity is verified to be ID_A could decrypt the ciphertext. The malicious user C could not make the fake binding of ID_A and pk_C anymore since in the IBE cryptosystem, $(ID_A = pk_A) \neq (ID_C = pk_C)$.

The IBE cryptosystem was firstly proposed in the year of 1994, while the first scheme was constructed in 2001 in Boneh and Franklin (2001).

Fuzzy Identity-Based Encryption

Since the first construction of IBE scheme in Boneh and Franklin (2001), there are more constructions based on different mathematical assumptions Waters (2005); Cocks (2001); Boneh and Boyen (2004). All these schemes have a common feature that identities are viewed as a string of characters.

Considering biometric identities which inherently include some noises in each sample, Sahai and Waters (2005) proposed fuzzy identity-based encryption which provides error-tolerance property. In fuzzy IBE, an identity is viewed as a set of descriptive attributes. For example, to encrypt a message M with an identity ID, the ciphertext is $C = ENC(M, ID)$. A user with identity ID' is capable to decrypt the ciphertext C if and only if ID and ID' are close to each other. The distance is measured by the "set overlap" distance metric. It is in Sahai and Waters (2005) where the concept "attribute-based encryption" was proposed. However in this paper, "attribute-based encryption" was just defined as a type of application where the data owner wishes to encrypt a file to all data users who were assigned a certain set of attributes.

In an "attribute-based encryption" system, both ciphertexts and users' keys are labeled with a set of attributes. A user could decrypt the ciphertext only if there is a match between the user's and the ciphertext's attributes. The decryption of the fuzzy IBE based ABE succeeds only when at least a pre-defined threshold, say k, attributes overlapped. Although this is enough for the biometrics identity applications, this type of ABE lacks expressibility, thus limiting its applications in other larger system Goyal *et al.* (2006). To this end, Goyal *et al.* proposed a new crypto-system, which used the same name, "Attribute-Based Encryption".

2.3.2 Definition

Based on where the access policy is enforced, ABE schemes could be classified into two types, *i.e.*, Key-Policy ABE (KP-ABE) where access policy is defined on user's private keys, and Ciphertext-Policy ABE (CP-ABE) where access policy is defined on the ciphertext. In the following, I describe both scheme's algorithm syntax.

Key-Policy Attribute-Based Encryption

According to the definition in Goyal *et al.* (2006), KP-ABE scheme consists of the following algorithms.

- **Setup** This is a randomized algorithm that takes no input other than the implicit security parameter. It outputs the public parameters PK and a master key MK.
- **Encryption** This is a randomized algorithm that takes as inputs a message m, a set of attributes γ, and the public parameters PK. It outputs the ciphertext E.
- **Key Generation** This is a randomized algorithm that takes as inputs an access structure $\mathbb{A}$, the master key MK and the public parameters PK. It outputs a decryption key D.
- **Decryption** This algorithm takes as input the ciphertext E that was encrypted under the set of attributes $\mathbf{S}$, the decryption key D for access control structure $\mathbb{A}$ and the public parameters PK. It outputs the message M if $\mathbf{S}$ satisfies $\mathbb{A}$.

Ciphertext-Policy Attribute-Based Encryption

According to the definition in Bethencourt *et al.* (2007a), CP-ABE scheme consists of the following algorithms.

- **Setup** The setup algorithm takes no input other than the implicit security parameter. It outputs the public parameters PK and a master key MK.
- **Encryption** The encryption algorithm takes as input the public parameters PK, a message M, and an access structure $\mathbb{A}$ over the universe of attributes.

The algorithm will encrypt M and produce a ciphertext CT such that only a user who possesses a set of attributes that satisfies the access structure will be able to decrypt the message. I will assume that the ciphertext implicitly contains $\mathbb{A}$.

- **Key Generation** The key generation algorithm takes as inputs the master key MK and a set of attributes S that describe the key. It outputs a private key SK.

- **Decrypt** The decryption algorithm takes as inputs the public parameters PK, a ciphertext CT, which contains an access policy $\mathbb{A}$, and a private key SK, which is a private key for a set of attributes S. If the set of attributes S satisfies the access structure $\mathbb{A}$ then the algorithm will decrypt the ciphertext and return a message M.

2.3.3 Benefits

- Discretionary access control: Compared with the traditional monitoring-based access control system, ABE provides a way for users to perform access control over their data in a discretionary way.

- Access control goes together with data: the data themselves contain access control. No matter where the data is copied or leaked, they are in an encrypted format and only authorized users could decrypt the data.

- Fine-grained access control: ABE is a type of attribute-based access control model, thus being expressive and flexible.

2.4 Blockchain

2.4.1 Bitcoin

Motivation

Traditional e-commerce relies on a trusted third party, *e.g.*, financial institutions, to deal with electronic payments. This suffers the following drawbacks Nakamoto (2008):

- mediation costs such as transaction costs. This limits the minimum practical transaction size and also cuts off possible small casual transactions.
- collecting unnecessary customer information, thus leading to privacy issues.
- unavoidable frauds.

To this end, an anonymous individual or organization named Nakamoto Satoshi proposed a peer-to-peer electronic cash system, named Bitcoin Nakamoto (2008). It allows online payment sent from one party to another directly without relying on a third trusted financial institution.

Blockchain Underlying Bitcoin

As shown in Figure 2.3, an electronic coin is a chain of digital signatures. Each owner in the system has a public/private key pair. A coin is transferred to the next owner by creating a digital signature with inputs of a hash of the previous transaction and the next owner's public key, and then append these to the coin. A payee verifies the coin ownership by verifying these signatures.

The chain above could not prevent a payer from double-spending, *i.e.*, send the money to two different payees by creating two signatures one for each payee's public

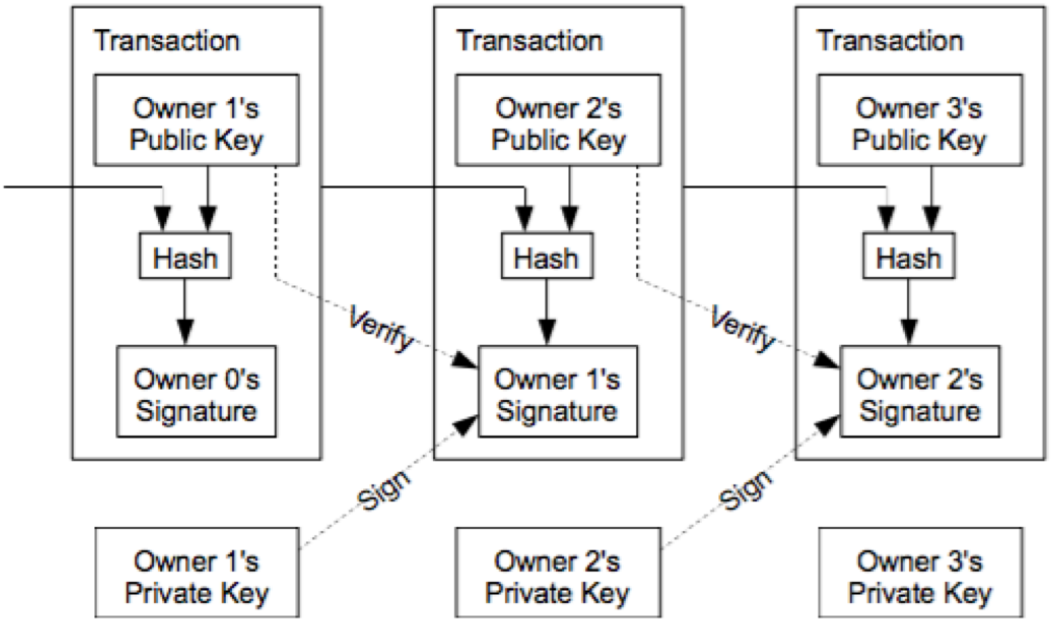

Figure 2.3: Blockchain Underlying Bitcoin Nakamoto (2008)

key. To prevent this double-spending problem, timestamp server was introduced. The timestamp server takes a hash of block of times to be timestamped (as shown in Figure 2.4) and then publishes the hash.

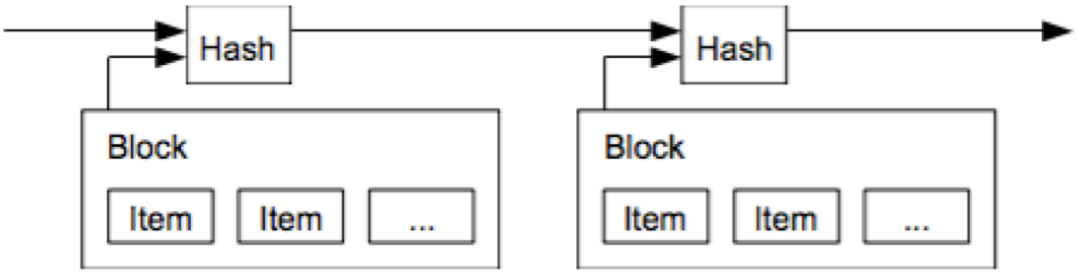

Figure 2.4: Timestamp Server Proposed in Bitcoin Nakamoto (2008)

To implement a peer-to-peer timestamp server so that the temporal sequence of the block is consistent system-wide, Bitcoin proposed a consensus protocol named

Proof-Of-Work (POW). It involves finding a value in a brute-force way, so that when hashed, the hash value begins with a pre-defined number of zero bits. In Bitcoin, as shown in Figure 2.5 it is implemented by increasing a nonce until it gives the block's hash value the required number of zero bits. If the malicious payee wants to double-spend a coin, it has to re-do the proof of work, *i.e.* for the block involving the modified transaction as well as all the following blocks, also it has to catch up with the work of the other benign nodes. The longest chain in the system represents the system consensus. It is essentially one-CPU-one-vote. Therefore, if the majority of the computation power in the system is controlled by the honest nodes, then the malicious party will fail in creating a longer chain.

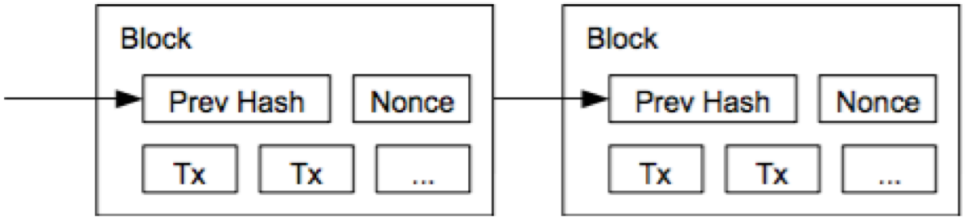

Figure 2.5: Proof-Of-Work in Bitcoin Nakamoto (2008)

The following summarizes the workflow of the Bitcoin network Nakamoto (2008).

- New transactions are broadcast to all nodes.
- Each node collects new transactions into a block.
- Each node works on finding a difficult proof-of-work for its block.
- When a node finds a proof-of-work, it broadcasts the block to all nodes.
- Nodes accept the block only if all transactions in it are valid and not already spent.

- Nodes express their acceptance of the block by working on creating the next block in the chain, using the hash of the accepted block as the previous hash.

2.4.2 The Blockchain Technology

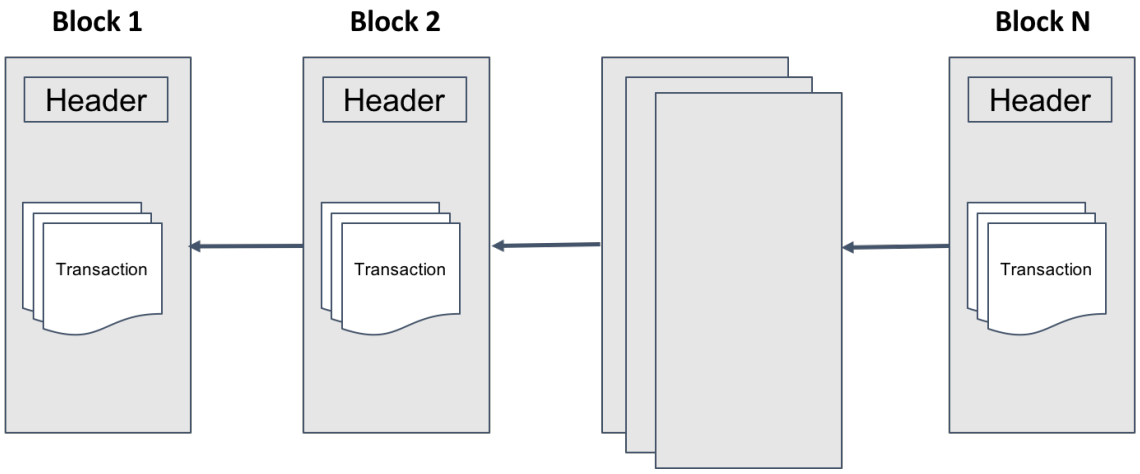

Figure 2.6: Blockchain in General

As shown in Figure 2.6, blockchain is considered to be a distributed and immutable transaction ledger. It is replicated across a distributed network of peer nodes each of whom collaborate in its maintenance. The blockcahin transactions are run by the nodes when they have been validated by a consensus protocol. These transactions will be grouped into blocks, which include a hash value that cryptographically links each block to the preceding block, thus forming a chain of blocks where the terminology *Blockchain* comes from Semiconductors (2019).

2.4.3 Blockchain Classification

Blockchain can be broadly classified as permissionless blockchains e.g., Bitcoin Nakamoto (2008), Etherum Wood (2014), etc., and permissioned blockchains e.g., Hyperledger Semiconductors (2019). With permissionless blockchain solutions, trans-

actions are executed on all the nodes in the network. Therefore, both source codes, (*i.e.*, smart contracts or chaincode) and transaction data are revealed in public, thus it has significant concerns on privacy and confidentiality for smart contracts. For instance, in a distributed supply-chain application Semiconductors (2019), a supplier does not want to reveal business data for a specific business partner (e.g., pricing quote, product information) to other irrelevant business partners.

Permission based blockchain solutions such as Hyperledger are designed to address the data/transaction privacy issues by creating a trusted permission group. It operates a distributed ledger among a set of identified participants under a governance model, thus eliminating the complicated consensus model, *e.g.,* PoW Gervais *et al.* (2016). Furthermore, identifiers allow the end-user of the system to control to which degree it interacts and shares information with the other parties inside and outside the system. In Hyperledger, a ledger corresponds to a channel. The ledger can only be shared by the members on the channel. In this way, unauthorized users and network nodes will be excluded from the blockchain network.

Chapter 3

ABE PRELIMINARIES

In this chapter, I will introduce some preliminaries of attribute-based encryption schemes.

3.1 Mathematical Background

3.1.1 Group and Cyclic Group

In mathematical context, a group $\mathbb{G}$ is a set of elements equipped with a binary operator $\times$ that are related with each other according to the following four well-defined conditions called group axioms.

- Closure: for any two elements x and y, $x \times y$ is in the group $\mathbb{G}$ as well.
- Associativity: for any three elements x, y and z, $(x \times y) \times z = x \times (y \times z)$.
- Identity element: there exists an element $e \in \mathbb{G}$ such that for every element $x \in \mathbb{G}$, $x \times e = e \times x = x$. The identity element e is denoted by 1.
- Inverse element: for each element $x \in \mathbb{G}$, there exists an element y such that $x \times y = y \times x = e$, where e is the identity element. y is the inverse element of x, and is denoted by x^{-1}.

A group is called cyclic if there exists at least one element $g \in \mathbb{G}$ such that all the elements in the group are powers of it. When using multiplication as the group binary operator shown as above, the elements of the group can be denoted by

$$\cdots, g^{-3}, g^{-2}, g^{-1}, g^0 = e, g, g^2, g^3, \cdots$$

3.1.2 Prime Order Bilinear Pairing

Prime order pairing is a bilinear map function $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, where $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are three cyclic groups with large prime order p. The $\mathbb{G}_1$ and $\mathbb{G}_2$ are additive group and $\mathbb{G}_T$ is multiplicative group .The discrete logarithm problem on $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are hard. Pairing has the following properties:

- *Bilinearity*:

$$e(g^a, h^b) = e(g, h)^{ab},\ \forall g \in \mathbb{G}_1, h \in \mathbb{G}_2, a, b \in \mathbb{Z}_p^*.$$

- *Nondegeneracy*:

  $e(g_0, h_0) \neq 1$ where g_0 is the generator of $\mathbb{G}_1$ and h_0 is the generator of $\mathbb{G}_2$.

- *Computability*:

  There exists an efficient algorithm to compute the pairing.

3.1.3 Composite Order Bilinear Pairing

There are multiple types of composite order bilinear pairing. In the following, I present one used in this thesis.

A three-prime composite pairing Lewko and Waters (2011): the three-prime composite pairing is a bilinear map function $e : \mathbb{G}^2 \to \mathbb{G}_T$ where $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order $N = p_1 p_2 p_3$ and p_1, p_2 and p_3 are distinct primes. The map function satisfieS the following conditions:

- Bilinear: $\forall g, h \in \mathbb{G}$, $a, b \in \mathbb{Z}_N$, $e(g^a, h^b) = e(g, h)^{ab}$.

- Nondegerate: $\exists g \in \mathbb{G}$ such that $e(g, g)$ has order n in $\mathbb{G}_T$.

Assume that the group operations in $\mathbb{G}$ and $\mathbb{G}_T$ as well as the bilinear map e are computable in polynomial time. $\mathbb{G}_{p_1}, \mathbb{G}_{p_2}$ and $\mathbb{G}_{p_3}$ denotes the subgroups of order p_1, p_2 and p_3 in $\mathbb{G}$ respectively. Suppose $h_i \in \mathbb{G}_{p_i}$ and $h_j \in \mathbb{G}_{p_j}$ for $i \neq j$.

- Orthogonality: $e(h_i, h_j)$ is the identity element of $\mathbb{G}_T$.

3.2 Access Structure

3.2.1 Definition

Definition 3.1. Access Structure Goyal *et al.* (2006). Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ 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 $\{P_1, P_2, \ldots, P_n\}$, *i.e.*, $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are defined as authorized sets, and sets that do not belong to $\mathbb{A}$ are defined as unauthorized sets.

3.2.2 Linear Secret Sharing Scheme

LSSS matrices can be utilized to express any monotonic access structure, which are the most commonly used access structures in most CP-ABE schemes. The algorithm of constructing an LSSS matrix is proposed by Lewko and Waters. Linear Secret Sharing Scheme, *a.k.a.*, LSSS is defined as follows.

Definition 3.2. Linear Secret Sharing Schemes(LSSS) Lewko and Waters (2011). A secret sharing scheme Π over a set of parties is called linear over $\mathbb{Z}_p$ if the following two conditions are satisfied

- the shares for each party form a vector over $\mathbb{Z}_p$;
- a share-generating matrix for Π has ℓ rows and n columns. For all $i = 1, \ldots, \ell$, the i^{th} row of M, I define $\rho(i)$ as the party labeling row i. For the column vector

$v = (s, r_2, r_3, \ldots, r_n)$ where $s \in \mathbb{Z}_p$ is the shared secret and $r_2, r_3, \ldots, r_n \in \mathbb{Z}$ are randomly chosen numbers, then Mv is the vector of ℓ shares of the secret s according to Π, where the share $(Mv)_i$ belongs to party $\rho(i)$.

As shown in Beimel (1996), every linear secret sharing-scheme according to the above definition also enjoys the following **linear reconstruction** property:

Assume that Π is an LSSS for the access structure $\mathbb{A}$. Define $\mathbf{S} \in \mathbb{A}$ as an authorized set and $\mathbf{I} \subset [1, l]$ as $\mathbf{I} = \{i : \rho(i) \in \mathbf{S}\}$. Then, constants $\{w_i \in \mathbb{Z}_p\}_{i \in \mathbf{I}}$ can be derived in polynomial time such that for valid shares $\{\lambda_i\}$ of any secret s, $\sum_{i \in \mathbf{I}} w_i \lambda_i = s$.

3.2.3 Conversion Algorithm

A monotonic access structure could be expressed by an access tree with "AND", "OR" gates as interior nodes and attributes as leaf nodes. Lewko and Waters' conversion algorithm works as follows.

3.2.4 Access Structure Example

Here, I show an example of how to convert a boolean formula to an access tree, and from an access tree to an access structure that is expressed by an LSSS matrix.

Denote the attribute by A_1, A_2, A_3, A_4 and A_5. The example boolean formula is as follows

$$A_5 \wedge ((A_1 \wedge A_2) \vee (A_3 \wedge A_4))$$

. The access tree with "AND" and "OR" gate is presented as in Figure 3.1. Each subset of the rows of matrix M includes $(1, 0, 0)$ in its span if and only if the corresponding attributes satisfy the boolean formula $A_5 \wedge ((A_1 \wedge A_2) \vee (A_3 \wedge A_4))$.

If a user has attributes A_1, A_2 and A_5, then the corresponding row of the matrix will be $(1, 1, 0)$, $(0, -1, 1)$ and $(0, 0, -1)$. If these three vectors are added together,

Algorithm 3.1 The Lewko-Waters Algorithm

Input: An Access Tree T, $c = 1$

Output: The Corresponding LSSS Matrix

1: **for** each level of the tree T

2: **for** each node N in T

3: **if** the parent node is an OR gate labeled by the vector v

4: **then**

5: Label the left child of N by vector $\leftarrow v$

6: Label the right child of N by vector $\leftarrow v$

7: **else**

8: Pad N's vector with 0 at the end (if necessary) to make it of length c

9: Label the left child by vector $\leftarrow v \| 1$

10: Label the right child by vector $\leftarrow (0, \cdots, 0) \| -1$ where $(0, \cdots, 0)$ denotes the vector $\mathbf{0}$ of length c

11: $c \leftarrow c + 1$

12: **endif**

13: **end loop**

14: **end loop**

$(1, 0, 0)$ could be obtained. This is the most important features used to construct an attribute-based encryption scheme, *i.e.*, only authorized user could recover $(1, 0, 0)$. If a random vector $v = (s, r1, r2)$ is created and M is changed to be $M = Mv$, then the recovered vector would be $(s, 0, 0)$. The value s is used in the encryption scheme to hide the message.

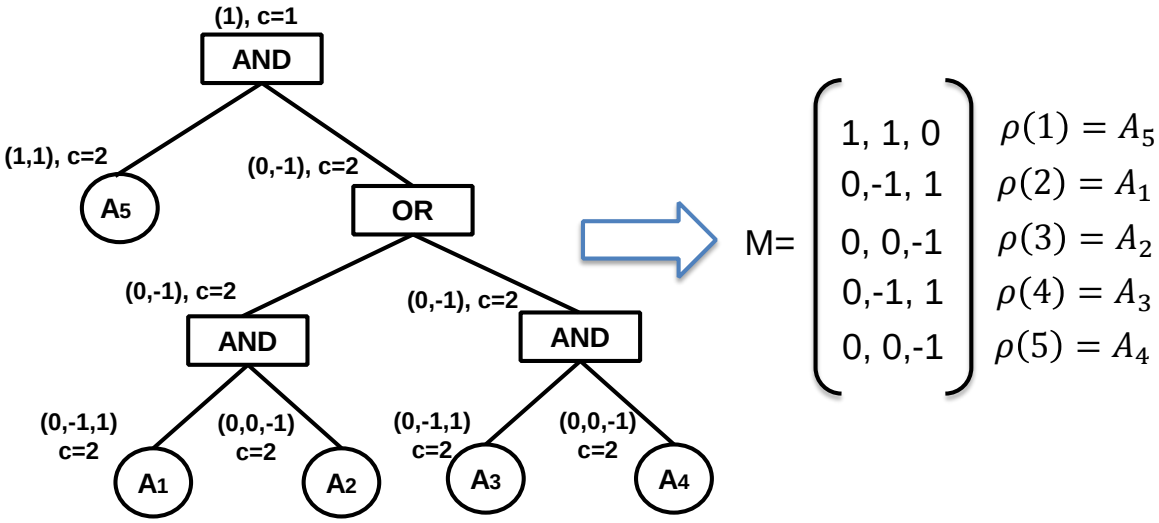

Figure 3.1: Access Tree and LSSS Matrix for Boolean Formula $A_5 \wedge ((A_1 \wedge A_2) \vee (A_3 \wedge A_4))$

Chapter 4

USER REVOCATION IN ABE

This chapter focuses on the problem of user revocation in ABE. I will introduce the background of this problem, propose a new ABE scheme which supports user revocation, and describe a construction and perform performance evaluation.

4.1 Background

4.1.1 Motivation

Because of its superior flexibility, accessibility, and capacity compared to traditional computing and storage methods, cloud computing is quickly becoming a mainstay for many companies today. Storage services over cloud, *e.g.*, Microsoft's Azure storage, Amazon's S3 and Google Cloud, are a fundamental component of cloud computing, which allows consumers to outsource their data to remote cloud servers. Storing data in the cloud relieves consumers from the burden of maintaining their data, which is usually extremely costly. However, service consumers and providers are not in the same trust domain, therefore the consumers would worry about their data privacy. Both malicious insiders, such as administrators, and outside attackers, such as hackers with root rights, may have full access to the server and consequently to consumers' data. Frequent cloud data leakage accidents in the recent years have raised both service consumer and provider concerns regarding the privacy of cloud-stored data and pushed the development of using cryptography to support access control on the cloud.

Ciphertext Policy Attribute-Based Encryption Bethencourt *et al.* (2007b) is regarded as one of the most expressive technologies and is a natural fit for attribute-based access control in cloud storage. In CP-ABE, each user is entitled a set of attributes, which are embedded into the private key by the trusted authority (TA) that is responsible for system setup and key generation/distribution. A data owner enforces an access policy over the shared data directly by encrypting the data with the access structure extracted from the access policy. Instead of by the server, the access checking is done "inside the cryptography", where only data users with eligible attributes (*i.e.*, satisfying the access structure) could decrypt the ciphertext. Different from identity-based and role-based cryptographic schemes, the public key and ciphertext size of CP-ABE are not related with the number of data users and no interactions among data owners and data users are needed. Moreover, CP-ABE is resistant against collusion attacks from unauthorized users. All these nice properties make CP-ABE very suitable for implementing fine-grained access control for secure data sharing in cloud computing where the storage server cannot be fully trusted.

As promising as it is, multiple users might share common attributes, thus making user management, especially user revocation extremely difficult when applying state-of-the-art CP-ABE schemes in practice. Previous researches define the revocation problem as *attribute-based revocation* Yu *et al.* (2010); Li *et al.* (2013); Hur and Noh (2011); Yang *et al.* (2013). However this solution suffers from the deficiencies described in Chapter 1.

4.1.2 Existing Solution

Boldyreva *et al.* Boldyreva *et al.* (2008) proposed an identity-based scheme with efficient user revocation capability. It applies key updates with significantly reduced computational cost based on a binary tree data structure, which is also applicable

to KP-ABE and fuzzy IBE user revocation. However, its applicability to CP-ABE is not clear. Libert *et al.* Libert and Vergnaud (2009) proposed an identity-based encryption scheme with stronger adaptive-ID sense to address the selective security issue of Boldyreva *et al.* (2008). Lewko *et al.* Lewko *et al.* (2010) proposed two novel broadcast encryption schemes with effective user revocation capability. EASiER Jahid *et al.* (2011) architecture is described to support fine-grained access control policies and dynamic group membership based on attribute-based encryption. It relies on a proxy to participate in the decryption and enforce revocation, such that the user can be revoked without re-encrypting ciphertexts or issuing new keys to other users. Chen *et al.* Chen *et al.* (2012) presented an identity-based encryption scheme using lattices to realize revocation. Li *et al.* Li *et al.* (2015) first introduced outsourcing computation in identity-based encryption and presented a revocable scheme in the server-aided settings. It achieves constant computation cost at public key generator and private key size at user end, and the user does not have to contact public key generator for key update.

To this end, I propose a new scheme named Discretionary User Revocable CP-ABE, DUR-CP-ABE for short Dong *et al.* (2018). Different from previous attribute-based approaches, the proposed scheme supports *identity-based revocation*. In the key generation phase, on the one hand attributes are allocated to users as in state-of-the-art CP-ABE schemes, on the other hand a unique identity (ID) is assigned to each user. That is, both attributes and the ID are embedded into a user's private key. The encryption algorithm works by two steps: first, specify attribute literals in conjunctive/disjunctive normal forms as an access structure to cover the recipients of the target group of data users; second, revoke undesired users by incorporating their identities into the ciphertext. In this way, only users whose attributes satisfy the access structure and meanwhile are not revoked by the data owners could decrypt

the ciphertext. In order to revoke users who are affiliated with the same organization in a batch, I introduce *hierarchical identity structure*. If an organization is revoked, then all the affiliated users will be revoked.

4.2 System and Security Model

4.2.1 Identity Format

In a database directory based on the X.500 standard, the following rules Harris and Maymi (2016) are used for object organizations:

- The directory has a tree structure to organize the entries using a parent-child configuration.
- Each entry has a unique name made up of attributes of a specific object.
- The attributes used in the directory are dictated by the defined schema.
- The unique identifiers are called distinguished names.

To illustrate how to encode user identities in the proposed scheme based on the aforementioned database directory, I present an example directory in Figure 4.1 of a security company. The non-leaf nodes represent organizations, among which the root node represents the trusted authority. The leaf nodes represent users. Each organization (and user) has a unique identity under the parent organization, which is called local identity (LID), and meantime a unique global identity (denoted by ID) within the whole system. Assume that the hierarchical identity structure tree has $H + 1$ layers (the root node is on the 0^{th} layer), then the organizations and users' identities can be constructed according to the following syntax:

$$0\text{-}ID := ID \text{ of the root trusted authority,}$$
$$i\text{-}ID := \text{parent } (i-1)\text{-}ID \parallel i\text{-}LID,\ (1 \leq i \leq H)$$

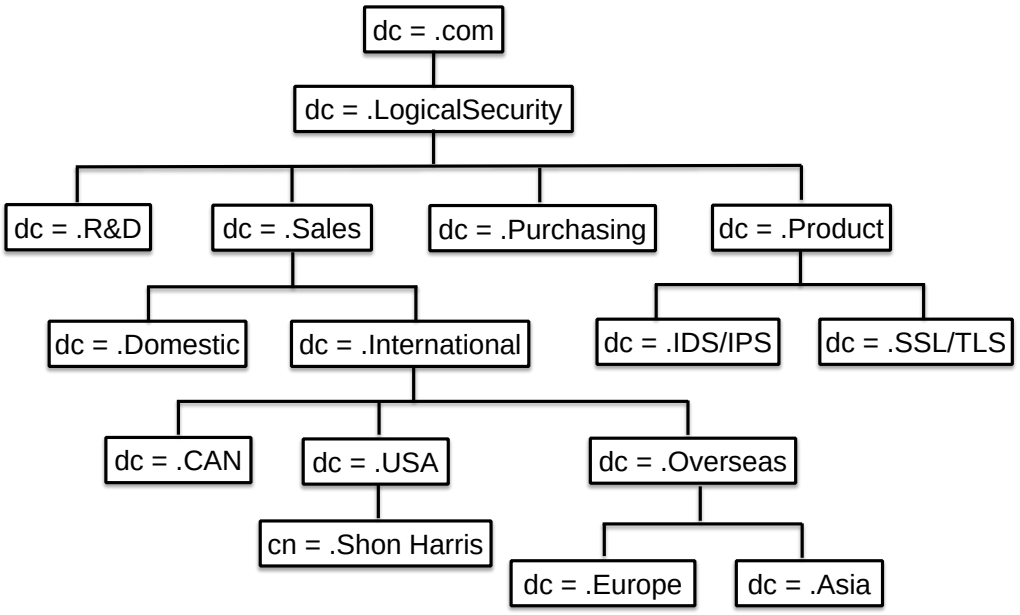

Figure 4.1: An Example of Organizational Structure Harris and Maymi (2016).

Take the hierarchical identities of *LocalSecurity* in Figure. 4.1 as an example. *dc=.LocalSecurity* is the trusted authority. There exist several departments under her administration, such as *dc=.R&D*, *dc=.Sales*, *etc.* An employee *cn=.Shon Harris* works in a fourth-level department with the hierarchical identity (*i.e.*, distinguished name) "dc=.LocalSecurity"$\|$"dc=.Sales"$\|$"dc=.International"$\|$"dc= .USA" $\|$"cn=.Shon Harris". For a user on the i^{th} layer, I also define:

$$ID_{|h} := \text{ancestor } h\text{-}ID,$$

where $h \in [1, i-1]$ and "ancestor h-ID" denotes the identity of the ancestor node on the h^{th} layer from root node to the user node. For the user Shon Harris,

$$ID_{|1} = \text{"dc=.LocalSecurity"} \| \text{"dc=.Sales"}$$

$$ID_{|2} = \text{"dc=.LocalSecurity"} \| \text{"dc=.Sales"} \| \text{"dc=.International"}$$

$$ID_{|3} = \text{"dc=.LocalSecurity"} \| \text{"dc=.Sales"} \| \text{"dc=.International"} \| \text{"dc=.USA"}$$

Both the user identity and the ancestors' identity will be embedded into the user's private key. In the hierarchical identity structure tree, users can be on any layer of the tree (except for the 0^{th} layer). With the proposed identity-based revocation mechanism and the hierarchical identity structure, on each layer, both individual users and organizations can be revoked. Furthermore, if an organization is revoked, then all the affiliated users will be revoked, thus achieving the goal of revocation in a batch.

4.2.2 System Model

I consider an access control system for cloud storage service as presented in Figure 4.2. There exist four entities in the system: Data owners, Cloud Server, Trusted Authority (TA) and Data Consumers (users).

The Data Owner chooses access policies and identities of undesired data users and encrypts the data under the policies and set of revoked identities before outsourcing them to the Cloud Server. A hybrid encryption approach will be adopted. The ABE scheme is used to encrypt a data encrypting key (DEK), and then the data is encrypted by the DEK with a symmetric encryption scheme. In Figure. 4.2, the DEK is the symmetric encryption key K. The Data Owner will upload the ABE ciphertext of K and the symmetric encryption ciphertext of the data M. In application scenarios where the data size is large, the Data Owner needs to divide the data into several blocks/components according to the logic granularities and then encrypt each block with symmetric encryption key(s). Therefore the format of the data stored on the cloud server might be in either format shown in Figure. 4.3.

The cloud server provides data storage service to data owners and data access service to users. Instead of by engaging the cloud server in the data access control,

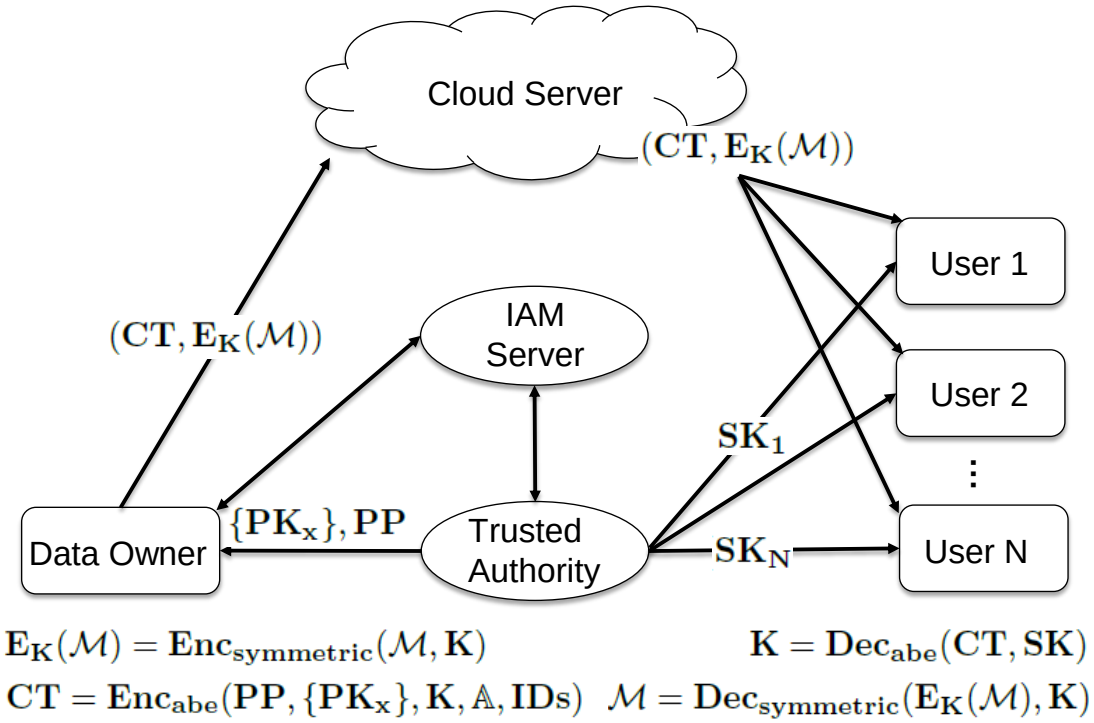

Figure 4.2: System Model of Access Control in Cloud Storage.

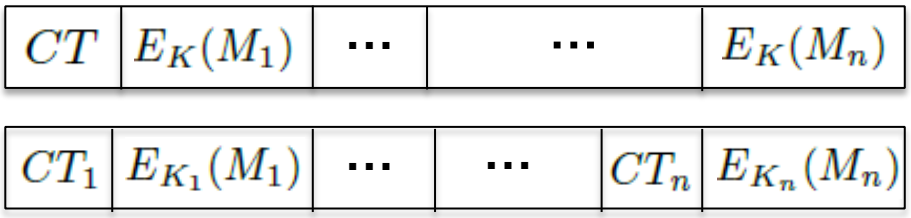

Figure 4.3: Data Format on the Cloud Server.

with ABE solutions the access checking is done "inside the cryptography". In this way, only data users whose attributes satisfy the access policy and are not revoked can decrypt the ciphertext.

The IAM server stores the organization's directory and provides directory services to both the Data Owner and the Trusted Authority. During encryption, the Data Owner searches the undesired users' full structural identities. The IAM might also provide the service of finding whether and how to implement affiliated revocation to

the Data Owner. Whenever a new user joins the organization, the TA asks for the information about the user from the IAM server and based on the obtained attributes and identity information to generate the secret key.

The TA as shown in Figure. 4.2, is the root node in the organizational structure. She is the root of trust in the whole organization and is responsible for entitling attributes, identities, and assigning private keys to users when there are entitled attributes and identities.

The data user is entitled a set of attributes and a unique ID according to his/her role or identity in the organizational structure. Because of various reasons such as interests conflicts, the data owners might revoke some of the users even if their attributes satisfy the access policy. Therefore, the user can decrypt the ciphertext only when he/she has eligible attributes and is not in the revoked set associated with the ciphertext.

4.2.3 Framework of Data Access Control Scheme

The framework of hierarchical identity revocable ABE scheme (DUR-CP-ABE) is defined as follows.

Definition 4.1. (CP-ABE WITH DISCRETIONARY USER REVOCATION). A DUR-CP-ABE scheme consists of the following algorithms: Setup, KeyGen, Encrypt and Decrypt where Encrypt is used for enforcing attribute-based access control and revoking undesired users.

- **Setup**$(\lambda, \mathbf{U}) \rightarrow \big(MSK, PP, \{PK_x\}\big)$: This algorithm is run by the Trusted Authority. It takes as inputs the security parameter λ and the attribute universe $\mathbf{U}$ and outputs master secret key MSK, public parameters PP, and the set of all the public attribute keys $\{PK_x\}$.

- **KeyGen**$(MSK, ID, \mathbf{S}) \rightarrow SK$: The private key generation algorithm is run by the Trusted Authority. It takes as inputs the master secret key MSK, a user's hierarchically structured identity ID, and a set of attributes $\mathbf{S}$ that describes the user's access privilege. It outputs the user's secret key SK.

- **Encrypt**$(PP, \{PK_x\}, M, IDs, \mathbb{A}) \rightarrow CT$: The data encryption algorithm is run by the data owner. It takes as inputs the public parameters PP, the set of public attribute key $\{PK_x\}$, a message M, the set IDs of revoked identities. It outputs a ciphertext CT.

- **Decrypt**$(CT, SK) \rightarrow M$ or $\perp$: The data decryption algorithm is run by the user. It takes as inputs the ciphertext CT and the private key SK. CT is associated with an access policy $\mathbb{A}$ and a set of revoked identities denoted by IDs. It outputs the message M if the attributes of the secret key holder with identity ID satisfy $\mathbb{A}$ and $ID \notin IDs$.

Consistency Constraint: Given that SK is the private key generated by **KeyGen** when it takes inputs of an identity ID and an attribute set $\mathbf{S}$; CT is the ciphtertext generated by **Encrypt** when it takes inputs of a revoked identity set IDs and an access structure $\mathbb{A}$. The DUR-CP-ABE scheme should satisfy the following consistency constraint:

$$\forall M : \textbf{Decrypt}(CT, SK) = M, \text{ if } ID \notin IDs \text{ and } \mathbf{S} \in \mathbb{A}$$

$$\textbf{AND}$$

$$\textbf{Decrypt}(CT, SK) = \perp \text{ if } ID \in IDs \text{ or } \mathbf{S} \notin \mathbb{A}.$$

In particular, only if a user is not revoked and his/her attribute set $\mathbf{S}$ satisfies the access structure $\mathbb{A}$, can the decryption algorithm work correctly. Here $ID \notin IDs$ means the user's ID is not in the revoked identity set and meanwhile the user is

not under the administration of the organization(s) whose ID is included in the revoked identity set. Take the organizational structure in Figure. 4.1 as an example, if "dc=.LocalSecurity"$\|$"dc=.Sales"$\|$"dc=.International" $\in IDs$, *i.e.*, the domain component is revoked, then all the data users in this domain component will be revoked as well. Therefore, the DUR-CP-ABE scheme not only supports individual user revocation but also supports affiliation-based revocation.

4.2.4 Security Model

I make the following security assumptions of cloud storage systems. First, the cloud server is honest but curious. The cloud server honestly follows the designated protocol, but curiously infers additional sensitive information based on the data available to him. Active attacks such as deleting or tampering with the stored data are out of scope in this thesis. Second, the cloud server might provide data access permission to unauthorized users either on purpose for more benefits or because of data leakage events. Third, the users are dishonest and may collude together in order to gain access privileges that they individually do not have. There are three categories of collusion. First attribute collusion: two non-revoked users A and B with attribute set $S_A = \{A_1, A_2\}$ and $S_B = \{A_3, A_4\}$ might collude in order to decrypt a ciphertext under access policy $A_1 \& A_4$. Second, a revoked user A whose attributes satisfy the access policy and a non-revoked user B whose attributes do not satisfy the access policy might collude together to try to gain data access permission. Third, two revoked users A and B whose attributes both satisfy the access policy might collude together to gain higher access permission.

To resist against the attacks above, I describe the security model for the DUR-CP-ABE system by the game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ as follows.

The DUR-CP-ABE security model is formalized by the game between a challenger and an adversary $\mathcal{A}$ as follows.

- **Init**: The adversary $\mathcal{A}$ commits to a challenge access structure $\mathbb{A}^*$ and a revoked identity set IDs^* and sends them to the challenger.
- **Setup**: The challenger runs the setup algorithm. The generated master secret key MSK is kept secret and the public parameters PP and the set of public attributes key $\{PK_x\}$ are given to the adversary.
- **Phase1**: The adversary $\mathcal{A}$ makes repeated private key queries $(\mathbf{S_i}, ID_i)_{i\in[1,q_1]}$ with two constrains: (1) if $\mathbf{S}_i \in \mathbb{A}^*$, then $ID_i \in IDs^*$; (2) if $ID_i \notin IDs^*$, then $\mathbf{S}_i \notin \mathbb{A}^*$.
- **Challenge**: The adversary sends to the challenger two randomly selected equal length messages M_0 and M_1. The challenger picks up a random bit $b \in \{0, 1\}$, and encrypts M_b under the access structure $\mathbb{A}^*$ and the revoked identity set IDs^*. The generated challenge ciphertext CT^* is sent back to the adversary $\mathcal{A}$.
- **Phase2**: Repeat **Phase1** with the same constrains.
- **Guess**: The adversary outputs a guess bit b' of b.

Definition 4.2. Define $\mathrm{Adv}_{\mathcal{A}} = |\Pr[b' = b] - \frac{1}{2}|$ as the advantage of the adversary $\mathcal{A}$ winning the game above. The DUR-CP-ABE scheme is secure if $\mathrm{Adv}_{\mathcal{A}}$ of any PPT adversary $\mathcal{A}$ is a negligible function [1] of the security parameter.

[1] A function $\mu(x) : \mathbb{N} \to \mathbb{R}$ is negligible. If for every positive polynomial $poly(\cdot)$ there exists an integer $N_{poly} > 0$ such that for all $x > N_{poly}$, $|\mu(x)| < \frac{1}{poly(x)}$, reference link: https://en.wikipedia.org/wiki/Negligible_function

4.3 Attribute-Based Access Control with Discretionary Revocation

In this section, I first present an overview of the proposed revocation approach and then provide a detailed construction.

4.3.1 Overview

I used the idea of revocation by redundant equations. Let the encryption algorithm define several "local" revocation equations. Use a "two equation" method for decryption. Intuitively, when decrypting, a user ID will apply his/her secret key to the ciphertext. If $ID \notin IDs$, he/she will get two independent equations and be able to extract the randomness used to mask the message. However, if $ID \in IDs$ (*i.e.*, he is revoked), then he will only get two dependent equations of a two variable formula and thus be unable to extract the randomness. Alternatively, the ciphertext cand be regarded as locally defining a degree one polynomial. A user ID will get two points on a fresh degree one polynomial if $ID \notin IDs$ (and otherwise the user will essentially only get one point on the polynomial, which is not enough to solve). This could be considered as a local revocation of each user to a ciphertext.

To resist against collusion attacks described in the security model, the key shares are randomized or "personalized" to each user to prevent combination of decryption shares. The cancelation techniques based on the power of a bilinear map is utilized in the construction.

4.3.2 Proposed Access Control System Construction

Let $\mathbb{G}$ be a bilinear group of prime order p, and let g be the generator of $\mathbb{G}$. All the string-format identities can be encoded as an element in $\mathbb{Z}_p$ through a hash function $\{0,1\}^* \to \mathbb{Z}_p$. The proposed system consists of the following four components.

Setting Up the System: The TA initializes the system by running the Setup algorithm. In particular, it chooses random exponents $\alpha, b \in \mathbb{Z}_p$ as the master secret key $MSK = \{\alpha, b\}$. Then it generates the public parameters as follows.

$$PP = \left(g, g^b, g^{b^2}, e(g,g)^\alpha\right)$$

For each attribute $x \in \mathbf{U}$, the TA generates a random group element $h_{xh} \in \mathbb{G}$ for each layer of the organizational trees structure. The following public attribute keys PK_x are generated.

$$PK_x = \{h_{xh}^b\}_{h\in[1,H]}$$

Generating Secret Key for Users: When a new user joins the system, the TA will assign it a set of attributes based on its role or identity. Based on the assigned attributes and identity in the organizational structure, the TA then generates secret keys for the user by running the KeyGen algorithm. It takes as inputs the master secret key MSK, the set of attributes $\mathbf{S}$ that describes the user's ID. Assume the user's ID is on the H'^{th} layer ($1 \leq H' \leq H$). It chooses a random $t \in \mathbb{Z}_p$ and generates the user's secret key in the following format and sends it to the user in a secure way.

$$SK = (K = g^\alpha g^{b^2 t}, K_x, L = g^{-t}), \text{ where}$$

$$K_x = \{K_{xh} = (g^{b\cdot ID_{|h}} h_{xh})^t\}_{\forall x\in\mathbf{S}, h\in[1,H],\ ID_{|h}=ID\ (h\in[H',H])}$$

Encrypting Data: The data owner will process the data to be outsourced with a hybrid encryption method as described in the system model. Therefore, the encrypted message of the access control system will be a data encryption key $\mathcal{K}$. The Encrypt algorithm works as follows. It takes as inputs the public parameters PP, the set of public attribute keys $\{PK_x\}$, a data encryption key $\mathcal{K}$, an access policy $\mathbb{A}$ that can be denoted by an LSSS access structure (M, ρ) and the revoked identity set $IDs = \{ID_1, \cdots, ID_r\}$ constructed by querying the IAM server or on its own (*e.g.*, revoke only one user whose ID is known). M is an $\ell \times n$ share-generating matrix

where ℓ denotes the number of attributes involved in the encryption. ρ is a function associating rows of M to attributes. ID_j ($j \in [1, r]$) is on the H_j^{th} layer in the organizational structure.

It chooses a random vector $v = (s, y_2, \cdots, y_n) \in \mathbb{Z}_p^n$ and for $k \in [1, l]$ calculates $\lambda_k = v \cdot M_k$ where M_k denotes the vector corresponding to the k-th row in the matrix M. It chooses random $\mu_1, ..., \mu_r \in \mathbb{Z}_p$ such that $\mu = \mu_1 + ... + \mu_r$. The ciphertext of the message $\mathcal{K}$ in the following format is uploaded onto the cloud server by the data owner.

For revocation of a single user, the ciphertext is:

$$CT = (C, C', \hat{C}, (M, \rho), ID'), \text{ where}$$
$$C = \mathcal{M}e(g, g)^{\alpha s},$$
$$C' = g^s,$$
$$\hat{C} = \{\hat{C}_k = g^{b \cdot \lambda_k}, \hat{C}'_k = (g^{b^2 \cdot ID'} h^b_{\rho(k)H'})^{\lambda_k}\}_{k \in [1,l]}$$

For revocation of multiple users, the ciphertext is:

$$CT = (C, C', \hat{C}, (M, \rho), \mathbf{IDs}), \text{ where}$$
$$C = \mathcal{K}e(g, g)^{\alpha s \mu},$$
$$C' = g^{s\mu},$$
$$\hat{C} = \{\hat{C}_{k,j} = g^{b \cdot \lambda_k \mu_j}, \hat{C}'_{k,j} = (g^{b^2 \cdot ID_j} h^b_{\rho(k)H_j})^{\lambda_k \mu_j}\}^{j \in [1,r]}_{k \in [1,l]}$$

Decrypting Data: The data user firstly downloads the encrypted data from the cloud server and then runs the Decrypt algorithm of the ABE scheme to obtain the data encryption keys and decrypts the data blocks with these DEKs. Here is how the Decrypt algorithm works. It takes as inputs CT which is the input ciphertext with an access structure (M, ρ) and a revoked identity ID_j and secret key SK for a set of attributes $\mathbf{S}$ and the identity ID. Suppose that $\mathbf{S}$ satisfies the access structure and let $\mathbf{I} \subset [1, l]$ be defined as $\mathbf{I} = \{i : \rho(i) \in \mathbf{S}\}$. Let $\{\omega_i \in \mathbb{Z}_p\}_{i \in \mathbf{I}}$ be a set of constants such

that if $\{\lambda_i\}_{i\in\mathbf{I}}$ are valid shares of any secret s according to M, then $\Sigma_{i\in\mathbf{I}}\omega_i\lambda_i = s$. If the condition $ID \notin \mathbf{IDs}$ holds, calculate A as follows. The value $e(g,g)^{\alpha s\mu}$ could be obtained by evaluating $\frac{e(C',K)}{A}$. The decryption algorithm then divides out this value from the ciphertext component C and obtains the message $\mathcal{K}$.

$$A = \prod_{j=1}^{r}\prod_{i\in\mathbf{I}}[e(K_{\rho(i)H_j}, \hat{C}_{k,j}) \cdot e(L, \hat{C}'_{k,j})]^{\frac{\omega_i}{ID_{|H_j} - ID_j}}$$

For a revocation set $\mathbf{IDs} = \{ID_1, \cdots, ID_r\}$, Encrypt creates an exponent $\mu \in \mathbb{Z}_p$ and splits it into r random shares $\mu_1, \cdots, \mu_r$. For each share, the ciphertext has two components $\hat{C}_{k,j}$ and $\hat{C}'_{k,j}$. If $ID_{|H_j} = ID_j$, it will get two linearly dependent equations and the exponent $\mathbf{b^2 t}\lambda_\mathbf{i}\mu_\mathbf{j}(ID_{|\mathbf{H_j}} - ID_\mathbf{j})$ will be $\mathbf{0}$, thus unable to solve the system; otherwise it gets $e(g,g)^{b^2 st\mu}$. If ID_j is a domain component's identity, then the component K_{xH_j} in the user secret key will not work, thus achieving affiliated revocation.

The decryption for ciphertext revoking a single user and multiple users are as follows respectively.

$$\begin{aligned}
A &= \textstyle\prod_{i\in\mathbf{I}}[e(K_{\rho(i)H'}, \hat{C}_i) \cdot e(L, \hat{C}'_i)]^{\frac{\omega_i}{ID-ID'}} \\
&= (\textstyle\prod_{i\in\mathbf{I}}[e((g^{b\cdot ID}h_{\rho(i)H'})^t, g^{b\cdot\lambda_i}) \\
&\quad \cdot e(g^{-t}, (g^{b^2\cdot ID'}h^b_{\rho(i)H'})^{\lambda_i})]^{\frac{\omega_i}{ID-ID'}} \\
&= (\textstyle\prod_{i\in\mathbf{I}}[e(g^{b\cdot ID\cdot t}, g^{b\cdot\lambda_i}) \cdot e(h^t_{\rho(i)H'}, g^{b\cdot\lambda_i}) \\
&\quad \cdot e(g^{-t}, g^{b^2\cdot ID'\cdot\lambda_i}) \cdot e(g^{-t}, h^{b\cdot\lambda_i}_{\rho(i)H'})]^{\frac{\omega_i}{ID-ID'}} \\
&= (\textstyle\prod_{i\in I}[e(g^{b\cdot ID\cdot t}, g^{b\cdot\lambda_i}) \cdot e(g^{-t}, g^{b^2\cdot ID'\cdot\lambda_i})]^{\omega_i})^{1/(ID-ID')} \\
&= (\textstyle\prod_{i\in\mathbf{I}}[e(g,g)^{b^2 t\lambda_i(ID-ID')}]^{\frac{\omega_i}{ID-ID'}} \\
&= \textstyle\prod_{i\in\mathbf{I}} e(g,g)^{b^2 t\lambda_i\omega_i} \\
&= e(g,g)^{b^2 t\sum_{i\in\mathbf{I}}\lambda_i\omega_i} \\
&= e(g,g)^{b^2 ts}
\end{aligned}$$

$$
\begin{aligned}
A &= \prod_{j=1}^{r}(\prod_{i\in\mathbf{I}}[e((g^{b\cdot ID_{|H_j}}h_{\rho(i)H_j})^t, g^{b\cdot\lambda_i\mu_j}) \\
&\quad \cdot e(g^{-t}, (g^{b^2\cdot ID'_j}h^b_{\rho(i)H_j})^{\lambda_i\mu_j})]^{\frac{\omega_i}{ID_{|H_j}-ID_j}} \\
&= \prod_{j=1}^{r}(\prod_{i\in\mathbf{I}}[e(g,g)^{\mathbf{b^2t\lambda_i\mu_j}(ID_{|\mathbf{H_j}}-ID_{\mathbf{j}})}]^{\frac{\omega_i}{ID_{|H_j}-ID_j}} \\
&= \prod_{j=1}^{r}(\prod_{i\in\mathbf{I}} e(g,g)^{b^2t\lambda_i\omega_i\mu_j}) \\
&= \prod_{j=1}^{r} e(g,g)^{b^2t\mu_j\sum_{i\in\mathbf{I}}\lambda_i\omega_i} \\
&= \prod_{j=1}^{r} e(g,g)^{b^2t\mu_j s} \\
&= e(g,g)^{b^2st\sum_{j=1}^{r}\mu_j} \\
&= e(g,g)^{b^2st\mu}.
\end{aligned}
$$

4.4 Analysis and Evaluation

4.4.1 Security Analysis

M-q-parallel-BDHE. The definition of the modified (decisional) q parallel Bilinear Diffie-Hellman Exponent problem is as follows. Choose a group $\mathbb{G}$ of prime order p, a random generator g of $\mathbb{G}$ and random $a, s, b_1, b_2, \cdots, b_q \in \mathbb{Z}_p$. Given

$$
\begin{aligned}
\mathbf{y} =&\{g, g^s, g^a, \cdots, g^{(a^q)}, , g^{(a^{q+2})}, \cdots, g^{(a^{2q})}, \\
&\forall_{1\le i\le q}\ g^{a/b_i}, \cdots, g^{a^q/b_i}, , g^{a^{q+2}/b_i}, ..., g^{a^{2q}/b_i}, \\
&\forall_{1\le j\le q}\ g^{a\cdot s/b_j}, \cdots, g^{a^q\cdot s/b_j}\},
\end{aligned}
$$

it is hard for a probabilistic polynomial time (PPT) adversary to distinguish $e(g,g)^{a^{q+1}s}$ from a random element $R \in \mathbb{G}_T$. An algorithm $\mathcal{B}$ that outputs $z \in \{0,1\}$ has advantage ϵ in solving the M-q-parallel-BDHE problem if the follwing equation holds.

$$
|\Pr[\mathcal{B}(\mathbf{y}, T = e(g,g)^{a^{q+1}s}) = 0] - \Pr[\mathcal{B}(\mathbf{y}, T = R) = 0]| \ge \epsilon.
$$

*The **M-q-parallel-BDHE assumption** holds if the advantage ϵ of any PPT adversary $\mathcal{B}$ to solve the M-q-parallel-BDHE problem is a negligible function of the security parameter.*

Theorem 4.1. *The Modified (decisional) q parallel Bilinear Diffie-Hellman Exponent assumption generically holds.*

Proof. Using the terminology from BBG Waters (2011) "$f = a^{q+1}s$ is independent of the polynomials P and Q" is needed to be proven. I set $Q = \{1\}$ since all given terms are in the bilinear group and

$$P = \{1, s, \forall_{i\in[1,2q],j\in[1,q],i\neq q+1} a^i, a^i/b_j, a^i \cdot s/b_j\}.$$

Choose a generator u. Set $g = u^{\prod_{j\in[1,q]} b_j}$. All the above terms are substituted by a set of polynomials with the maximum degree $3q + 1$. Now, check whether f is symbolically independent of any two polynomials in P and Q. To realize f from P and Q, a term of the form $a^{m+1}s$ is needed. Whereas, no such terms can be realized from the product of any two polynomials $p, p' \in P$. To form such a term, a polynomial with a single factor of s is needed. If s is used as p, then p' has to be a^{q+1} which doesn't exist in P. If $p = a^i \cdot s/b_j$, there always exists b_j, which cannot be canceled. Hence it can be concluded that the M-q-parallel-BDHE assumption is generically secure. □

The security of the proposed scheme can be concluded by the following Theorems.

Theorem 4.2. *Suppose that the M-q-parallel-BDHE assumption holds. Then no PPT adversary can selectively break the DUR-CP-ABE scheme with a challenge access structure* $\left(\mathrm{M}^*, \rho^*\right)$*, where the size of* M^* *is* $\ell^* \times n^*$ *and* $\ell^*, n^* \leq q$*.*

Proof Sketch: The basic idea of the proof is using the reduction technology as shown Figure. 4.4, where a simulator is constructed to simulate a DUR-CP-ABE game for the attacker by answering its queries and programming the challenge access structure together with the revoked identity set into the public parameters and the set of public attribute keys.

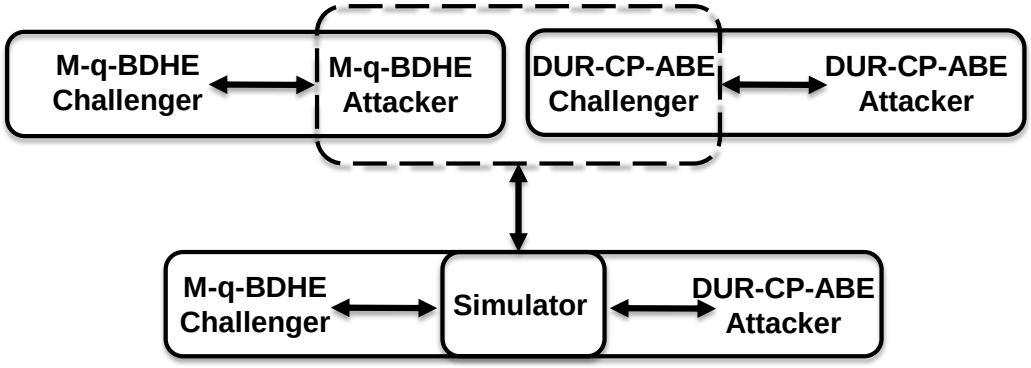

Figure 4.4: Process of Reduction to the M-q-BDHE Problem.

I present the proof for the DUR-CP-ABE scheme with only one revoked identity. The proof of the scheme with multiple revoked identities can be obtained by adapting the following proof easily.

Proof. **Init** The simulator takes in an M-q-BDHE challenge $\{\mathbf{y}, T\}$. Then the adversary declares the revoked identity ID^* and gives the simulator the challenge access structure $\mathcal{A}^*$ that is described by (M^*, ρ^*), where M^* has n^* (less than q) columns. Let the challenge matrix $M^* = (\overrightarrow{M_1^*}, \cdots, \overrightarrow{M_l^*})^T$, where each row vector $\overrightarrow{M_i^*} = (M_{i,1}^*, \cdots, M_{i,n}^*)$ for $1 \leq i \leq l$.

Setup The simulator chooses a random value α' and lets $e(g,g)^{\alpha} = e(g,g)^{\alpha'} e(g^a, g^{a^q})$ to implicitly set $\alpha = \alpha' + a^{q+1}$. Moreover, let $g^b = g^a$, $g^{b^2} = (g^{a^2})$ to implicitly set $b = a$.

To embed ID^* and (M^*, ρ^*) in $\{h_{xi}^b\}_{x \in \mathbf{U}, i \in [1,H]}$, I regard the challenge matrix M^* as a row vector set and divide it into three subsets $M^{*\prime}$, $M^{*\prime\prime}$ and $M^{*\prime\prime\prime}$ such that $M^{*\prime} \cup M^{*\prime\prime} \cup M^{*\prime\prime\prime} = M^*$ and $M^{*\prime} \cap M^{*\prime\prime} \cap M^{*\prime\prime\prime} = \emptyset$. Specifically, $M^{*\prime}$, $M^{*\prime\prime}$ and $M^{*\prime\prime\prime}$ are initially set to be empty set. Define the n-dimension vectors $\overrightarrow{e} = (1, 0, ..., 0)$ and $\overrightarrow{\mu} = (a^2, a^3, ..., a^{n+1})$. For $i = 1$ to l, if $\overrightarrow{M_i^*}$ is linearly independent on $M^{*\prime}$ and $\overrightarrow{e}$ cannot be linearly expressed by $M^{*\prime} \cup \{\overrightarrow{M_i^*}\}$, then I merge $\overrightarrow{M_i^*}$ into $M^{*\prime}$; if $\overrightarrow{M_i^*}$ is

linearly independent on $M^{*\prime}$ and $\overrightarrow{e}$ can be linearly expressed by $M^{*\prime} \cup \{\overrightarrow{M_i^*}\}$, then I merge $\overrightarrow{M_i^*}$ into $M^{*\prime\prime\prime}$; if $\overrightarrow{M_i^*}$ is linearly dependent on $M^{*\prime}$, then I merge $\overrightarrow{M_i^*}$ into $M^{*\prime\prime}$. As a result, $M^{*\prime}$ is a linear independent vector group while each vector in $M^{*\prime\prime}$ can be linearly expressed by $M^{*\prime}$. Although $\overrightarrow{e}$ cannot be spanned by $M^{*\prime}$, it can be linearly expressed by $M^{*\prime}$ merged with each vector in $M^{*\prime\prime\prime}$. Therefore, each vector in M can be linearly expressed by $M^{*\prime} \cup \{\overrightarrow{e}\}$.

Next, I describe how the simulator "programs" $\{h_{xi}^b\}_{x\in\mathbf{U},i\in[1,H]}$. X denotes the set of indices i so that $\rho^*(i) = x$. Assume there are m vectors in $M^{*\prime}$ and $M^{*\prime} = (\overrightarrow{M_1^{*\prime}}, \cdots, \overrightarrow{M_m^{*\prime}})^T$. For $i \in X$, its row vector $\overrightarrow{M_i^*}$ can be written as $\varepsilon_{i0}\overrightarrow{e} + \varepsilon_{i1}\overrightarrow{M_1^{*\prime}} + \cdots + \varepsilon_{im}\overrightarrow{M_m^{*\prime}}$, where $(\varepsilon_{i0}, \varepsilon_{i1}, \cdots, \varepsilon_{im}) \in Z_p^m$. For each $\overrightarrow{M_i^*}$, define a vector $\overrightarrow{M_i^{**}}$, where $\overrightarrow{M_i^{**}} = \varepsilon_{i1}\overrightarrow{M_1^{*\prime}} + \cdots + \varepsilon_{im}\overrightarrow{M_m^{*\prime}}$. As a result, I get a new vector group $M^{**} = (\overrightarrow{M_1^{**}}, \cdots, \overrightarrow{M_l^{**}})$ and each $\overrightarrow{M_i^{**}}$ is in the span of $M^{*\prime}$. By choosing a random value z_{xi}, the simulator programs h_{xi} and h_{xi}^b as follows:

$$\begin{aligned} h_{xh} &= g^{z_{xh}} g^{-aID^*_{|h}} \textstyle\prod_{i\in X} g^{(\varepsilon_{i1}\overrightarrow{M_1^{*\prime}} + , \cdots, + \varepsilon_{im}\overrightarrow{M_m^{*\prime}})\cdot\overrightarrow{\mu}/b_i} \\ h_{xh}^b &= g^{z_{xh}} g^{-a^2 ID^*_{|h}} (\textstyle\prod_{i\in X}\prod_{j=1}^{n} g^{M^{**}_{i,j} a^{j+1}/b_i}). \end{aligned}$$

If X is an empty set, I set $h_{xh}^b = g^{z_{xh}}$. Then the simulator publishes the above parameters $(g, g^b, g^{b^2}, \{h_{xh}^b\}_{x\in\mathbf{U},h\in[1,H]}, e(g,g)^\alpha)$ as the public parameters.

Phase I For a query (S, ID), the simulator constructs the private key as follows. Since each $\overrightarrow{M_i^{**}}$ is in the span of $M^{*\prime}$ while $\overrightarrow{e}$ is not in the span of $M^{*\prime}$, I can still find a vector $\overrightarrow{\omega}$ with $\omega_1 = -1$ and $\overrightarrow{\omega}\cdot\overrightarrow{M_i^{**}} = 0$, where $1 \le i \le m$.

Therefore, the simulator selects a random value r and calculates the private key L as

$$L = g^{r+\overrightarrow{\omega}\cdot\overrightarrow{\nu}} = g^r \prod_{i=1,\cdots,n^*} (g^{a^{q-i}})^{\omega_i},$$

which implicitly sets the randomness t as

$$t = r + \overrightarrow{\omega}\cdot\overrightarrow{\nu} = r + \omega_1 a^{q-1} + \omega_2 a^{q-2} + , \cdots, + \omega_n a^{q-n^*},$$

where $\overrightarrow{v} = (a^{q-1}, a^{q-2}, \cdots, a^{q-n^*+2})$. Since $g^{a^2 t}$ contains a term of $g^{-a^{q+1}}$ the unknown term in g^{α} can be canceled when creating the K component in the private key. The simulator constructs K as follows.

$$K = g^{\alpha'} g^{a^2 r} \prod_{i=0,\cdots,n-2} (g^{a^{q+i}})^{\omega_i}.$$

For $\forall x \in \mathbf{S}$, if there is no i such that $\rho^*(i) = x$, the simulator simply sets $K_{xh} = L^{z_{xh}}$. For those used in the challenge access structure, it must be ensured that there are no terms of the form g^{a^{q+1}/b_i} that the simulator cannot simulate. Since $\overrightarrow{w} \cdot M_i^{**\prime} = 0$, all of these terms can be canceled. Define X as the set of all i such that $\rho^*(i) = x$, the simulator creates K_{xh} as follows.

$$K_{xh} = (g^{z_{xh}} g^{a(ID_{|h} - ID^*_{|h})} \prod_{i \in X} g^{\overrightarrow{M_i^{**}} \cdot \overrightarrow{\mu}/b_i})^{(r + \overrightarrow{w} \cdot \overrightarrow{v})}$$

Challenge In this phase, the adversary provides to the simulator two challenge messages $\mathcal{M}_0$, $\mathcal{M}_1$ with the challenge matrix M of dimension at most n^* columns.

First, The simulator flips a coin β and creates the ciphertext component $C = \mathcal{M}_\beta T \cdot e(g^s, g^{\alpha'})$, $C' = g^s$. Then the simulator chooses random value $y'_2, y'_3, \cdots, y'_n$ and share the secret s using the vector

$$\overrightarrow{v} = (s, y'_2, y'_3, \cdots, y'_n).$$

Next, it calculates

$$\lambda_k = \overrightarrow{v} \cdot (\varepsilon_{k0} \overrightarrow{e} + \varepsilon_{k1} \overrightarrow{M_1^{*\prime}} + \varepsilon_{k2} \overrightarrow{M_2^{*\prime}} + ... + \varepsilon_{km} \overrightarrow{M_m^{*\prime}})$$

And it generates the ciphertext component C_k^* as:

$$\hat{C}_k = g^{as(M_{k1}^{**} + \varepsilon_{k0})} \cdot \prod_{i=2}^{n} g^{M_{ki}^{**} y'_i}$$

For $k = 1, \cdots, n^*$, X_k is defined as the set of the index i in such that $\rho(i) = \rho(k)$. Finally, the simulator builds the ciphertext component C'_k as:

$$\hat{C}'_k = (g^{a^2 ID^*} g^{z_{xH'}} g^{-a^2 ID^*} \prod_{i \in X_k} g^{\overrightarrow{M_i^{**}} \cdot \overrightarrow{\mu}/b_i})^{\lambda_k}$$

Phase II Same as **Phase I**.

Guess The adversary will eventually output a guess β' of β. The simulator then outputs 0 to guess that $T = e(g,g)^{sa^{q+1}}$ if $\beta' = \beta$; otherwise, it outputs 1 to indicate that it believes T is a random group element in $\mathbb{G}_T$. When T is a tuple the simulator $\mathcal{B}$ gives a perfect simulation so that

$$Pr[\mathcal{B}(\overrightarrow{X}, T = e(g,g)^{sa^{q+1}}) = 0] = \frac{1}{2} + Adv_{\mathcal{A}}.$$

When T is a random group element, the message M_β is completely hidden from the adversary and $Pr[\mathcal{B}(\overrightarrow{X}, T = R) = 0] = \frac{1}{2}$. Therefore, $\mathcal{B}$ can play the modified decisional q-parallels $BDHE$ game with non-negligible advantage. □

Theorem 4.3. *The DUR-CP-ABE scheme is resistant against unauthorized access.*

Proof. As discussed in the security model, there are two categories of unauthorized accesses: 1) one unauthorized user whose attributes do not satisfy the access policy or is revoked by the data owner; 2) two unauthorized colluding users.

The first scenario is stated directly in the the query constrains, *i.e.*, any PPT adversary who is not allowed to ask for a secret key with eligible attributes for the access policy $\mathbb{A}^*$ and non-revoked identity can guess correctly which message is encrypted with only a negligible probability.

For the second scenario, let's look into the detailed construction. The key shares for both attributes and revoked identities are "personalized" to each user to prevent combination of decryption shares. In particular, each user's secret key is randomized by an exponent t such that when decrypting each user recovers shares $t\lambda_i w_i \mu_j$ where $\lambda_i w_i$ corresponds to the attributes and μ_j associates with the revoked user ID_j. Therefore, the secret keys of two users can not work together to recover $ts\mu$ which is the key to successful decryption. □

4.4.2 Performance Analysis

In this section, I analyze the propsed construction in terms of computation, storage, and communication overhead. Since the scheme is constructed based on the CP-ABE scheme by Waters in Waters (2011) (denoted by W-CP-ABE), which itself and adapted constructions are broadly used Akinyele *et al.* (2011); Lai *et al.* (2013); Akinyele *et al.* (2013), I use this scheme as the baseline. To demonstrate how the size of the revoked identity set influences the overheads of the system, Let "O-DUR-CP-ABE" denote the scheme where $r = 1$ and "M-DUR-CP-ABE" denote the scheme where $r > 1$.

In the following sections, let m denote the number of attributes defined in the system; H is the number of layers in the organizational structure tree; r is the number of revoked identities; $\mathbf{S}$ indicates the set of attributes entitled to a user; l denotes the number of attributes involved in encryption; $|\mathbf{I}|$ is the number of attributes (subset of $\mathbf{S}$) used in decryption.

There are four types of time-consuming operations in all the schemes, *i.e.*, pairing, exponentiation, multiplication and inversion. According to Li *et al.* (2014), the pairing and exponentiation operations take the dominant computation costs. Therefore, I use the number of pairing and exponentiation operations as metrics for computation complexity.

Computation Complexity Analysis

Table 4.1 and Table 4.2 present computation costs comparisons of the three schemes. In the Setup algorithm of all these three schemes, there is only one pairing operation that is brought by evaluating $e(g, g)^{\alpha}$. In W-CP-ABE, the number of exponentiations is $m + 3$. In the other two schemes, there are $mH + 3$ exponentiation operations

Table 4.1: Computation Complexity Comparison in terms of the Number of Pairing Operations

Schemes	W-CP-ABE	O-DUR-CP-ABE	M-DUR-CP-ABE
Setup	1	1	1
KeyGen	0	0	0
Encrypt	0	0	0
Decrypt	$2\|\mathbf{I}\|+1$	$2\|\mathbf{I}\|+1$	$2\|\mathbf{I}\|r+1$

because of the organizational structure. In the KeyGen algorithm of W-CP-ABE, the number of exponentiations is $|S|+3$. In the two DUR-CP-ABE schemes, this number increases to $H|S|+H'+3$. The increment comes from the fact that all layers in a user's identity structure are embedded in the key component for each attribute.

For the Encrypt algorithm of W-CP-ABE, the number of exponentiation operations is $3l+2$. In O-DUR-CP-ABE, the number is $2l+3$. In M-DUR-CP-ABE, the number is $(2l+1)r+2$. In W-CP-ABE, the number of pairing needed for decryption is $2|\mathbf{I}|+1$, which is the same as that of O-DUR-CP-ABE. The number increases to $2|\mathbf{I}|r+1$ in M-DUR-CP-ABE. The number of exponentiations in W-CP-ABE, O-DUR-CP-ABE and M-DUR-CP-ABE is $|\mathbf{I}|$, $|\mathbf{I}|$, and $|\mathbf{I}|r$. Increased overhead in M-DUR-CP-ABE is due to multiple user revocation.

Storage and Communication Overhead Analysis

The main storage overheads come from the Setup algorithm and KeyGen algorithm. The communication overheads come from the ciphertext generated by the encryption algorithm. Table 4.3 and Table 4.4 summarize the storage and communication

Table 4.2: Computation Complexity Comparison in terms of the Number of Exponentiation Operations

Schemes	W-CP-ABE	O-DUR-CP-ABE	M-DUR-CP-ABE
Setup	$m+3$	$mH+3$	$mH+3$
KeyGen	$\|\mathbf{S}\|+3$	$H\|\mathbf{S}\|+H'+3$	$H\|\mathbf{S}\|+H'+3$
Encrypt	$3l+2$	$2l+3$	$(2l+1)r+2$
Decrypt	$\|\mathbf{I}\|$	$\|\mathbf{I}\|$	$\|\mathbf{I}\|r$

Table 4.3: Storage Overhead Comparison

Schemes	W-CP-ABE	O-DUR-CP-ABE	M-DUR-CP-ABE
Setup	$m+4$	$Hm+6$	$Hm+6$
KeyGen	$\|\mathbf{S}\|+2$	$H\|\mathbf{S}\|+3$	$H\|\mathbf{S}\|+3$

overhead of the three schemes.

The storage overhead in the Setup algorithm of W-CP-ABE is $m+4$. In the two DUR-CP-ABE schemes, it is $mH+6$ because of the public attribute keys generated for the organizational structure. In W-CP-ABE, the overhead of storing a private key is $|\mathbf{S}|+2$. In the two DUR-CP-ABE schemes, the private key storage overhead increases to $H|\mathbf{S}|+3$. The ciphertext size of W-CP-ABE, O-DUR-CP-ABE and M-DUR-CP-ABE is $2l+2$, $2l+2$, and $2lr+2$ respectively.

Table 4.4: Communication Overhead Comparison

Schemes	W-CP-ABE	O-DUR-CP-ABE	M-DUR-CP-ABE
Encrypt	$2l + 2$	$2l + 2$	$2lr + 2$

4.4.3 Implementation and Testing Results

The proposed scheme are implemented in C using PBC library Lynn (2006) on Ubuntu 14.04. All of the results are obtained by running the programs ten times. To evaluate the relations between the number of attributes and the computation overhead, I set r to 1 and H to 2, *i.e.*, only one identity is revoked and the height of the organizational tree is 3.

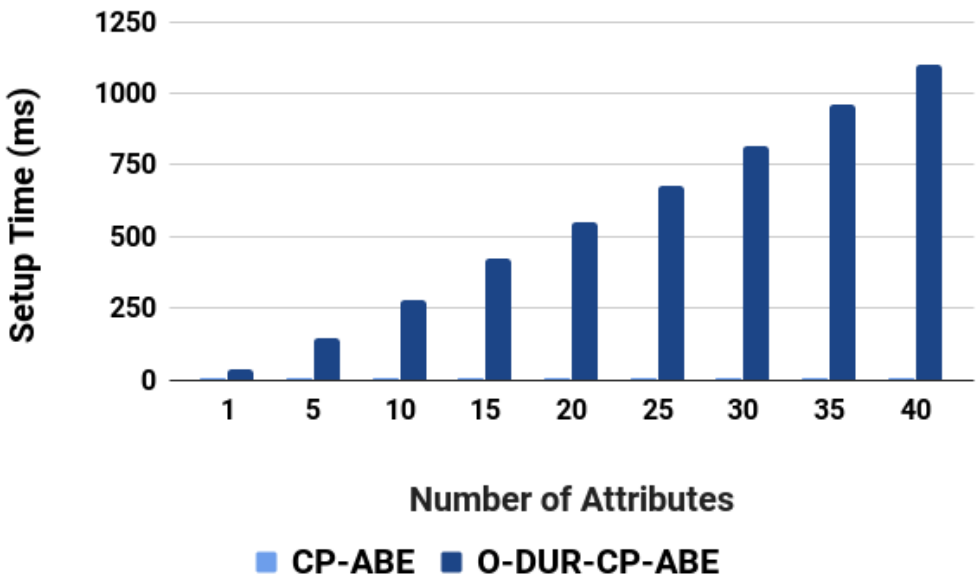

Figure 4.5: Relations between the Number of Attributes and Time Consumption for Setup.

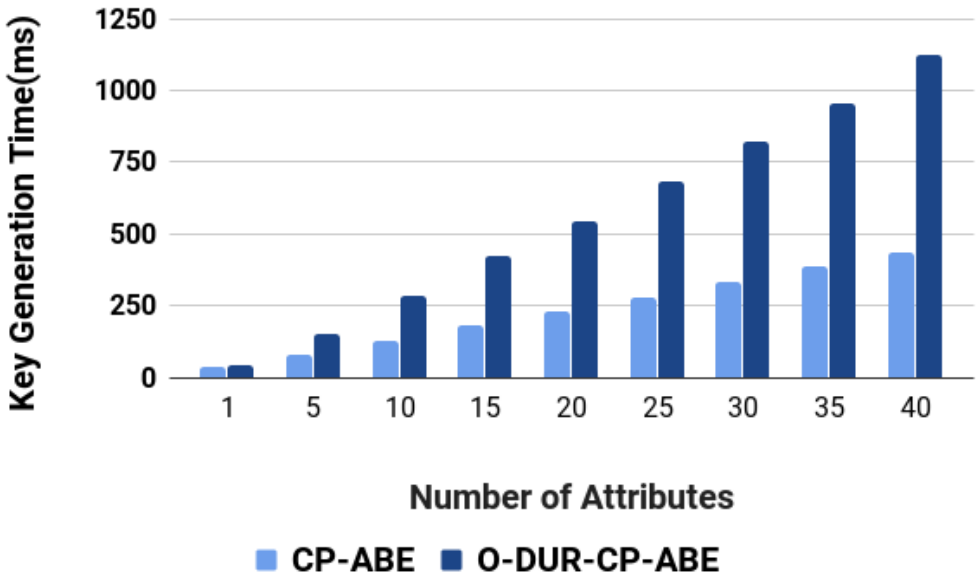

Figure 4.6: Relations between the Number of Attributes and Time Consumption for Key Generation.

Figure. 4.7 and Figure. 4.8 show that the discretionary revocation does not increase the computation overhead on both the Data Owner end where encryption is performed and the User end where decryption is performed.

Figure. 4.6 demonstrates that when H is set to be 2, the key generation overhead of O-DUR-CP-ABE is around two times the overhead of W-CP-ABE. H influences the added overhead, whereas in practice usually $H \leq 10$ (*i.e.*, a small constant), thus the added overhead can be handled easily taking into consideration the increasing computing speed of enterprise servers. From Figure. 4.5, O-DUR-CP-ABE will take much longer time to initialize the system. The good news is the Setup algorithm will be run only once. Furthermore, compared with the attribute-based revocation solutions where the TA has to generate and distribute private updating information of revoked attributes for each data owner discretionary user revocation, this one-time overhead is dominantly more efficient.

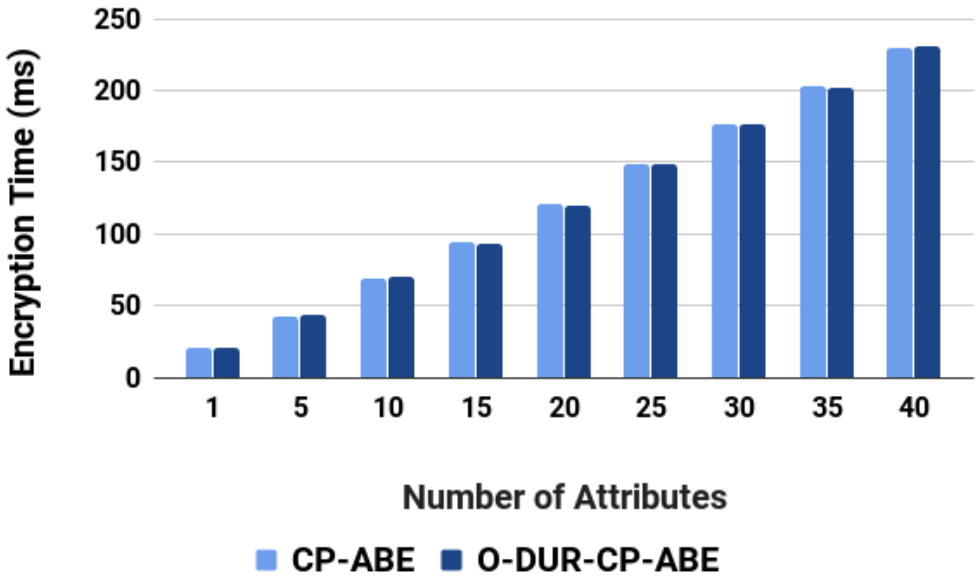

Figure 4.7: Relations between the Number of Attributes and Time Consumption for Encryption.

Figure. 4.9 shows that the computation complexity of the Encrypt and Decrypt algorithm is linear in r. To build a desired broadcast group, there exist trade-offs between system scalability, key storage overhead and computation overhead. Considering the cloud storage application scenarios and the ever-increasing computing speeds, system scalability is much more important than computation overheads. Assume that users with compromised key can be handled by attribute-based revocation, the data owner does not need to worry about these users, instead it just excludes few undesired ones in the group built with the access policy. Since the access policy has already narrowed down the target group, the number of revoked users will generally be small. Moreover, taking the affiliation-based revocation into consideration, in practice, the complexity could be further decreased.

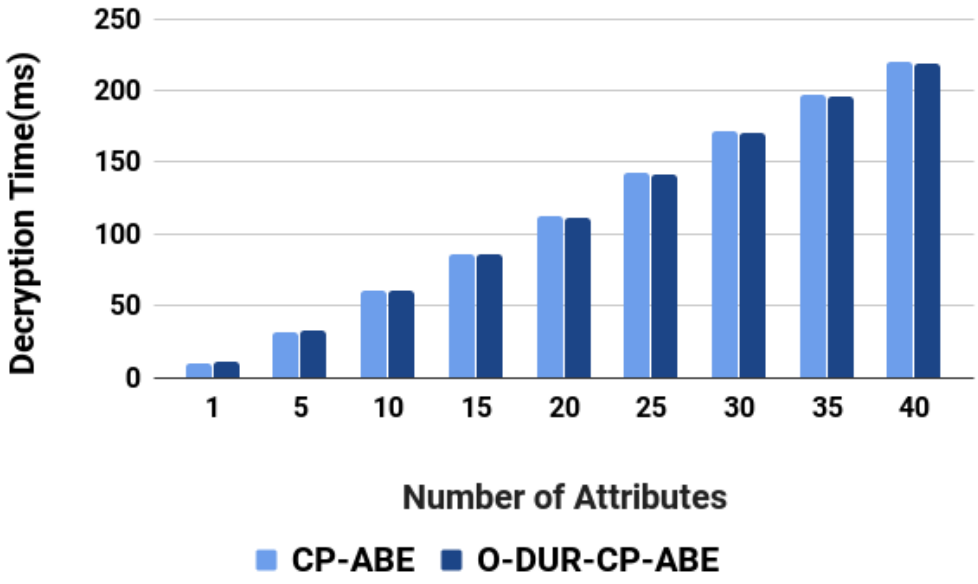

Figure 4.8: Relations between the Number of Attributes and Time Consumption for Decryption.

4.5 Summary

In this chapter, I investigate the problem of how to revoke users when applying the CP-ABE scheme for secure data sharing. Different from the previous researches on attribute-based revocation, our approach focuses on identity-based revocation mechanism. The revocation mechanism remedies the deficiencies of attribute-based revocation that cannot work without the help of the trusted authority and provides more flexible and efficient affiliation-based revocation. I propose the primitive of DUR-CP-ABE, give its security definition and present the constructions. Through analysis and experimental evaluation, I validate the security and efficiency of the proposed scheme. DUR-CP-ABE adds the broadcast encryption (BE) technique Fiat and Naor (1993) by Lewko and Sahai (LS for simplicity) Lewko *et al.* (2010) conjunctively to a CP-ABE scheme Waters (2011). In LS, the size of public and private keys is a constant

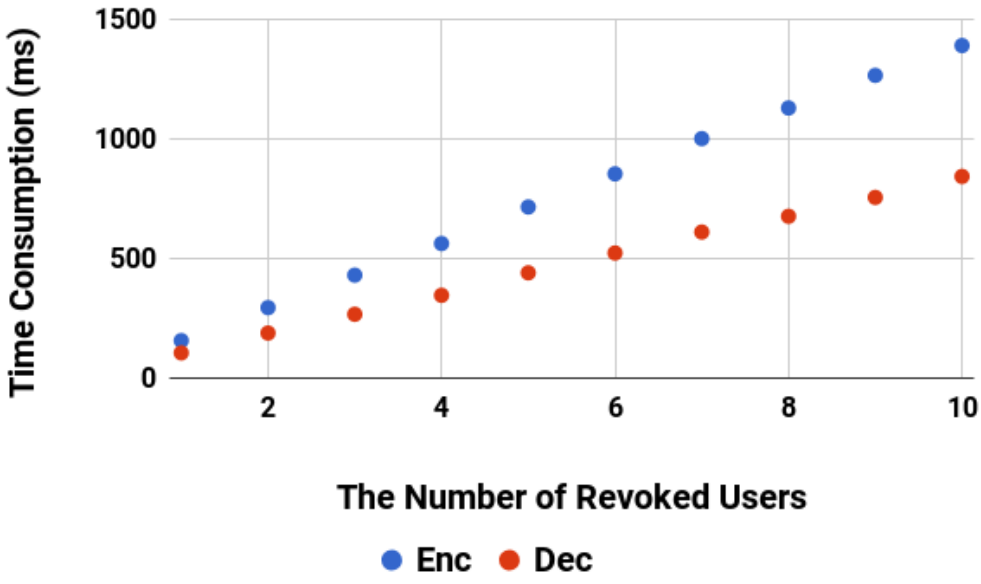

Figure 4.9: Relations between the Number of Revoked Identities and Time Consumption in M-DUR-CP-ABE.

number and the public key allows data owners to encrypt an unbounded number of users, thus is efficient and scalable. Although there exists an BE system Naor *et al.* (2001) (denoted by NNL) with ciphetext size $O(\log n)$ which is smaller than r when $r = O(n)$, in this work I adopted LS because: first, the private key size of NNL is $(O(\log n))^2$; second, with ABE, the access structure has narrowed down the size of candidate authorized users to be much smaller than n, so in practice r is usually much smaller than $\log n$, especially when I consider affiliated user revocation.

There are several research issues need to be further investigated. In this work all the private components of a user's private key are obtained from a single trusted authority, in the next chapter I will investigate how to delegate key generation to the organizations in the hierarchical management structure tree and provide more salient features for practical applications.

Chapter 5

EXTENDED ABE SCHEME SUPPORTING FEDERATION, DELEGATION AND INTER-OPERABILITY

This chapter focuses on the problem of extending the identity revocable ABE scheme presented in the last chapter to support federation, delegation and inter-operability features. I will introduce the background of this problem, present the new ABE scheme and describe performance evaluation.

5.1 Background

5.1.1 A Motivation Use Case

I use an IoT-based application in healthcare as the use-case study. Device-to-device communication in IoT are envisaged through various communication protocols, *e.g.*, Message Queue Telemetry Transport (MQTT) Hunkeler *et al.* (2008), MQTT for sensor networks (MQTT-SN), which have been used by various applications in social networks, remote healthcare, vehicle to vehicle communication, and sensor networks. Existing communication protocols for IoT have limited or devoid of security features. Thus, embedding secure data access control into data can significantly reduce the overhead and complexity on maintaining an uniform access infrastructure on all involved data storage services, and thus break the barrier for data sharing among different administrative domains. As shown in Figure. 5.1, MQTT is a Publisher-Subscriber protocol for data owners to share information with data users through a broker. I will discuss this in the context of a healthcare use-case. In the future, hospitals will be digitized and full of networked medical devices that constantly interact with each other. For example, a blood sugar monitor need to send data to the tablet

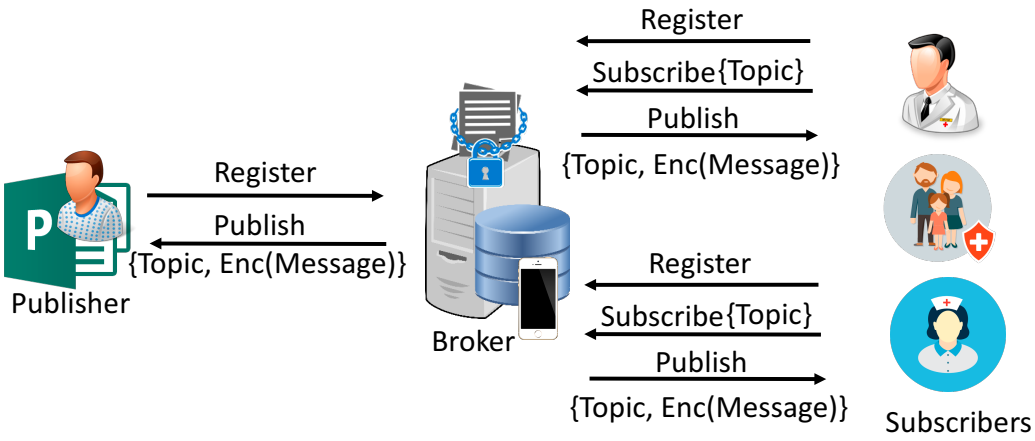

Figure 5.1: An Example of DC-ABAC Secured IoT.

of an attending physician or nurse, to intravenous fluid pumps, to the pharmacy, to medical record logging devices, to diagnostics devices that use data, to consulting physicians in other hospitals, to family members that requested data on their smartphones, etc. The web of information sharing will be extremely complex and dynamic. HIPAA regulations require that medical data should be kept private. That means the data must be protected through access control. It also means that the access control policy must be fine-grained enough to ensure that access is granted to and only to authorized people or devices. HIPPA also demands individual rights HIPAA (2017) to access and manage their health and medical data. The patient must be able to take possession of their healthcare data and share the data in their desired way. The access control must be flexible enough to grant access to arbitrary users. A good solution to the data access control in this application scenario is CP-ABE-based attribute-based access control. In particular, the publisher device encrypts data based on access policies specified by the data owner. Later, a subscriber is able to decrypt the ciphertext if he/she has the attributes that satisfy the access policy associated with the ciphertext.

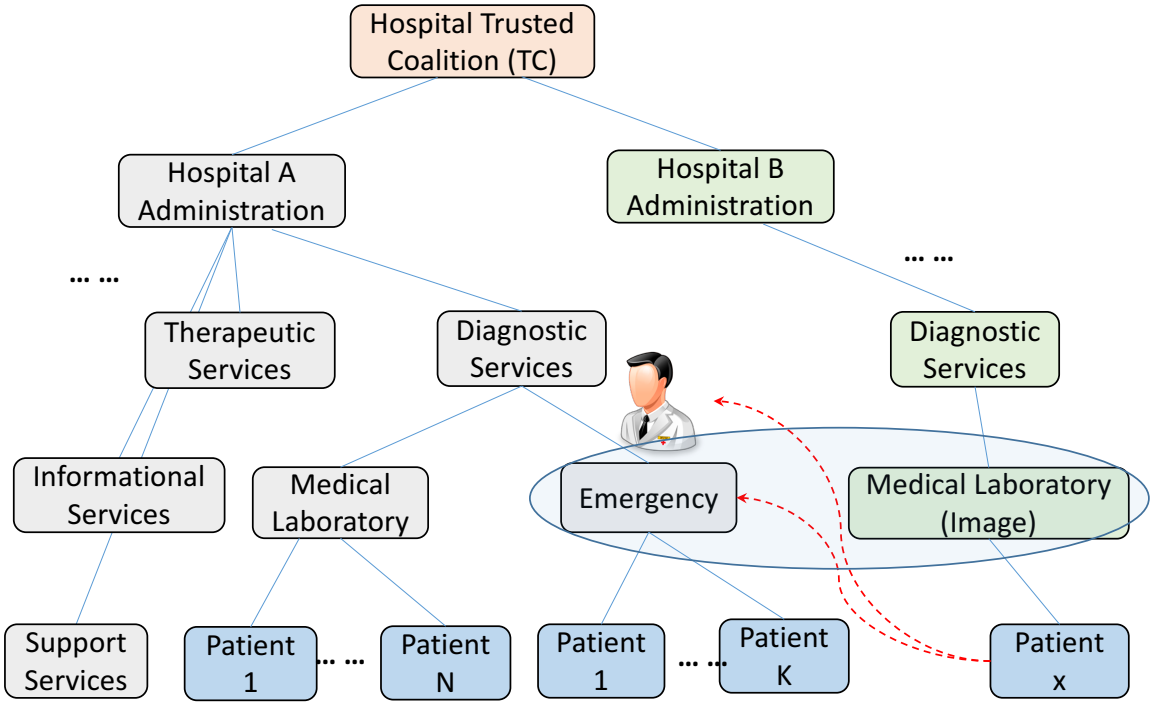

Figure 5.2: An Example of Hospital Organization structure.

I present this solution as Data-Centric ABAC (DC-ABAC). A part of hospital management framework example is shown in Figure. 5.2, where patients may interact with multiple divisions/offices of an hospital. A patient can keep his/her health-care and medical data and provide it to corresponding parties for various healthcare situations, and a doctor is usually associated with a division and may temporarily affiliate with another hospital. Therefore, trust must be established among the coalition of hospitals.

5.1.2 Design Features and Solutions

To achieve the above described application scenario, several important access control management features need to be addressed. Since ABE-based ABAC relies on the cryptographic algorithm, these capabilities need to be addressed by the ABE algorithm itself.

Federation

According to dictionary.com, a federation is "a federated body formed by a number of nations, states, societies, unions, etc., each retaining control of its own internal affairs". Take the aforementioned hospital use case as an example: hospital A and B are federated. The desired feature is that common attributes of hospital A and hospital B could be used in both hospitals. For example, data encrypted by an access policy "MRI image" by a data owner in hospital A could be decrypted by a data user in hospital B with the attribute "MRI image". This requires that the cryptographic parameters should be shared among different hospitals. Generating a common public parameter requires to have a common Trusted Authority (TA), who has the master secret key for each hospital.

However, it is impossible for all hospitals to select a single hospital to play the role of this trusted authority. Even if they could get one candidate, this might be against data privacy laws. Another problem is that this is highly risky. Once the TA got compromised, the data within the federation will all be leaked.

To deal with this issue, in this chapter, I design a protocol involving a coalition of trusted authorities. In particular, the approach is based on a secure multiple party computation protocol running among a of selected participating organizations (say n) to generate system parameters and private keys for the root authority of each organization. The presented solution is resistant against $n-1-out-of-n$ collusion attack among the trusted authorities. Therefore, even $n-1$ participating trusted authorities are compromised, the data within the federation is still secure.

Delegation

ABE schemes require a Trusted Authority (TA) to generate users' private keys, where an ABE key generation authority can grant full or partial of its key generation privileges to one or multiple delegators. Delegating TA's key generation capabilities can not only relieve the single-point failure issue, but also naturally follow the organization's hierarchical management structure. Following the hospital's hierarchical management structure, the delegation is flowing down to the bottom level above patients. Doctors, nurses, and other hospital employees can derive their attributes and the corresponding private keys at their registration delegator, namely parent delegator.

To achieve the delegation feature from a parent to a delegation node (or called child), I design a delegation protocol to allow a parent node to securely delegate its key generation capability to its children. A child can fully or partially inherit the parent's key generation capability based on secrets shared between them. Children delegators cannot collude to derive more capability beyond their delegated key generation capabilities.

Interoperability

In practice, users need to interact with several different organizations that may not trust each other; a user also may own attributes belonged to different subdivisions of an organization. Moreover, at the organization level, it is difficult to find a root authority to generate private keys for all the users in these organizations. In the presented example, a doctor belongs to the hospital A's emergency room and he also affiliates with hospital B in the medical laboratory (imaging division), and thus he can derive two attributes from two delegators in the trust hierarchy and the doctor should be able to use them together.

To achieve the interoperability feature, a closest common ancestor is needed to assist two delegators to issue the private keys for a user. To follow the hierarchical delegation trust framework and prevent collusion issue where two delegators sharing secret to generate secret keys for a user, the presented protocol incorporates a random exponent in each generated private keys to enforce the interoperability protocol must go through the closest ancestor.

Revocation

Revocation is an inherently difficult problem in public key cryptographic algorithms. Existing ABE schemes mainly incorporate a "NOT" logic gate to revoke an attribute when constructing an access policy tree, which may not provide the level of granularity or accuracy. For example, when revoking one specific doctor using a policy $\neg$*(emergency room)* cannot easily identify the revoked user, and more attributes need to be involved. As a result, using a unique identifier for a user or a group is a natural approach and can be easily adopted. Previous approaches Yamada *et al.* (2014) treat a group ID or a user's ID as an attribute, and then it is simple to revoke a known group of entities or individuals by implementing the "NOT" logic on the attribute in the ABE scheme. However, this approach will significantly increase the size of the attribute set and make attributes management extremely complicated when the user-group is large. To address this issue, the solution is to incorporate users' and groups' IDs into their allocated private keys. In this way, a user or a group (e.g., a group generated under one delegator) can be revoked directly by revoking their IDs. The solution will be compatible with traditional attribute-based revocation, and a data owner can build an access policy with both attributes and a set of revoked identities.

5.2 System and Models

5.2.1 DC-ABAC Trust Framework

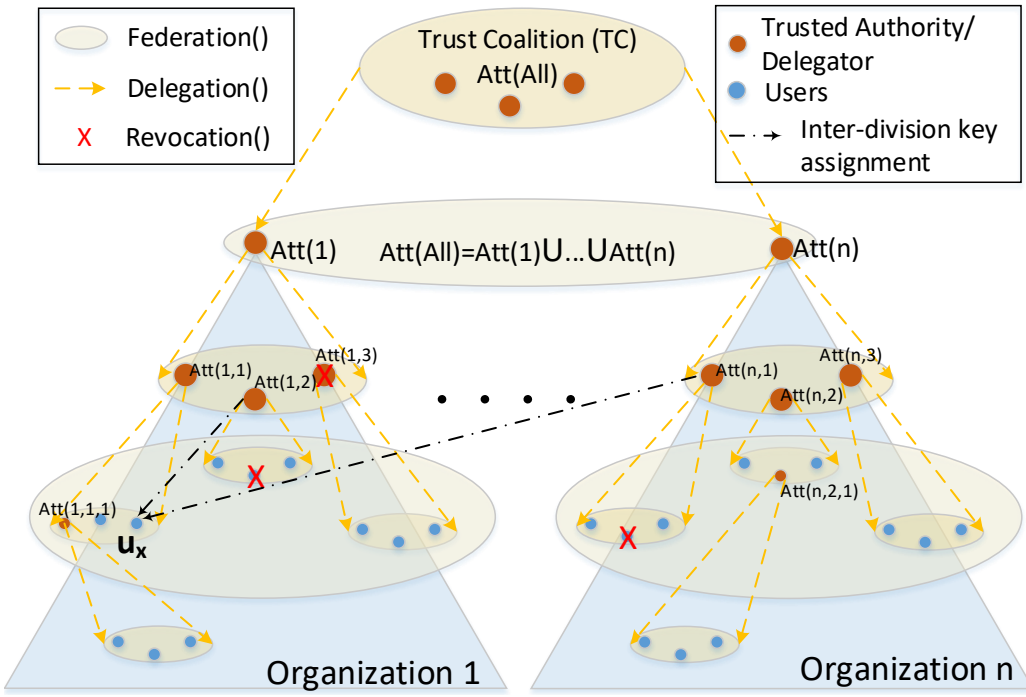

Figure 5.3: DC-ABAC System and Models.

The overall DC-ABAC trust framework is presented in Figure. 5.3, which follows a decentralized trust architecture. At the top, the Trust Coalition (TC) is in charge of all attributes' registration $Att(All)$ and public security parameters management. I propose to use TC to avoid the single point failure issue. Using a multi-party computation protocol, a private key delegation model, where neither a single trusted authority nor a subset of trusted authorities can generate a valid private key for an attribute, can be created.

Delegation follows the hierarchical trust framework. Under TC, multiple organizations exist to manage their attributes, e.g., $Att(i)$, where $i = 1, k.., n$, $|Att(All)| = |Att(1)| \cup \cdots \cup |Att(n)|$, and $|Att(i)| \cap |Att(k)|$ may not be empty ϕ. Each organiza-

tion can form their own trust domain by establishing a hierarchical trust framework, where the organization's root node needs to register its managed attributes at the top level, and it can delegate its key generation capabilities to delegators at a lower level, e.g., $Att(1) \rightarrow Att(1, *)$ and $Att(1, 1) \rightarrow Att(1, 1, 1)$.

5.2.2 Security Model

Compared to traditional access control model, where the data storage service provider is usually fully trusted, in the DC-ABAC solution, the storage servers and communication network are assumed to be honest-but-curious. They might do some statistical analysis over the encrypted data or collude with some unauthorized data users to obtain access to protected data.

As for entities in the organizational structure, they are divided into four categories: *data owner*, *data consumer/user*, *authorities*, and *members of the trust coalition*. The data owner is fully trusted. The data consumer is assumed to be dishonest, *i.e.*, they may collude to access data that they do not have access privileges. In particular, data consumer A with private key SK_A linked with attribute set U_A and data consumer B with private key SK_B linked with attribute set U_B (both key is generated by the protocol External Delegation and $U_A \neq U_B$) might collude together in order to gain access privilege extracted from attribute set $U_A \cup U_B$.

Within one organization, assume that there are two authorities DA_1 and DA_2 with the closest ancestor authority P. DA_1 and DA_2 has private delegation key for generating private keys for attribute set Att_1 and Att_2 respectively ($Att_1 \neq Att_2$). These two domain authorities might want to obtain more access privileges without the authorization from their common ancestor domain authorities, say constructing access policies from attribute set $Att_1 \cup Att_2$. DA_1 and DA_2 could either be domain authorities within an organization or two root authorities of two different organizations. If

they are two authorities within an organization, it is the protocol Internal-Delegation to guarantee the collusion security problem. If they are two root authorities, the Federated Key Generation protocol should be designed to resist against the collusion problem.

Assume that the members in the trusted coalition will not collude with the organizations but parts of the members of the trusted coalition might collude together aiming at controlling the whole structure, but not all of them. Assume that there are N members in the trusted coalition and fewer than $N-1$ members will collude together, the Federated Setup protocol should resist against the collusion attack.

I will analyze each security problems in the security analysis section.

5.3 DC-ABAC Protocols

As shown in Figure. 5.3, there exist two structures among organizations, *i.e.*, the inter and the inner. The inter level corresponds to the federation among multiple organizations which do not trust each other. The inner level corresponds to the organizational structure within each individual organization. The organizational structure is actually a tree structure. For clarity of statement, I define some terminologies related with the tree structure below.

In the presentation, *Root* denotes the top node in a tree, *e.g.*, the root node of the organizational structure is *Att(all)*. *Child* denotes a node directly connected to another node, *e.g.*, the node *Att(1, 1)* is the child of the node *Att(1)*. *Parent* is the converse notion of child, *e.g.*, the node *Att(1)* is the parent of the node *Att(1, 1)*. *Ancestor* is a node reachable by repeatedly proceeding from child to parent, *e.g.*, the node *Att(1)* and *Att(all)* are the ancestor of the node *Att(1, 1)*. *External node or leaf* is a node with no children. For example, an individual user in the organizational structure is denoted by a leaf node. *Internal node* is a node with at least one child.

Level is defined by 1+(the number of connections between the node and the root), *e.g.* the root authority of each organization is on Level-2 since there exists one connection between the root authority and the *Root*.

For all the organizations working in a federated way, I designed a secure multiple party system setup protocol and a secure multiple party computation key generation protocol to generate system parameters and generate private attribute keys for the root authority of each individual organization respectively. The selected group of organizations have benefit collisions, thus will not collude with each other. I name the selected group of organizations TC. The protocols run to enable the functionalities of the DC-ABAC model are presented in the following sections.

5.3.1 Global Setup

In this phase, global parameters are negotiated between the trusted authorities. The global parameters include the universal set of attributes denoted by U, the pairing that will be used. The public parameters include the pairing e being used, the generator g of the group $\mathbb{G}_1$ as well as the group elements of the attributes, denoted by $\{h_x\}$. $GP = \{U, g, e, \{h_x\}_{x \in U}\}$.

5.3.2 Federated Setup and Key Generation Protocol

Each member in TC= $\{TC_1, \cdots, TC_N\}$ (N is the number of members in TC) will run the setup protocol of the ABE scheme to generate their share of the master secret key and the public parameter and then generate the system-wide master secret key and public parameters by running a secure multiple party computation protocol. Figure. 5.4 shows the workflows of system setup and private key generation for organizational root authorities. The two protocols are described as follows.

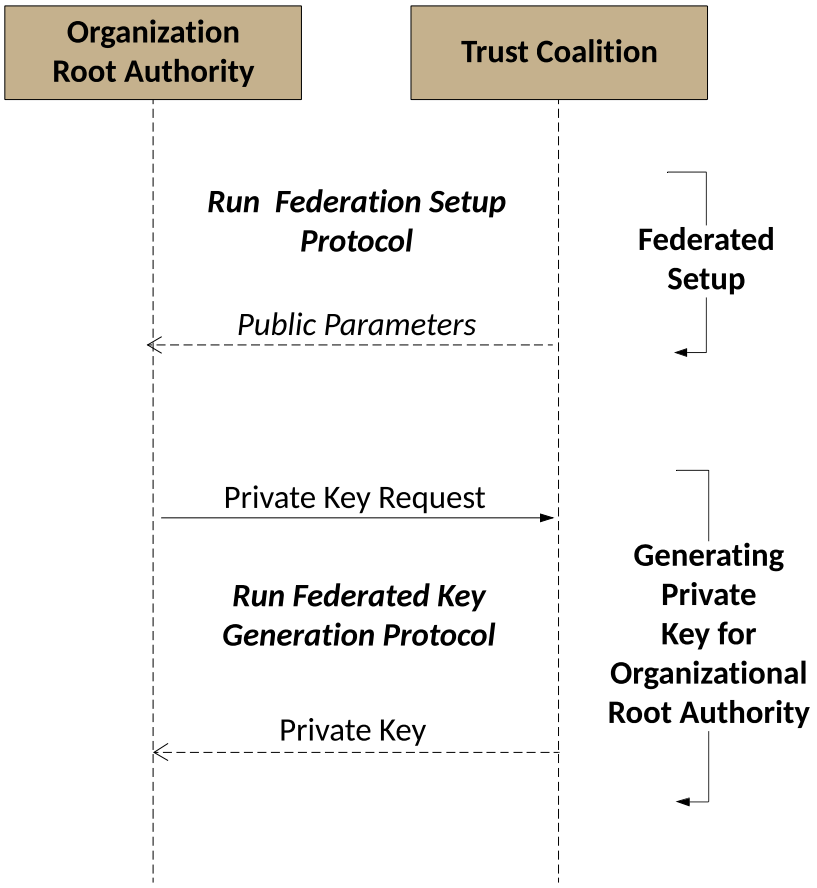

Figure 5.4: Flowchart of Federated Setup and Key Generation.

Federation Setup

Each TC member performs computation and communication as described in Protocol 5.1. When the protocol completes, the system wide master secret key and public parameters will be as follows

$$MSK = (\alpha, b, s_0)$$

$$PK = (g, g^b, g^{b^2}, e(g,g)^\alpha, \{h_x^b, h_x^{b^2}\}_{x \in U}, \{g^{bsID_j{}^{-1}}, g^{sID_j^{-1}}\}_{ID_j \in Orgs})$$

where $\alpha = \sum_{i=1}^{N} \alpha_i, b = \prod_{i=1}^{N} b_i, s_0 = \sum_{i=1}^{N} s_i$.

s_{ID_j} denotes the component generated for all the internal nodes in the tree structure. $sID_j = ID_j^{s_0}$ if ID_j is the root authority of an organization, otherwise,

$s_{ID_{child}} = ID_{child}^{s_{ID_{parent}}}$. $Orgs$ denotes the set of all internal nodes in the tree structure. For statement clarity, I focus on a single root organization use case. This could be extended to multiple organizations scenario in an easy way.

Protocol 5.1 Federated Setup

Input: Each TC_i has inputs of security parameter λ, attribute set U, a prime-order group $\mathbb{G}$, the generator g of $\mathbb{G}$, random elements $\{h_x\}_{x\in\mathbf{U}}$ selected from $\mathbb{G}$. The organization tree structure.

Output: The system public parameters.

1: TC_i generates secret key in the format of $\alpha_i, b_i, s_i \in \mathbb{Z}_p$, and public key in the format of $\big(g^{b_i}, g^{b_i^2}, e(g,g)^{\alpha_i}, (h_x^{b_i}, h_x^{b_i^2})_{x\in U}\big)$.

2: If $i = 1$, then TC_1 will calculate s_{ID} for the root organization, and at the end got the parameters $\big(g^{b_1}, g^{b_1^2}, e(g,g)^{\alpha_1}, (h_x^{b_1}, h_x^{b_1^2})_{x\in U}, s_{ID} = ID^{s_1}\big)$.

3: If $i \neq 1$, TC_i receives parameters in the format of PK_{i-1} and calculates PK_i as shown below.

$$PK_{i-1} = \{pk_1, pk_2, pk_3, \{pk_{x1}, pk_{x_2}\}_{x\in U}, pk_{s_{ID}}\}$$

$$PK_i = \{pk_1^{b_i}, pk_2^{b_i^2}, pk_3 \cdot e(g,g)^{\alpha_i}, \{pk_{x1}^{b_i}, pk_{x_2}^{b_i^2}\}_{x\in U}, pk_{s_{ID}} \cdot ID^{s_i}\}$$

4: If $i \neq N$, TC_i calculates PK_i and sends it to TC_{i+1}.

5: If $i = N$, TC_N calculates all the $\{g^{bs_{ID_j}^{-1}}, g^{s_{ID_j}^{-1}}\}_{ID_j\in Orgs}$ and publishes the public parameters

$$PK = (g, g^b, g^{b^2}, e(g,g)^{\alpha}, \{h_x^b, h_x^{b^2}\}_{x\in U}, \{g^{bs_{ID_j}^{-1}}, g^{s_{ID_j}^{-1}}\}_{ID_j\in Orgs})$$

Federated Key Generation

The members of the trusted coalition work together to generate the private key for the root authority of an organization ID as shown in Protocol 5.2. The generated private key is in the form of SK.

$$SK = (K_0 = g^{\alpha} g^{b^2 t},\ L_a = g^{-s_{ID}^{-1} t},\ L_u = g^{-t},\ K_a = (g^{bs_{ID}} h_x^b)^t,\ K_u = (g^{bID} h_x)^t,$$

$$gbsID = g^{bs_{ID}},\ hx = h_x,\ gbt = g^{bt},\ ht = h_x^t,\ hbt = h_x^{bt},\ s_{ID})_{x \in U_{ID}},$$

where $t = \sum_{i=1}^{N} t_i$, $b = \prod_{i=1}^{N} b_i$, $s_{ID} = ID^s, s = \sum_{i=1}^{N} s_i$ and x is in the set of attributes managed by organization ID. With the private component s_{ID}, each root authority generates private components for the domain authorities with the rule $s_{child} = (ID_{child})^{s_{parent}}$. $g^{bs_{child}^{-1}}$ and $g^{s_{child}^{-1}}$ are part of the system public parameters.

5.3.3 Key Generation Delegation Protocol

On the inner-organization level, the hierarchical structure naturally reflexes the internal organizations' authority and responsibility. For the internal nodes in an organizational structure, the key generation delegation privilege of the parent domain authority could be delegated to a child domain authority. For the external nodes (individual users) in an organizational structure, it is the internal nodes which are the parent of the external nodes to generate the private key for them. The workflow of the key generation delegation is shown in Figure. 5.5.

Delegation-Internal

Protocol 5.3 is run between a parent domain authority ID_P and a child domain authority ID_C to generate the private key for the child domain authority on level $i+1$ based on that of the parent domain authority on level i. The detailed description is presented as follows.

Protocol 5.2 Federated Key Generation

Input: Each TC_i has public parameters, the secret share α_i, b_i, s_i, and the attribute set U_{ID} and the identity ID of the root organization.

Output: The private attribute key of the root organization.

1: TC_i generates $t_i \in \mathbb{Z}_p$.

2: If $i = 1$, TC_1 calculates SK_1 and sends it to TC_2

3: If $i \neq 1$, TC_i receives SK_{i-1} and calculates SK_i as shown below.

$$SK_{i-1} = \{K_0, L_a, L_u, K_a, K_u, gbsID, hx, gbt, ht, hbt\}_{x \in U_{ID}})$$

$$SK_i = \{K_0 \cdot g^{\alpha_i} \cdot (g^{b^2})^{t_{i+1}}, L_a \cdot (g^{-s_{ID}^{-1}})^{t_{i+1}}, L_u \cdot g^{-t_{i+1}}, K_a \cdot (g^{bs_{ID}} h_x^b)^{t_{i+1}}, K_u \cdot (g^{bID} h_x)^{t_{i+1}},$$

$$gbt \cdot (g^b)^{t_{i+1}}, ht \cdot (h_x)^{t_{i+1}}, hbt \cdot (h_x^b)^{t_{i+1}}\}$$

4: If $i \neq N$, TC_i sends the generated components to TC_{i+1}

5: If $i = N$, TC_N sends the generated private key SK to the root authority of organization ID.

Delegation-External

Protocol 5.4 is run by a domain authority to generate a private key for a user.

The components in the private key of the domain authority are updated as follows. The integer t' is a random integer selected by the domain authority.

$$g^{\alpha} g^{b^2 t_u} = g^{\alpha} g^{b^2 t_a} \cdot (g^{b^2})^{t'},$$

$$g^{s_{ja}^{-1} t_u} = g^{s_{ja}^{-1} t_a} \cdot (g^{s_{ja}^{-1}})^{t'},\ g^{-t_u} = g^{-t_a} \cdot g^{-t'},$$

$$(g^{bs_{ja}} h_x^b)^{t_u} = (g^{bs_{ja}} h_x^b)^{t_a} \cdot (g^{bs_{ja}} \cdot h_x^b)^{t'},$$

$$g^{bt_u} = g^{bt_a} \cdot g^{bt'},\ h_x^{t_u} = h_x^{t_a} \cdot h_x^{t'},$$

Protocol 5.3 Delegation-Internal

Input: Parent DA with identity ID_P and attribute set U_{ID_P} has private key SK_{ia} shown below.

$$SK_P = \{K_0 = g^{\alpha} g^{b^2 t_P},\ L_a = g^{-s_{ID_P}^{-1} t_P},\ L_u = g^{-t_P},\ K_a = (g^{bs_{ID_P}} h_x^b)^{t_P},$$

$$K_u = (g^{bID_P} h_x)^{t_P}, gbsID = g^{bs_{ID_P}},\ hx = h_x,\ gbt = g^{bt_P},$$

$$ht = h_x^{t_P},\ hbt = h_x^{bt_P},\ s_{ID_P}\}_{x \in U_{ID_P}},$$

Output: The private attribute key of the child DA with the identity ID_C and attribute set U_{ID_C}

1: The components in the private key of the child domain authority are updated in the following way, where $t' \in \mathbb{Z}_p$ and $x \in U_{ID_C}$

$$g^{\alpha} g^{b^2 t_C} = g^{\alpha} g^{b^2 t_P} \cdot (g^{b^2})^{t'},$$

$$g^{s_{ID_C}^{-1} t_C} = (g^{s_{ID_P}^{-1} t_P})^{s_{ID_P} \cdot s_{ID_C}^{-1}} \cdot (g^{1/s_{ID_C}})^{t\prime},$$

$$g^{-t_C} = g^{-t_P} \cdot g^{-t'},$$

$$(g^{bs_{ID_C}} h_x^b)^{t_C} = (g^{bt_P} \cdot g^{bt'})^{s_{ID_C}} \cdot h_x^{bt_P} \cdot h_x^{bt'},$$

$$(g^{bID_C} h_x)^{t_C} = (g^{bt_P} \cdot (g^b)^{t'})^{ID_C} \cdot h_x^{t_P} \cdot h_x^{t'}$$

$$g^{bs_{ID_C}} = (g^b)^{s_{ID_C}},$$

$$g^{bt_C} = g^{bt_P} \cdot (g^b)^{t'},$$

$$h_x^{t_C} = h_x^{t_P} \cdot h_x^{t'},$$

$$h_x^{bt_C} = h_x^{bt_P} \cdot (h_x^b)^{t'}, s_{ID_C} = ID^{s_{ID_P}}$$

2: The parent DA sends the generated private attribute key to the subdivision.

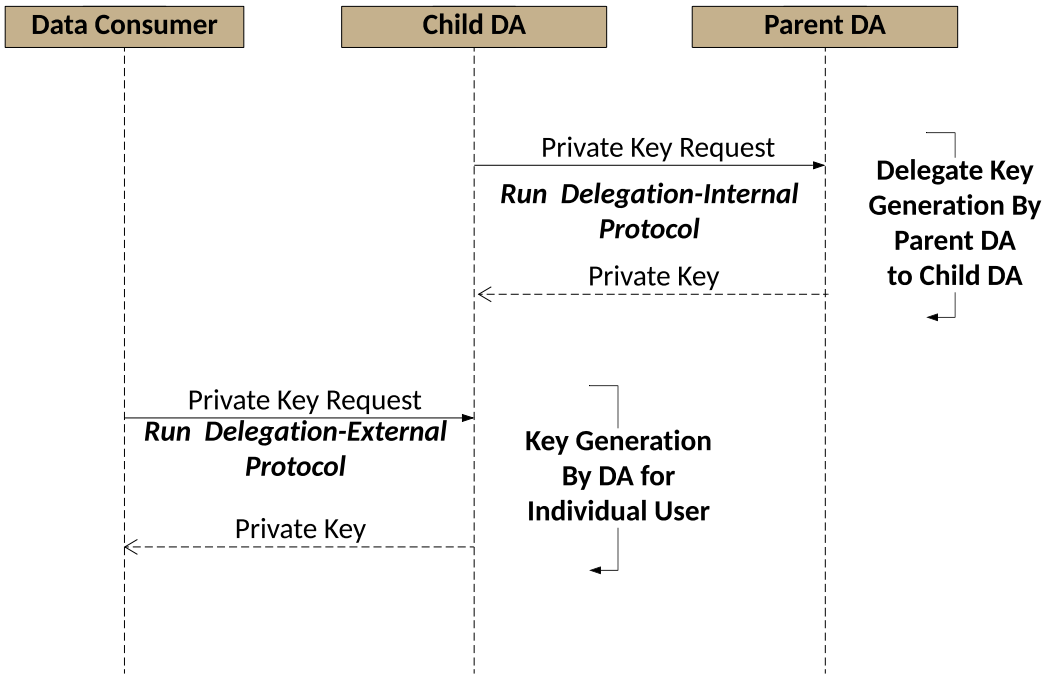

Figure 5.5: Flowchart of Key Generation Delegation.

5.3.4 Interoperability Within and Between Organization

When a user needs attributes from two different domain authorities (DA1 and DA2) within an organization or two root authorities (RA1 and RA2) of two different organizations, the request will finally go to closest common ancestor of these two authorities. From Protocol 5.2 and Protocol 5.3, there exists a random exponent in each domain authority's private key and the user's private key. Therefore, it is impossible to generate a private key with attributes from two authorities. To cancel the random exponent, the common ancestor of the two related authorities has to help to make the random exponent to be the same, so that all these components could work together. I choose the closest common ancestor authority but not the root authority in order to reduce the overheads on the root authority.

Protocol 5.4 Delegation-External

Input: Domain authority with identity ID_a has private key SK_a shown below.

$$SK_a = \{K_0 = g^{\alpha} g^{b^2 t_a},\ L_a = g^{-s_{ID}^{-1} t_a}, L_u = g^{-t_a}, K_a = (g^{b s_{ID_a}} h_x^b)^{t_a}, K_u = (g^{bID} h_x)^{t_a},$$

$$gbsID = g^{b s_{ID_a}},\ hx = h_x,\ gbt = g^{bt_a},\ ht = h_x^{t_a},\ hbt = h_x^{bt_a},\ s_{ID_a}\}_{x \in U_{ID_a}},$$

Output: The private attribute key of an individual user with identity ID_u and attribute set U_{ID_u}.

1: The components in the private key of the child domain authority are updated in the following way, where $t' \in \mathbb{Z}_p$ and $x \in U_{ID_u}$

$$g^{\alpha} g^{b^2 t_u} = g^{\alpha} g^{b^2 t_a} \cdot (g^{b^2})^{t'},$$

$$g^{s_{ID_a}^{-1} t_u} = (g^{s_{ID_a}^{-1} t_a}) \cdot (g^{s_{ID_a}^{-1}})^{t\prime},$$

$$g^{-t_u} = g^{-t_a} \cdot g^{-t'},$$

$$(g^{b s_{ID_a}} h_x^b)^{t_u} = (g^{bt_a} \cdot g^{bt'})^{s_{ID_a}} \cdot h_x^{bt_a} \cdot h_x^{bt'},$$

$$(g^{bID_u} h_x)^{t_u} = (g^{bt_a} \cdot (g^b)^{t'})^{ID_u} \cdot h_x^{t_a} \cdot h_x^{t'}$$

2: The DA sends the generated private attribute key to the user.

5.3.5 Identity Revocable Data Sharing and Access Protocol

As shown in Figure. 5.6, the revocation is enforced during encryption. The data owner would at first construct an access policy tree and then revoke the undesired data consumers by adding their identities into a revocation identity set. Taking the access policy, revoked identity set as well as the plaintext data as inputs, the encryption algorithm outputs the ciphertext to be shared. In this way, only data consumers whose attributes satisfy the access policy and are not revoked by the data owner can decrypt the ciphertext by running the decryption algorithm described below.

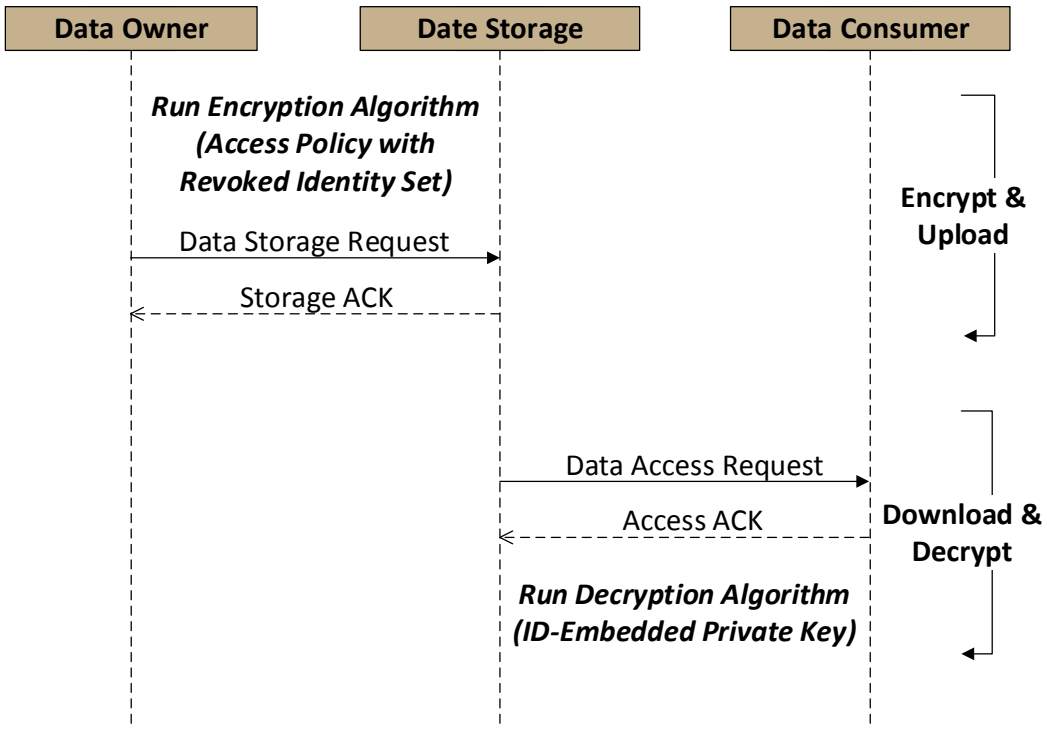

Figure 5.6: Flowchart of Identity Revocable Access Control.

Encrypt$(\boldsymbol{PK}, (M, \rho), \mathcal{M}, \mathbf{ID})$

This is an algorithm revoking both multiple users and multiple domain authorities. The *Encrypt* algorithm takes as inputs an LSSS access structure (M, ρ), where M is an $l \times n$ matrix and the function ρ associates each row of M to corresponding attributes. $\mathbf{ID} = \mathbf{ID}_a \cup \mathbf{ID}_u$ and $|\mathbf{ID}_a| + |\mathbf{ID}_u| = r_a + r_u = r$. Denote the set of revoked domain authority identities as $\mathbf{ID}_a = \{(ID'_{a,j})\}_{j\in[1,r_a]}$. The set of revoked user identities is denoted by $\mathbf{ID}_u = \{ID'_{u,j}\}_{j\in[1,r_u]}$. The *Encrypt* algorithm first chooses a random vector $v = (s, y_2, \cdots, y_n) \in \mathbb{Z}_p^n$. These values will be used to share an encryption exponent s. For $x \in [1, l]$, it calculates $\lambda_x = v \cdot M_x$. The *Encrypt* algorithm chooses random $s \in \mathbb{Z}_p$. The algorithm chooses random μ_a, μ_u such that $\mu = \mu_a + \mu_u$, and $\mu'_1, \cdots, \mu'_{r_u} \in \mathbb{Z}_p$ such that $\mu_u = \mu'_1 + \cdots + \mu'_{r_u}$. The ciphertext is as follows:

$$CT = (C, C_0, \hat{C}_a, \hat{C}_u, \mathbf{ID}), \text{ where}$$

$$C = \mathcal{M}e(g,g)^{\alpha s\mu},$$

$$C_0 = g^{s\mu},$$

$$\hat{C}_a = \Big(C^*_{akj} = g^{bs^{-1}_{ID_{aj}}\lambda_k u_a}, C'_{ak} = (h^{b^2}_{\rho(k)})^{\lambda_k u_a}\Big)^{j\in\mathcal{I}_{nr}}_{k\in[1,l],ID_{aj}\notin ID_a},$$

$$\hat{C}_u = \Big(\{C^*_{ukj} = g^{b\lambda_k u'_j}\}_{k\in[1,l],j\in[1,r_u]},$$

$$\{C'_{ukj} = (g^{b^2\cdot ID_{u,j}} h^b_{\rho(k)})^{\lambda_k u'_j}\}_{k\in[1,l],j\in[1,r_u]}\Big),$$

Decrypt($\boldsymbol{CT}, \boldsymbol{SK}$):

CT is the ciphertext with access structure (M, ρ) and SK is a private key for a set $\mathbf{S}$. Suppose that $\mathbf{S}$ satisfies the access structure and let $\mathbf{I} \subset [1, l]$ be defined as $\mathbf{I} = \{i : \rho(i) \in \mathbf{S}\}$. Let $\{\omega_i \in \mathbb{Z}_p\}_{i\in\mathbf{I}}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret s according to M, then $\Sigma_{i\in\mathbf{I}}\omega_i\lambda_i = s$. For the j^{th} revoked user identity, denote the identity of the non-revoked domain authority administrating ID_u by $ID_{a,j}$. The decryption algorithm calculates $e(g,g)^{b^2ts\mu'_j}$ as follows:

$$(\prod_{i\in\mathbf{I}}[e(K_{\rho(i)u}, C^*_{uij}) \cdot e(C'_{uij}, L_u)]^{\omega_i})^{\frac{1}{(ID_u - ID_j)}}, ID_u \neq ID'_{u,j}$$

Then $e(g,g)^{b^2ts\mu_u}$ could be obtained in the following way:

$$e(g,g)^{b^2ts\mu_u} = \prod_{j\in[1,r_u]} e(g,g)^{b^2ts\mu'_j}.$$

For the j^{th} revoked domain authority, denote the identity of the domain authority on the ${h_j}^{th}$ layer managing ID_u by $ID_{a,j}$. The decryption algorithm evaluates $e(g,g)^{b^2ts\mu_a}$ as follows:

$$e(g,g)^{b^2ts\mu_a} = \prod_{i\in\mathbf{I}}[e(K'_{a\rho(i)}, C^*_{aij}) \cdot e(L_a, C'_{ai})]^{\omega_i}.$$

If SK's holder is not managed by any revoked domain authority and is not among the revoked users, then get $e(g,g)^{\alpha s\mu} = \frac{e(C_0,K)}{e(g,g)^{b^2 ts\mu_u} \cdot e(g,g)^{b^2 ts\mu_a}}$ can be obtained. Finally get the message $\mathcal{M}$ by evaluating $\frac{C}{e(g,g)^{\alpha s\mu}}$.

5.4 Analysis and Evaluation

In this section, the ABE scheme proposed in this paper are evaluated in terms of their computation, storage, and communication performance.

5.4.1 Complexity Analysis

Following the notations provided in TABLE 5.1, I show the complexity analysis summarized in TABLE 5.2 of the designed scheme. There are four types of time-consuming operations: pairing, exponentiation, multiplication and inversion, included in the schemes. Among them, the pairing and exponentiation operations are the dominant costs. Therefore, I utilize the number of pairing and exponentiation operations as metrics for computation complexity of each scheme. The main storage overhead comes from the setup algorithm and key generation algorithm.

Since the setup of the master secret key and system public parameters is performed by all the members of the Trust Coalition, I show the computation complexity for each of the TC member. Each member will perform one pairing and $2|\mathcal{RI}| + |U| + 2$ exponents. Since each member will send the intermediate result to the next member, there will be $2|\mathcal{RI}|+2|U|+3$ elements transmitted from one member to another. After the system setup, all the TC members will store both the public parameters and the share of the master secret key. Totally, the storage complexity will be $2|U|+2|\mathcal{RI}|+7$.

There are two kinds of key generation, the first one is generating private key by the Trust Coalition for the root authority of each organization. The second is the key

Table 5.1: Notations.

$\mathbf{U}$	the attribute set defined in the system, $\|\mathbf{U}\| = m$
$\mathbf{U}_{ID}$	the attribute set of a paritcular domain authority ID
p	the prime order of the multiplicative cyclic group $\mathbb{G}$
m	the number of attributes defined in the system
$\mathbb{Z}_p$	$\mathbb{Z}_p = \{0, 1, \cdots, p-1\}$
$[1, n]$	$[1, n]$ denotes a set of integers *i.e.*, $\{1, 2, \cdots, n\}$
M_x	the x^{th} row of matrix M
l	row number in matrix M of an LSSS access structure $\big(M, \rho\big)$
H	the number of layers in the identity structure tree
r_g	the number of revoked domain authorities
r_u	the number of revoked users
$\mathbf{S}$	the set of attributes created for a specific user
$\mathcal{I}_a$	the set of all the domain authority identities
$\mathcal{I}_{nr}$	the set of non-revoked domain authority identities
$\mathcal{RI}$	the set of root authorities
N	the number of members in the Trust Coaliation

generation by the domain authority within the organization for either child domain authority or individual users. Since generating delegation private keys for the domain authorities within an organization is system overall computation overhead, I exclude it here. The computation complexity of each TC when generating the private key for the root authority is $2|U_{ID}| + 5$, where $|U_{ID}|$ is the number of attributes of the root authority. I assume that the height of the identity structure tree is 2, *i.e.*, $H = 1$.

Table 5.2: Complexity Analysis

Overhead	Setup	KeyGen-RA	KeyGen-U	Encrypt	Decrypt
Computation (Pairing)	1	0	0	1	$2\|\mathbf{I}\|(r_u+1)$
Computation(Exponent)	$2\|\mathcal{RI}\|+2\|U\|+3$	$2\|U_{ID_C}\|+5$	$2\|U_{ID_u}\|+3$	$x(3r_ul+2)$ + $y((\|\mathcal{I}_{nr}\|+1)l+2)$	$x(\|\mathbf{I}\|r_u)+y\|\mathbf{I}\|$
Communication	$2\|\mathcal{RI}\|+2\|U\|+4$	$2\|U_{ID_C}\|+5$	$2\|U_{ID_u}\|+3$	$x(2lr_u)$ + $y(\|\mathcal{I}_{nr}\|l+1)+2$	0
Storage	$2\|\mathcal{RI}\|+2\|U\|+7$	$2\|U_{ID_C}\|+5$	$2\|U_{ID_u}\|+3$	0	0

The complexity of generating a private key for a user is $2|U_{ID_u}| + 4$, where U_{ID_u} is the set of attributes assigned to the user.

One pairing computation is performed during the encryption. the number of exponents is $x(3r_u l + 2) + y((|\mathcal{I}_{nr}| + 1)l + 2)$. If only multiple users are revoked then $x = 1, y = 0$; if only multiple domain authorities are revoked then $x = 0, y = 1$; if there are both multiple users and multiple domain authorities revoked then $x = 1, y = 1$. The communication complexity of the encryption algorithm is $x(2lr_u)+y(1+l|\mathcal{I}_{nr}|)+2$, where x and y is the same as above. The computation of decryption consists of $2|\mathbf{I}|(r_u + 1)$ pairings and $x(|\mathbf{I}|r_u) + y|\mathbf{I}|$ exponents.

Figure. 5.8 to Figure 5.13 show the experimental performance evaluation of the algorithms. The algorithms are implemented in python using PBC library on Mac OS 10.10.5, 1.4 GHz Intel Core i5 and 4 GB 1600 MHz DDR3. I set the number of revoked identities to 1 and evaluate the relations between the number of attributes and the computation overhead. The basic hierarchical management structure is shown as below.

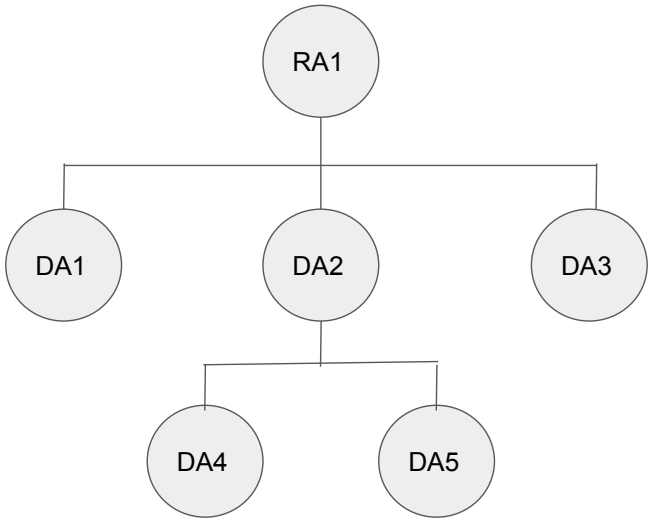

Figure 5.7: An Example of the Organization Structure.

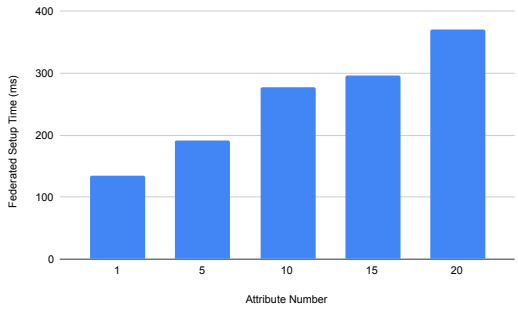

Figure 5.8: Federated Setup

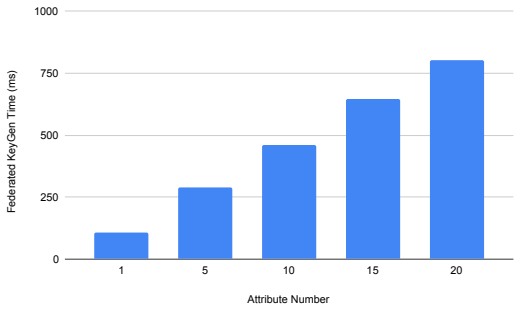

Figure 5.9: Federated KeyGen.

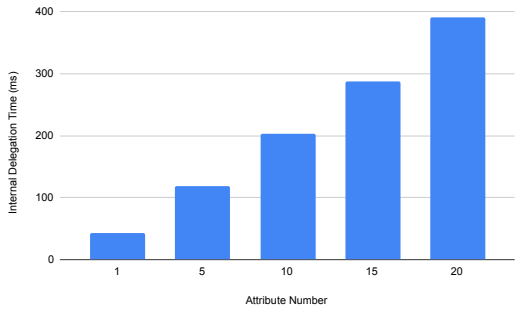

Figure 5.10: Internal Delegation.

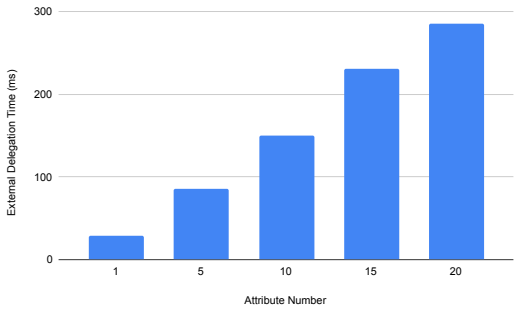

Figure 5.11: External Delegation.

5.4.2 Security Analysis

Security Against Colluding users/Authorities

The proof of security against colluding users and authorities are similar. Because of the limited spaces, here I just demonstrate the proof for resistance against colluding data consumers. Assume there are two individual users A and B with private key SK_A and SK_B as follows.

$$SK_A = (g^{\alpha} g^{b^2 t_A}, L_a = g^{-s_{ID_A}^{-1} t_A}, L_u = g^{-t_A}, K_a = (g^{bs_{ID_A}} h_x^b)_A^t, K_u = (g^{bID_A} h_x)^{t_A},$$

$$gbsID = g^{bs_{ID_A}},\ hx = h_x,\ gbt = g^{bt_A},\ ht = h_x^{t_A},\ hbt = h_x^{bt_A},\ s_{ID_A})_{x \in U_{ID_A}},$$

$$SK_B = (g^{\alpha} g^{b^2 t_B}, L_a = g^{-s_{ID_B}^{-1} t_B}, L_u = g^{-t_B}, K_a = (g^{bs_{ID_B}} h_x^b)^{t_B}, K_u = (g^{bID_B} h_x)^{t_B},$$

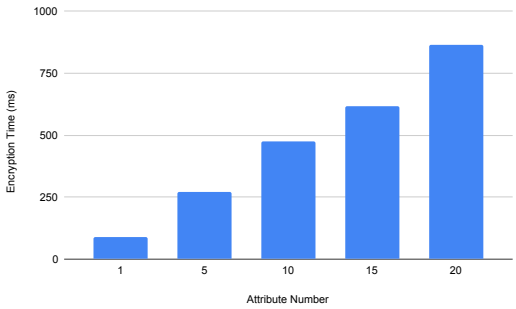

Figure 5.12: Data Encryption.

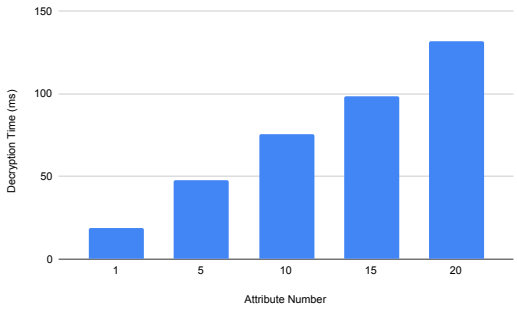

Figure 5.13: Data Decryption.

$$gbsID = g^{bs_{ID_B}},\ hx = h_x,\ gbt = g^{bt_B},\ ht = h_x^{t_B},\ hbt = h_x^{bt_B},\ s_{ID_B})_{x \in U_{ID_B}}.$$

t_A and t_B are two different random integers. Even if A and B put components of their private key together, they cannot produce a valid private key.

Security Against Colluding TC members

Federated efforts from members in the trust coalition are needed in two processes, *i.e.* federated setup and federated key generation for the root authority of each organization. As discussed, I assume the members in the trusted coalition have benefit collision and will not collude with each other. Hence the worst situation will be $N-1$ members collude. Since the final master secret key or the private key contains secret shares from all the members in the trust coalition, without the remaining honest member, the $N-1$ members cannot generate legitimate system parameters or private keys.

5.5 Summary

In this chapter, to perform the ABE-based Attribute-Based Access Control model in federated organizations, I propose a new ABE scheme, which supports federation, delegation, interoperability and identity-based revocation all at once. Compared with

the ABAC model proposed by NIST, the proposed scheme is lack of environmental attributes, which might be time, location, *etc.* In next chapter, I will investigate a new feature, *i.e.*, attribute expiration, which enables the attribute-based encryption schemes to involve environment condition, *i.e.*, time, into the access control policy and attribute distribution.

Chapter 6

AUTOMATIC ATTRIBUTE EXPIRATION IN ABE

This chapter focuses on the problem of adding a new feature, namely attribute expiration, into the ciphertext-policy attribute-based encryption scheme. By supporting this feature, when assigning attributes to users, the trusted authority is capable to control during what time period the assigned attributes are valid. Only users with attributes that satisfy the access policy and all these attributes do not expire are capable to decrypt ciphertexts.

6.1 Background

Ciphertext-policy attribute-based encryption (CP-ABE) provides fine-grained access control over encrypted data. Despite great benefits provided by CP-ABE schemes, they have not been widely adopted. One of the reasons is that existing solutions lack an important feature for any practical security solution, *i.e.*, attribute expiration. In the real-world, users' attributes should only be effective for a limited time period. One way to implement this feature is at the protocol level by rekeying periodically. However, this approach is costly, cumbersome, and inflexible, especially considering the fact that different attributes can have different periods of validity. Another way to implement attribute expiration is by leveraging attribute revocation Yu *et al.* (2010); Attrapadung and Imai (2009). However, revocation itself is highly inefficient, usually requires rekeying of all other users, or non-monotonic access policies with negative attributes Yamada *et al.* (2014). A time-based approach for attribute expiration is presented in BSW Bethencourt *et al.* (2007a). In this approach, the authority simply appends an expiration date to the attribute string. However, this approach leads

to a proliferation of attributes, large access policy trees, and significant interaction overhead between users and the trusted authority.

In this chapter, I propose a new CP-ABE scheme supporting automatic attribute expiration. I take a decentralized CP-ABE scheme Lewko and Waters (2011) as the basis. A comparable numerical range is added for each attribute in a private key, which represents the period of validity. The range is set by an attribute authority for each attribute when it hands out attribute private keys to users. During encryption, data owner sets temporal constrains to all the attributes in the access tree. Attributes are valid for use in decryption if and only if they do not expire. The proposed scheme is an approach without need of online services. It does not rely on the clock and enforces expiration according to the current time. Instead, it simply relies on the time specified by the data owner. To make the attribute management more natural and efficient, the proposed scheme can be decentralized where multiple trust authorities can interoperate. The proposed scheme also supports temporal delegation where users can generate a set or subset of their private attribute key with more constrained time periods validity that can be temporarily handed off to a third party.

In summary, the proposed scheme provides the following salient features.

- **Decentralization**: in the access control system, the data owner is able to share data based on an access control policy written over attributes issued across multiple authorities in different organizations with different roots of trust. This is an important feature since in practice, each organization will manage their relevant attributes while access policies could cross organizational boundaries. For example, an education organization assigns attributes like Professor, Students, Department of Computer Science, *etc*, while a medical organization will assign attributes such as Doctor, Nurse, Department of Emergency, Medicine, *etc*. A data owner might want to share data for academic researches by a medical

school professor and for inspection by doctors in a hospital. The access policy of this data owner will include attributes from both the education organization and the medical organization. With this decentralization feature, attribute private keys can be distributed by different trust authorities independently while are still able to interoperate in a secure way.

- **Automatic Attribute Expiration**: the proposed scheme assigns a time period of validity for each attribute during the key generation phase. When the data owner encrypts data, a temporal value is attached to each attribute in the access policy tree. In this way, the data owner could make a more fine-grained access control on the time dimension. For example, a data user with an assigned attribute "Doctor" with a temporal constrain "From 2011 To 2018" is capable to decrypt encrypted data with an access policy "Doctor" with temporal constraint "2017", while not capable to decrypt it if the access policy is "Doctor" with a temporal constraint "2019" when his attribute expired. In this way, a data user's attributes will only take effect when its assigned effective time period covers the temporal constrain enforced by a data owner. More flexibly, the data owner can choose not only the current time as the temporal constrain but also the time in the past and in the future.

6.2 System and Trust Model

Figure. 6.1 shows the system model in a typical cloud file sharing application, where five types of entities are involved, *i.e.*, the Global Authority, the Data Owner, the Cloud Server, multiple Attribute Authorities and multiple Users (*a.k.a.*, data consumers).

The Global Authority (GA) is responsible for generating the global public parameters GP and publishing them on the cloud server for the other entities in the

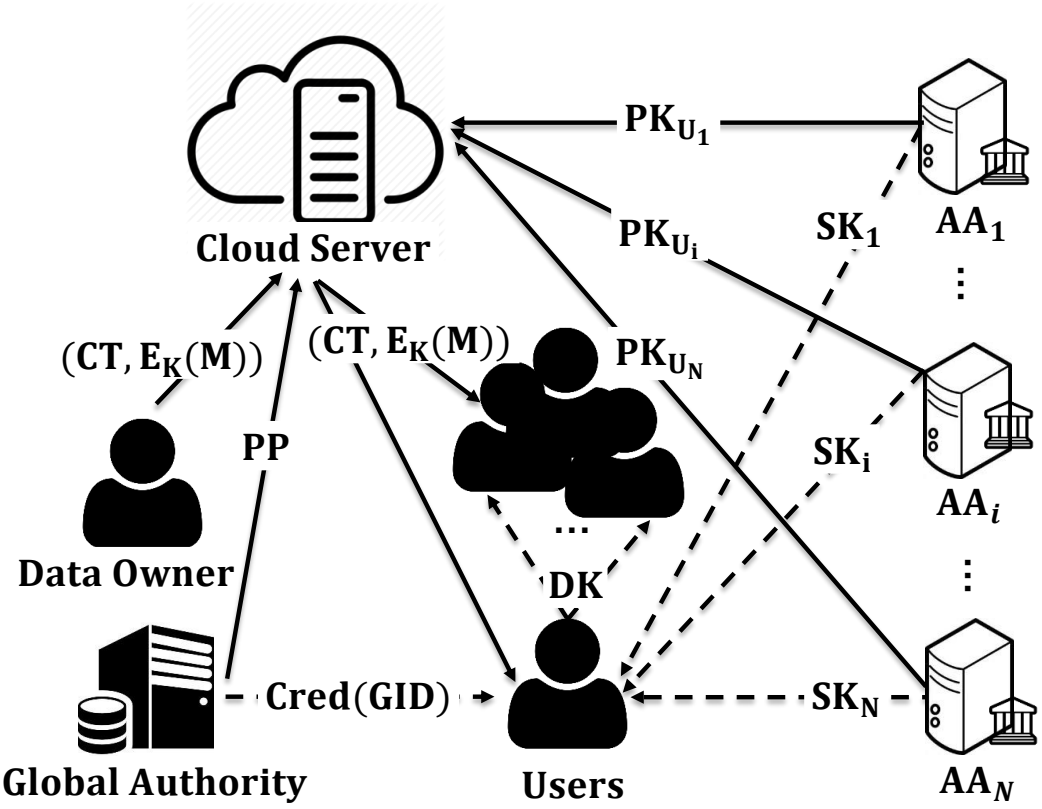

Figure 6.1: System Overview

system to download. The Global Authority also assigns a global unique identifier GID for each user by signing the global identifier by its private key and sending the signature to the user by a secure channel. In multiple authority ABE schemes, global identifiers are significantly important to resist against collusion attacks launched by multiple users aiming at obtaining unauthorized access. GA is not involved in the ABE algorithms but just distributes signature of GID. Therefore, GA does not have any capability of issuing decryption keys.

The Data Owner chooses an access policy and encrypts the data in a hybrid way. In particular, the data is encrypted by a Data Encryption Key (DEK) of a symmetric encryption scheme such as AES, and then the ABE scheme will be used to encrypt the DEK. As shown in Figure. 6.1, K is the DEK used to encrypt the data. The Data

Owner will upload the access policy, the ABE ciphertext of K and the encrypted data to the cloud server.

Each Attribute Authority (AA) manages independently a set of attributes $U_i (i \in [1, N])$ and generates corresponding public attribute keys. Each data user will be ascribed by a set of attributes based on its identity or role in AA's management domain. In the proposed scheme, each attribute authority could manage an unlimited number of attributes. The mapping between attributes and attribute authorities will be stored on the cloud server for users' reference, which I will discuss below. The attribute authorities will issue private keys to a user based on its attributes and global identifier. No coordination between attribute authorities is necessary.

The Cloud Server provides the data storage service to the Data Owner, the data access service to the data users, and also stores the public parameters uploaded by GA and public attribute keys uploaded by each attribute authorities. Instead of relying on the cloud server, the access control in the proposed system is enforced "inside the cryptography".

The User is ascribed by an attribute set that includes attributes entitled by multiple attribute authorities. The User's private key reflects their attributes in each attribute authorities.

The Cloud Server is regarded to be honest-but-curious, *i.e.*, it will follow the designated protocols but try to get sensitive information about the stored data by utilizing the collected data. The global authority is trusted to generate certificates for all the users in the system. With the global authority, each user will be assigned a unique global identifier. No coordination between attribute authorities exists. Each authority will be trusted to issue the designated attributes to authorized users. Users are assumed to be malicious, they might collude to gain authorized data access privileges.

6.3 Preliminaries

6.3.1 Forward/Backward Derivation Functions

I take use of the definition and notations from Zhu *et al.* (2012). Let $U = \{t_1, t_2, \cdots, t_T\}$ denote the countable set of time slots which have total ordering $0 \leq t_1 \leq t_2 \leq \cdots \leq t_T \leq Z$ (Z is the maximum value of the time slots). Define two maps $\psi : U \to V$ with $V = \{v_{t_1}, \cdots, v_{v_T}\}$ and $\bar{\psi} : U \to \bar{V}$ with $\bar{V} = \{\bar{v_{t_1}}, \cdots, \bar{v_{v_T}}\}$.

Definition 6.1. Forward Derivation Function (FDF) Zhu *et al.* (2012): A function is called $f : V \to V$ based on a set $(U, \leq)$ a forward derivation function if it satisfies the following conditions

- Easy to compute: given v_{t_i}, if $t_i \leq t_j$, $v_{t_j} \leftarrow f(v_{t_i})$ can be computed with a polynomial-time algorithm.
- Hard to invert: given v_{t_i}, if $t_i > t_j$, no polynomial-time algorithm can compute v_{t_j} from v_{t_i}.

Definition 6.2. Backward Derivation Function (BDF) Zhu *et al.* (2012): A function is called $\bar{f} : \bar{V} \to \bar{V}$ based on a set $(U, \leq)$ a backward derivation function if it satisfies the following conditions

- Easy to compute: given $\bar{v}_{t_i}$, if $t_i \geq t_j$, $\bar{v}_{t_j} \leftarrow \bar{f}(\bar{v}_{t_i})$ can be computed with a polynomial-time algorithm.
- Hard to invert: given $\bar{v}_{t_i}$, if $t_i < t_j$, no polynomial-time algorithm can compute $\bar{v}_{t_j}$ from $\bar{v}_{t_i}$.

Construction of the functions in this paper is based on the composite order group $\mathbb{G}$ of RSA-type composite order $N = p_1 p_2 p_3$ where p_1, p_2 and p_3 are three large primes.

The constructions are as follows. ϕ is a random generator of group $\mathbb{G}$, *i.e.*, $\phi^N = 1$. $\lambda, \mu \in \mathbb{Z}_N^*$ with sufficiently large order in $\mathbb{Z}_N^*$.

$$v_{t_i} \leftarrow \psi(t_i) = \phi^{\lambda^{t_i}} \in \mathbb{G}$$

$$\bar{v}_{t_i} \leftarrow \bar{\psi}(t_i) = \phi^{\mu^{Z-t_i}} \in \mathbb{G}$$

6.4 The Proposed Solution

6.4.1 Scheme Syntax

The proposed multiple authority CP-ABE scheme consists of the following basic algorithms.

- **GlobalSetup**(λ)$\rightarrow GP$: This algorithm will be run by the global authority. This algorithm takes as inputs the security parameter λ and outputs global parameters GP for the system.

- **AttributeAuthoritySetup**(GP) $\rightarrow SK_i, PK_i$: This algorithm will be run by each attribute authority. For each attribute authority AA_i ($i \in [1, N]$), this algorithm takes GP as inputs and outputs its private/public key pair (SK_i, PK_i).

- **KeyGeneration**(ATC_i, SK_i, GID, GP) $\rightarrow PrivK_{i,GID}$: This algorithm will be run by an attribute authority AA_i where $i \in AAS \subset [1, N]$. ATC_i is the description of attributes together with their temporal constrains for the attribute authority AA_i.

- **KeyCombination**($\{PrivK_{i,GID}\}_{i \in AAS \subset [1,N]}$) $\rightarrow PrivK_{GID}$: This algorithm will be run on the user end, the user obtains private keys from multiple attribute authorities, she will combine them together and generate an integral private key denoted by $PrivK_{GID}$.

- **Encrypt**$(M, (\mathbb{A}, \rho, ATC), GP, \{PK_i\}_{i\in[1,N]}) \rightarrow CT$: This algorithm takes in a message M, an access matrix $(\mathbb{A}, \rho)$, the set of public keys for relevant authorities, and the global parameters. It outputs a ciphertext CT. The access policy will include not only attributes but also the temporal constrains on these attributes.

- **Decrypt**$(CT, GP, PrivK_{GID})$: This algorithm takes in the global public parameters, the ciphertext, and the private attribute keys, which are associated with the same fixed identity GID. If the collection of attributes and their temporal constrains satisfy the access policy corresponding to the ciphertext, it outputs the message M, otherwise it fails.

Correctness Constrains: A multi-authority CP-ABE system is said to be correct if whenever GP is obtained from the global setup algorithm, CT is obtained from the encryption algorithm on the message M, and $PrivK_{GID}$ is the private key generated for user GID and for a set of attributes satisfying the access structure of the ciphertext, $Decrypt(CT, GP, PrivK_{GID}) = M$. The same holds for the private key generated by the *Delegation* algorithm.

6.4.2 Security Model

The following game between a challenger and an attacker is used to define security for the proposed attribute expiration CP-ABE scheme. In the game, the adversaries could query the private keys in an adaptive way. S denotes the set of corrupted attribute authorities and U denotes the universe of attributes. Similar to Lewko *et al.* (2010), for simplicity, I assume each authority only assigns one attribute (in practice, each authority is able to assign multiple attributes).

- **Setup** During this phase the GlobalSetup is run. The adversary defines a set of corrupted authorities by $S' \subset S$. The challenger will get the public/private key pair for the un-corrupted authorities by running the AttributeAuthoritySetup algorithm and sending the public keys to the adversary.

- **Key Query Phase 1** The adversary specifies tuples (i, t_i, GID) and sends them to the challenger where i denotes a particular attribute managed by an un-corrupted attribute authority, t_i is the corresponding effective time period for the attribute i and GID is the global identifier. The challenger returns $PrivK_{i,GID}$ to the adversary.

- **Challenge Phase** The adversary chooses two messages M_0 and M_1, an access matrix $(\mathbb{A}, \rho)$ and temporal constrains for each attributes. The variables have to meet the following requirements. Denote the subset of rows in $\mathbb{A}$ labelled by corrupted attribute authorities' attributes by V. For each global identifier GID, V_{GID} denotes the subset of $\mathbb{A}$ rows labelled by attributes i that has already been queried by the adversary. The constrain is that for each global identifier GID, the subspace spanned by $V \cup V_{GID}$ is required to not include $(1, 0, \cdots, 0)$ or at least one time period of the quarried attribute does not meet the temporal access policy. That is, the adversary is required not to ask for a set of private keys which are able to decrypt. The adversary is required to send the challenger the public keys for the corrupted attribute authorities whose attributes are included in the labelling ρ. After this, the challenger chooses a random bit $\beta \in \{0, 1\}$ and sends the attacker the ciphertext of M_β encrypted with the access matrix $(\mathbb{A}, \rho)$ and a selected time point.

- **Key Query Phase 2** In query phase2, the adversary is allowed to submit more private key queries under the same constrain as described in the Challenge phase.

- **Guess** The adversary outputs a guessed bit β' for β. If $\beta' = \beta$, the adversary wins the game, otherwise fails.

The attacker's advantage in this game is defined to be

$$Advantage = |\Pr[\beta = \beta'] - \frac{1}{2}|$$

Definition 6.3. A scheme defined based on the syntax above is secure (against static corruption of attribute authorities) if no polynomial time adversaries can win the game with a non-negligible advantage in the security game above.

6.4.3 Construction Overview

The key idea of the construction is to split the shared secret w_x in a linear secret sharing scheme (LSSS) into two parts. One part is for the value of the attribute, and the other is for the temporal constraint. These two parts together consist of the integral constraint in an access control policy. The same idea is enforced to the temporal constraint. One is for the lower bound and the other is for the upper bound. In this way, when a private entry corresponding to a particular attribute is used, the decryption algorithm will check not only whether the attributes satisfy the access policy but also whether their temporal constraints meet the requirement.

6.4.4 Scheme Construction

GlobalSetup$(\lambda) \rightarrow GP$ In the global setup, a bilinear group $\mathbb{G}$ of order $N = p_1p_2p_3$ is chosen. The global public parameters, GP, are N and a generator g_1 of $\mathbb{G}_{p_1}$.

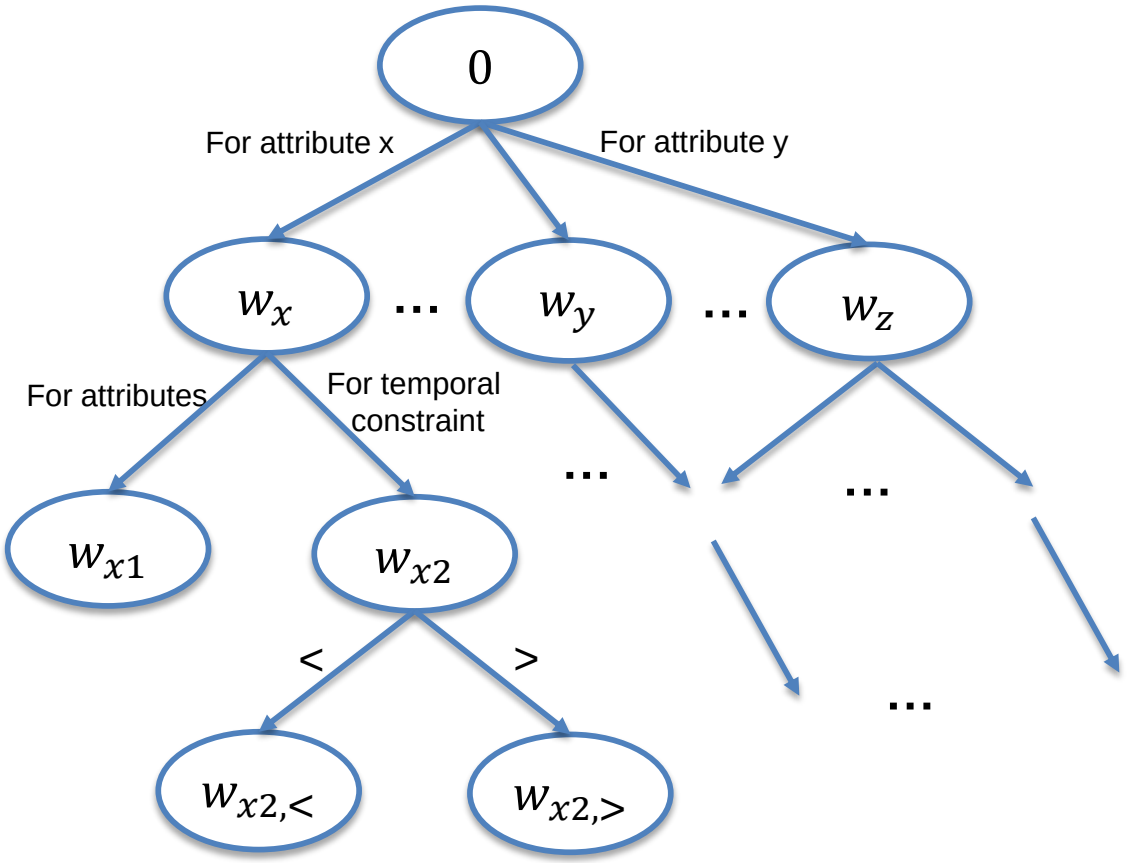

Figure 6.2: Share of Parameters in the Construction

Additionally, the description of a hash function $H : \{0,1\}^* \rightarrow \mathbb{G}$ that maps global identities GID to elements of $\mathbb{G}$ is also published.

AttributeAuthoritySetup$(GP) \rightarrow (PK, SK)$ For each attribute i managed by the attribute authority, the authority chooses two random exponents $\alpha_i, y_i \in \mathbb{Z}_N^*$ and publishes $PK = \{e(g_1, g_1)^{\alpha_i}, g_1^{y_i}\}$ as its public key. It keeps private $SK = \{\alpha_i, .y_i\}$ as its secret key.

KeyGen$(GID, ATC_i, SK, GP) \rightarrow Priv_{i,GID}$ To create a key for GID for attribute ATC_i managed by an authority, the authority computes:

$$K_{i,GID} = g_1^{\alpha_i} H(GID)^{y_i}$$

$$K_{i,GID}^{<} = g_1^{\alpha_i}(H(GID)^{\lambda^{t_{i,<}}}\phi^{\lambda^{t_{i,<}}})^{y_i}, K_{i,GID}^{>} = g_1^{\alpha_i}(H(GID)^{\lambda^{Z-t_{i,>}}}\phi^{\lambda^{Z-t_{i,>}}})^{y_i}$$

In this algorithm, the attribute authority will assign both attributes and the related temporal constraints on them. In practical applications, this interval could be a set. If that is the case, in the encryption, the ciphertext need to be extended to support this. I denote the private key as follows:

$$Priv_{i,GID} = (K_{i,GID}, K^{<}_{i,GID}, K^{>}_{i,GID})$$

Encrypt$(M, (\mathbb{A}, \rho, ATC), GP, \{PK\}) \rightarrow CT$ The encryption algorithm takes in a message M, an $l \times n$ access matrix $\mathbb{A}$ with ρ mapping its rows to attributes, the global parameters, and the public keys of the relevant authorities. It chooses a random $s \in \mathbb{Z}_N$ and a random vector $v \in \mathbb{Z}_N^n$ with s as its first entry. Let λ_x denote $\mathbb{A}_x \cdot v$, where $\mathbb{A}_x$ is row x of $\mathbb{A}$. It also chooses a random vector $w \in \mathbb{Z}_N^n$ with 0 as its first entry. Let ω_x denote $\mathbb{A}_x \cdot \omega$. For each row $\mathbb{A}_x$ of $\mathbb{A}$, it chooses a random $r_x \in \mathbb{Z}_N$. The current time is denoted by t'_c. $ATC = t'_c$ The ciphertext is computed as

$$C_0 = Me(g_1, g_1)^s, C_{1,x} = e(g_1, g_1)^{\lambda_x} e(g_1, g_1)^{\alpha_{\rho(x)} r_x},$$

$$C_{2,x} = g_1^{r_x}, C_{3,x} = g_1^{y_{\rho(x)} r_x} g_1^{\lambda^Z \cdot \omega_{x1}}$$

$$C_{3,x,<} = g_1^{y_{\rho(x)} r_x} g_1^{\omega_{x2,<} \cdot \lambda^{Z - t'_c}}, C_{3,x,>} = g_1^{y_{\rho(x)} r_x} g_1^{\omega_{x2,>} \cdot \lambda^{t'_c}}$$

$$C'_{3,x,<} = g_1^{y_{\rho(x)} r_x} g_1^{\omega_{x1} \cdot \lambda^{Z - t'_c}}, C'_{3,x,>} = g_1^{y_{\rho(x)} r_x} g_1^{\omega_{x2} \cdot \lambda^{t'_c}}$$

$$C_{4,x,<} = \phi^{\lambda^{t'_c}}, C_{4,x,>} = \phi^{\lambda^{Z - t'_c}}$$

$$\omega_{x1} + \omega_{x2} = \omega_x, \omega_{x2,<} + \omega_{x2,>} = \omega_{x2}.$$

Decrypt$(CT, \{Priv_{i,GID}\}, GP) \rightarrow M$ I assume the ciphertext is encrypted under an access matrix $(\mathbb{A}, \rho)$. To decrypt, the decryptor first computes $H(GID)$. If the decryptor has the secret keys $\{K_{\rho(x),GID}\}$ for a subset of rows $\mathbb{A}_x$ of $\mathbb{A}$ such that $(1, \cdots, 0)$ is in the span of these rows, then the decryptor proceeds as follows. For each such x, the decryptor computes:

$$E_{x1} = \frac{C_{1,x} \cdot e(H(GID), C_{3,x})}{e(K_{\rho(x),GID}, C_{2,x})} = e(g_1, g_1)^{\lambda_x} \cdot e(H(GID), g_1)^{\lambda^Z \cdot \omega_{x1}}$$

$$E_{x2} = \frac{C_{1,x}^{\lambda^{t'_c - t_{\rho(x),<}}} \cdot e(H(GID)^{\lambda^{t'c}}, C_{3,x,<}) \cdot e(C_{4,x,<}, C'_{3,x,<})}{e((K^{<}_{\rho(x),GID})^{\lambda^{t'_c - t_{\rho(x),<}}}, C_{2,x})}$$

$$= e(g_1, g_1)^{\lambda_x \lambda^{t'_c - t_{\rho(x),<}}} \cdot e(H(GID), g_1)^{\lambda^Z \cdot \omega_{x2,<}} \cdot e(\phi^{\lambda^Z}, g_1)^{\omega_{x1}}$$

$$E_{x3} = \frac{C_{1,x}^{\lambda^{t'_c - t_{\rho(x),<}}} \cdot e(H(GID)^{\lambda^{t'_c}}, C_{3,x,>}) \cdot e(C_{4,x,>}, C'_{3,x,>})}{e((K^{>}_{\rho(x),GID})^{\lambda^{t_{\rho(x),>} - t'c}}, C_{2,x})}$$

$$= e(g_1, g_1)^{\lambda_x \lambda^{t_{\rho(x),>} - t'c}} \cdot e(H(GID), g_1)^{\lambda^Z \cdot \omega_{x2,>}} \cdot e(\phi^{\lambda^Z}, g_1)^{\omega_{x2}}$$

$$E_x = E_{x1} \cdot E_{x2} \cdot E_{x3}$$

$$= e(g_1, g_1)^{\lambda_x (1 + \lambda^{t_{\rho(x),>} - t_{\rho(x),<}})} \cdot e(H(GID), g_1)^{\lambda^Z \cdot \omega_x} \cdot e(\phi^{\lambda^Z}, g_1)^{\omega_x}$$

The decryptor then chooses constants $c_x \in \mathbb{Z}_N$ such that $\sum_x c_x \mathbb{A}_x = (1, 0, \cdots, 0)$ and computes:

$$\prod_x \left(E_x\right)^{c_x} = e(g_1, g_1)^{(1+\lambda^\Delta)s}$$

(Recall that $\lambda_x = \mathbb{A}_x \cdot v$ and $\omega_x = \mathbb{A}_x \cdot \omega$, where $v \cdot (1, 0, \cdots, 0) = s$ and $\omega \cdot (1, 0, \cdots, 0) = 0$.) The message can then be obtained as:

$$M' = M^{(1+\lambda^\Delta)} = C_0^{(1+\lambda^\Delta)} / \prod_x \left(E_x\right)^{c_x}.$$

$$M = M'^{(1+\lambda^\Delta)^{-1}}$$

6.5 Analysis and Evaluation

6.5.1 Security Analysis

To prevent collusion attacks, I use the global identifier to "tie" together the various attributes belonging to a specific user so that they cannot be combined with another user's attributes in decryption. More specifically, the encryption algorithm blinds the

message M with $e(g_1, g_1)^s$, where g_1 is a generator of the subgroup $\mathbb{G}_{p_1}$, and s is a randomly chosen value in $\mathbb{Z}_N$. The value s is then split into shares λ_x according to the LSSS matrix, and the value 0 is split into shares ω_x. The decryptor must recover the blinding factor $e(g_1, g_1)^s$ by pairing their keys for attribute, identity pairs (i, GID) with the ciphertext elements to obtain the shares of s. In doing so, the decryptor will introduce terms of the form $e(g_1, H(GID))^{\omega_x}$. If the decrytor has a satisfying set of keys with the same identity GID, these additional terms will cancel from the final result, since the ω_x's are shares of 0. If two users with different identities GID and GID' attempt to collude and combine their keys, then there will be some terms in the format $e(g_1, H(GID))^{\omega_x}$ and the other terms in the format $e(g_1, H(GID'))^{\omega_x}$, and they will not cancel with each other, therefore preventing the recovery of $e(g_1, g_1)^s$.

The proposed scheme is based on Lewko's work Lewko and Waters (2011). To prove the security of the proposed scheme. A simple way is to prove that all the changes made by the proposed scheme will not harm the original scheme's security.

The difference between the proposed newly constructed scheme and the basic decentralized ABE, denoted by DCABE, lie in two parts. The first part is private key and the second part is the ciphertext.

At first, let's compare the two schemes' keys. DCABE's generated private key is in the following format:

$$K'_{i,GID} = g_1^{\alpha_i} H(GID)^{y_i}$$

The proposed scheme is in the following format.

$$K_{i,GID} = g_1^{\alpha_i} H(GID)^{y_i}$$

$$K^{<}_{i,GID} = g_1^{\alpha_i} (H(GID)^{\lambda^{t_{i,<}}} \phi^{\lambda^{t_{i,<}}})^{y_i}$$

$$K^{>}_{i,GID} = g_1^{\alpha_i} (H(GID)^{\lambda^{Z-t_{i,>}}} \phi^{\lambda^{Z-t_{i,>}}})^{y_i}$$

The difference is that I keep their part and add two more. In the key generation part, the most important issue is to keep the master secret key y_i and α_i secret. It is easy to verify that the two newly added component will not leak any information about α_i or y_i.

Second, let's compare the two schemes' ciphertexts or the encrytion algorithm. The goal of DCABE is that only users with attributes satisfying the access policy could decrypt the ciphertext. For the proposed scheme, not only the attributes need to satisfy the access policy, but also these attributes do not expire. The DCABE's ciphertext is as follows.

$$CT =< C_0, C_{1,x}, C_{2,x}, C_{3,x} >,\ where$$

$$C_0 = Me(g_1, g_1)^s, C_{1,x} = e(g_1, g_1)^{\lambda_x} e(g_1, g_1)^{\alpha_{\rho(x)} r_x},$$

$$C_{2,x} = g_1^{r_x}, C_{3,x} = g_1^{y_{\rho(x)} r_x} g_1^{\omega_x}, \forall x.$$

The ciphertext constructed in the proposed scheme is as follows:

$$C_0 = Me(g_1, g_1)^s, C_{1,x} = e(g_1, g_1)^{\lambda_x} e(g_1, g_1)^{\alpha_{\rho(x)} r_x},$$

$$C_{2,x} = g_1^{r_x}, C_{3,x} = g_1^{y_{\rho(x)} r_x} g_1^{\lambda^Z \cdot \omega_{x1}}$$

$$C_{3,x,<} = g_1^{y_{\rho(x)} r_x} g_1^{\omega_{x2,<} \cdot \lambda^{Z - t'_c}}, C_{3,x,>} = g_1^{y_{\rho(x)} r_x} g_1^{\omega_{x2,>} \cdot \lambda^{t'_c}}$$

$$C'_{3,x,<} = g_1^{y_{\rho(x)} r_x} g_1^{\omega_{x1} \cdot \lambda^{Z - t'_c}}, C'_{3,x,>} = g_1^{y_{\rho(x)} r_x} g_1^{\omega_{x2} \cdot \lambda^{t'_c}}$$

$$C_{4,x,<} = \phi^{\lambda^{t'_c}}, C_{4,x,>} = \bar{\phi}^{\lambda^{Z - t'_c}}$$

$$\text{and } \omega_{x1} + \omega_{x2} = \omega_x, \omega_{x2,<} + \omega_{x2,>} = \omega_{x2}.$$

I split ω_x into ω_{x1} and ω_{x2} in a random way. Therefore, the first part of the ciphterxt shown below:

$$C_0 = Me(g_1, g_1)^s, C_{1,x} = e(g_1, g_1)^{\lambda_x} e(g_1, g_1)^{\alpha_{\rho(x)} r_x},$$

$$C_{2,x} = g_1^{r_x}, C_{3,x} = g_1^{y_{\rho(x)} r_x} g_1^{\lambda^Z \cdot \omega_{x1}}$$

will not even disclose the plaintext even if the attributes satisfy the access policy. The users have to recover the ω_{x2} by using the temporal part of their private key.

$C_{3,x,>}$ and $C_{3,x,<}$ are only related with w_{x2}, and could not leak any information about w_x.

$C'_{3,x,>}$ and $C'_{3,x,<}$ are in similar format, however the coefficients are different, so they could not be used together in the decryption according to the generation way of w_x. Therefore, these newly added components will not harm the security of the DCABE scheme as well.

$C_{4,x,<}$ and $C_{4,x,>}$ are ciphertexts to ensure the temporal constrains. Only when all users attributes used in the decryption do not expire can they run the backward and forward function to recover the components used in the decryption. These two components are unrelated with the DCABE scheme ciphertext, therefore will not harm the security of the DCABE ciphertext.

There is a requirement on the temporal constrains of each attribute. First, the assigned valid time for all the attribute could not be the same. The reason is that if they are all the same, it is easy to get $g_1^{\lambda^Z \cdot \omega_x}$, then the temporal constrains will be effective anymore. The key idea is to avoid the parameters of $y_{\rho(x) r_x}$ to be the same for all the attributes. An easy way to deal with this is to add a dummy attribute which will be assigned to all the users and will be used in all the encryption operations with an "AND" gate in the access policy tree.

Taking all-above analyses into consideration, if the DCABE scheme is secure in terms of the security model described in 6.4.2, then the constructed scheme is also secure in terms of that security model. According to Lewko and Waters (2011), the DCABE scheme is proven to be secure if some hard problems defined in the composite pairing as I used in this scheme is computationally hard.

6.5.2 Performance Analysis

According to Zhu *et al.* (2012), the order of the composite pairing have to ensure the security of the RSA problem, chose type A1 bilinear pairing from JPBC library with order that is 2048-bits. I implement the scheme with Java and tested the scheme efficiency on Mac OS 10.10.5, 1.4 GHz Intel Core i5 and 4 GB 1600 MHz DDR3. Usually, the order of prime order bilinear pairing is 160-bits, and composite bilinear pairing with each prime being 160-bits. However, in our scheme, to implement the expiration feature and ensure the security of the FDF and BDF, the parameter is much larger, thus making the scheme efficiency issues. In this section, the test of the scheme is performed only with the number of attribute from 1 to 5, and leave the scheme optimization as future work. I take the following setting when testing the implementation. First, there is only one attribute authority. Second, all access policies consist of "AND" gates, which are the most time-consuming access policies. Third, the maximum time slot Z is set to be 10000000. Fourth,

I test the attribute authority setup algorithm assuming that all the attributes are managed by a single attribute authority.

6.6 Summary

In this chapter, I propose a new solution to the problem of automatic attribute expiration in the ciphertext-policy attribute-based encryption schemes. During private

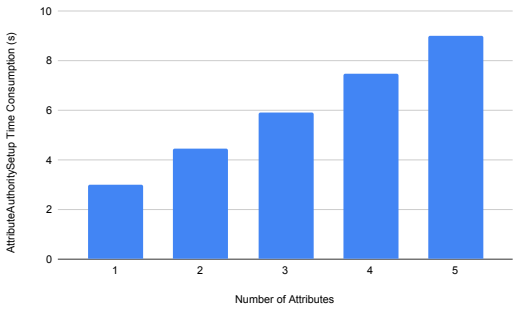

Figure 6.3: Attribute Authority Setup.

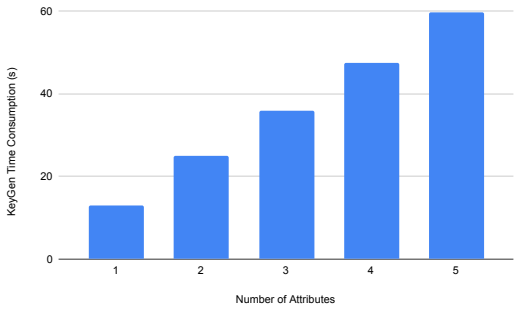

Figure 6.4: Key Generation.

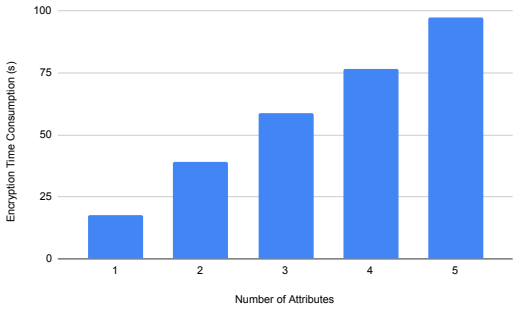

Figure 6.5: Data Encryption.

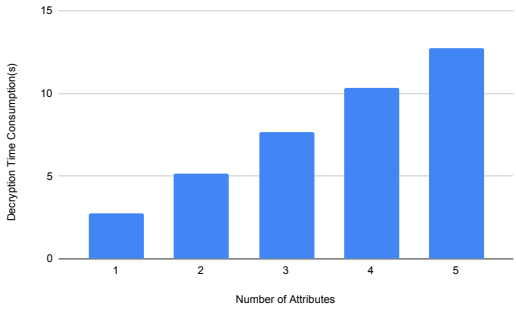

Figure 6.6: Data Decryption.

key generation phase, the attribute authorities will assign a corresponding expiration time constraint for each attributes. When a data owner encrypts the data, he or she will also add a time point for each attribute included in the access policy. Commonly the time point will be the current time according to how certificate expiration is checked. In such a way, the proposed scheme can achieve salient features all together, which cannot be achieved by existing schemes, *i.e.*, decentralization and automatic attribute expiration. Through security analysis, I show that the proposed scheme is secure in the defined security model. In the future, I would like to investigate how to enhance the efficiency of the proposed scheme.

Chapter 7

DISCRETIONARY ATTRIBUTE-BASED ACCESS CONTROL IN HYPERLEDGER FABRIC

In this chapter, I will study how to enable discretionary attribute-based access control in Hyperledger Fabric, which is a popular blockchain platform being used in the industry. I will describe the workflow of Hyperledger Fabric, its trust framework, the existing ABAC implementation in Hyperledger Fabric. Then I will present the motivation of using ABE in Hyperledger Fabric by a simple example. Based on this, I will describe how to add ABE into the existing trust framework of Hyperledger Fabric. In addition, I will show how to implement the proposed trust framework and show how ABE could help provide fine-grained access control over sensitive data in Hyperledger Fabric by a file-sharing application.

7.1 Hyperledger Fabric

In this section, I show an example system model of a hyperledger fabric application. In Figure. 7.1 I show a channel where clients and peer nodes are from three different organizations, *i.e. Organization 1, Organization 2* and *Organization 3*. For simplicity, all the clients, peer nodes in the blockchain network is called actors. I might use actor, participant, and member alternately in the remaining of the paper. The client application, *e.g.*, A_1, A_2, is an interface a client interacts with the chaincode (*i.e.*, the software that defines assets in the blockchain and the transaction instructions to modify or query the status of the asset). It is responsible for submitting a transaction proposal, collecting responses, assembling a formal transaction and responding to events. There are two types of peer nodes. One is the endorsing peer

and the other is the validating peer. The endorsing peer is responsible for executing the transaction proposals received from the client applications and sending back the executed results. Only when the client application receives responses from a set of endorsing peers, the validating peers validate the transactions according to the endorsing policy before updating their locally stored ledger. Usually an endorsing peer is at the same time a validating peer.

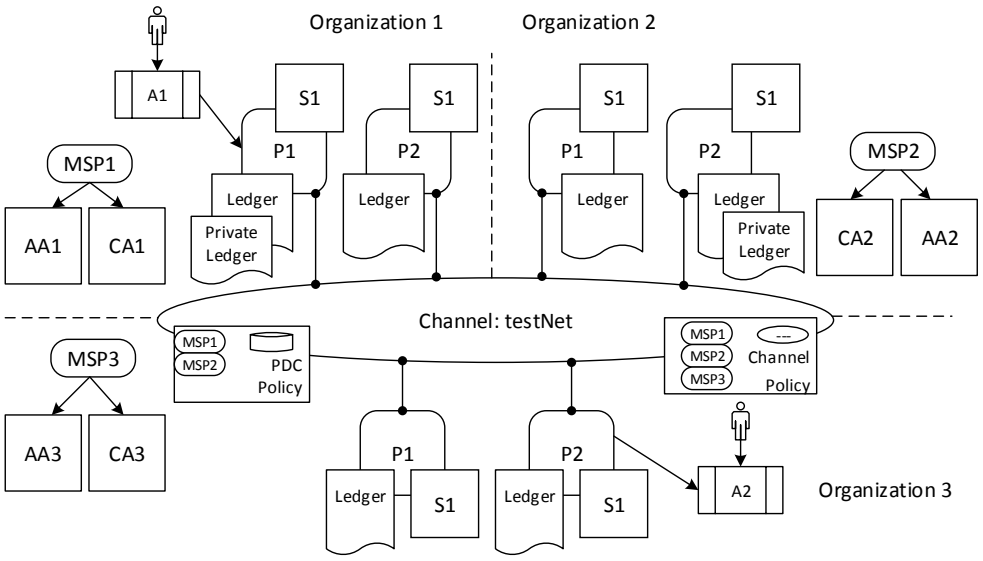

Figure 7.1: System Framework

For the channel *testNet* with channel policy that only members from the aforementioned three organizations are permitted to join the channel, this subset of members will create a ledger (denoted by *Ledger* on this peer node P_i) that is separated from the other participants. Logically, all the peers will maintain a same copy of the ledger, which is achieved by way of a consensus protocol. To deal with scenarios where a subset of organizations on a channel need to keep data private from the others, *e.g.*, in the figure, *Organization 1* and *Organization 2* wants to keep data private from *Organization 3*, they will build a private data collection (dented by PDC in the fig-

ure). On the basis of a PDC policy that only members from *Organization 1* and *Organization 2* are able to endorse, commit or query the private data.

For permissionless blockchain systems, there have to be a mechanism for membership management. In the proposed system framework, I merge the attribute management into the existing identity or certificate management infrastructure, namely MSP (*i.e.*, membership service provider). In the existing Fabric framework, the MSP's functionality is to define the governing rules of valid identities for an organization, channel, or blockchain network. In the proposed framework, the MSP will also govern rules of valid attributes for members who can look into the details of private transactions.

7.1.1 Terminology Description

- Chaincode: chaincode (*i.e.*, smart contract) is the software that defines objects (which is denoted by a key-value pair) and the operations to modify the status of the objects. In Hyperledger, the objects are named assets and the operations are called transaction instructions.

- Network: The network is the infrastructure that provides the distributed ledger and the chaincode as well as its API to the client applications. The network is a peer-to-peer network with nodes called peers.

- Channel: channel is a mechanism allowing a subset of participants to create a ledger that is separated from the other participants. One channel corresponds to one ledger.

- Ledger: the copy of ledger located on each peer is the physical ledger, which consists of key-value pairs. The ledger which is on the right-most side of the figure is the logical view of the ledger. It represents that all the peers will main-

tain a same copy of the ledger by way of a consensus protocol. In Hyperledger Fabric, the consensus algorithm is a pluggable module, where an orderer will collect all the transactions and send them in ordered blocks.

- Client: client application is the interface a client interacts with the chaincode. It is responsible for submitting a transaction proposal, collecting the responses, assembling a formal transaction and responding to events.

- Peer: there are two types of peer nodes. One is the endorsing peer and the other is the validating peer. The endorsing peer is responsible for executing the transaction proposals received from the client applications and sending back the executed results. The client application receives responses from a set of endorsing peers (the group of endorsing peers satisfy an endorsing policy defined by a boolean formula). The validating peers validate the transactions according to the endorsing policy before updating their locally stored ledger. Usually an endorsing peer is at the same time a validating peer.

- Orderer: orderers are also named as ordering service nodes. These nodes form the ordering service of Hyperledger. They work on establishing the total order of all the transactions within the system.

All the participants above are named an actor in Hyperledger. I might use actor, participant and members exchangeably in the remaining of this paper.

7.1.2 Identity Management

Unlike permissionless blockchain platforms where trust is established by a resource-consuming consensus protocol, in permissioned blockchain trust is established by identifying all the participants including the clients, peer nodes, the orderers.

- Identities: in Hyperledger, a digital identity is encapsulated in an X.509 digital certificate. The identity of a participant determines what resources it can access.
- Attributes: properties of an actor, for example, its organization, organizational unit, role, profession, etc. Identity and attribute are unified to be principal in Hyperledger.
- Membership Service Provider (MSP): MSP's functionality is to define the governing rules of valid identities for an organization, channel, or blockchain network. In another word, MSP converts actor's verifiable identities (X.509 certificate) into roles (or members).
- Organization: an organization is defined to be a managed group of members. The concept of organization here is different from that in an X.509 certificate. Each organization will have a local MSP for membership and access control management.
- Membership Service: it assigns identifiers for both peers and client applications. In the system, it is composed of two parts. The first part works on certificate management in PKI and the second part is on attribute management, such as attribute assignment, validation and attribute-based encryption related infrastructures.
- Auditors: auditors are responsible for system logs.

7.1.3 System Workflow

Before digging into detailed design of data protection mechanism, I firstly introduce how the data flow among the multiple system components.

I divive the workflow into two phases, *i.e.*, Proposal & Packaging and Validation & Updating.

Proposal & Packaging

1. A client submits a transaction proposal to the endorsing peers.
2. The endorsing peers execute the code associated to the transaction, generate the read-write set and sign the result.
3. The signed result is sent back to the client.
4. The client collects a sufficient number of endorsements so that the endorsement policy can be satisfied.
5. The client broadcasts the transaction to the orderers.
6. The orderers collect transactions, order them chronologically and create a signed block.

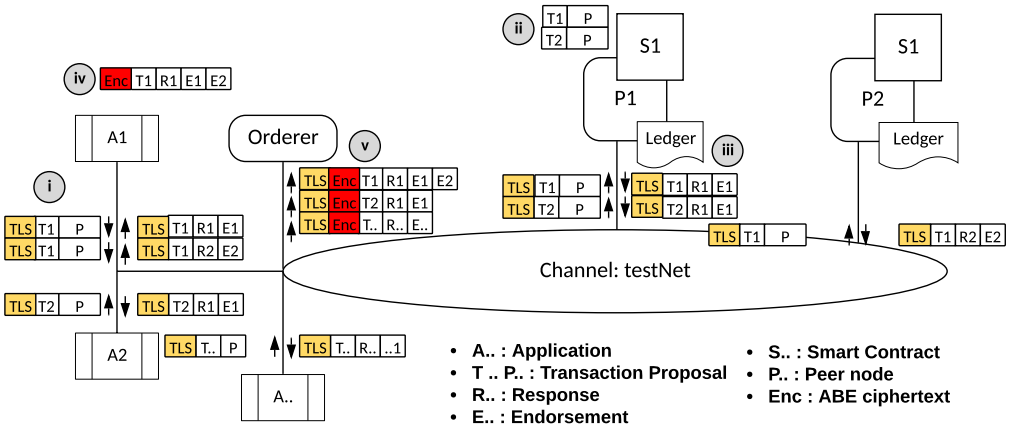

Figure 7.2: Transaction Workflow1: Proposal and Packaging

Validation & Updating

1. The orderers distribute the new block to all the peers (including both endorsing and committing peers).

2. The endorsing and committing peers execute the validation of the transactions contained in the received block; they do two verifications: first, if the read-write set is coherent with the current state of the ledger; second, if the endorsement policy has been satisfied.

3. The client is notified of the result of execution of the transaction.

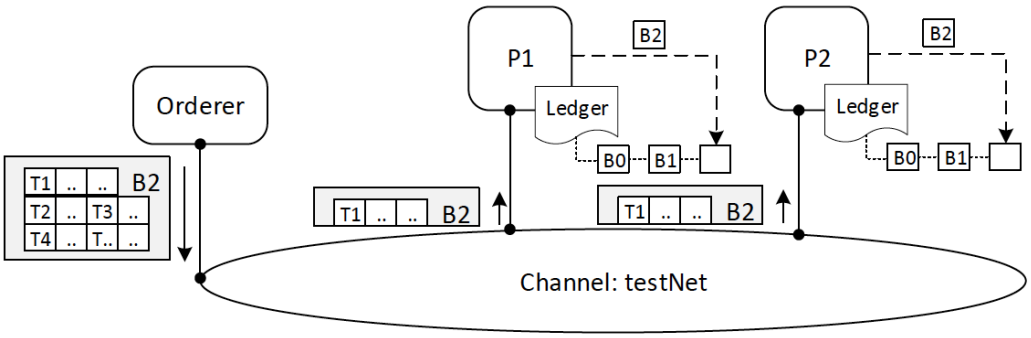

Figure 7.3: Transaction Workflow2: Validation and Updating

The transaction described above include both the deployment transaction and the invocation transaction. The deployment transaction will be submitted at the chaincode deployment phase. This transaction will be submitted only once. After the chaincode is installed and initialized on the peer nodes, client applications invoke the functions by constructing and submitting invocation transactions. Without privacy protection, the transactions will be public to all the members on the channel. Access control should be enforced when sending chaincode query requests. The goal is that only authorized users are able to access private information, *e.g.*, invoked chaincode function, arguments, and the responses.

7.2 Trust Framework

In this section, I summarize the trust framework of Hyperledger Fabric and add the attribute-based encryption related modules into the framework.

Based on its original workflow, I show how the added modules *AVC* and *KGC* could work together.

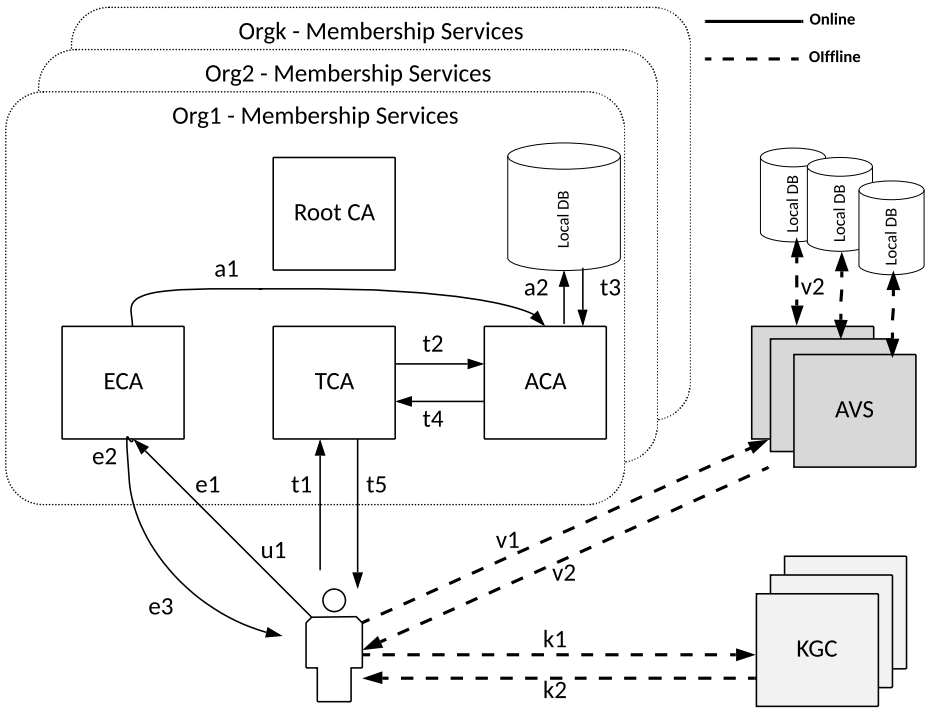

Figure 7.4: Trust Framework in Hyperledger Fabric

To create an access control policy in Hyperledger Fabric, the most important part is user's identities and how they are managed in the system, *i.e.*, the trust framework in Hyperledger Fabric. In Hyperledger Fabric, the trust in built on public key infrastructure. Each organization might have their certificate authority to distribute certificates for their managed members. As shown in Figure 7.4, there are three types of certificate authorities, *i.e.*, Enrollment Certificate Authority (ECA), Transaction Certificate Authority (TCA) and Attribute Certificate Authority (ACA).

- ECA: ECA is responsible for new user registration. During the registration, ECA will generate a certificate for a user. The certificate includes user's identity.

- TCA: To achieve transaction anonymity and unlinkability, Hyperledger Fabric also provides the TCA for deploying and invoking chaincode. Each user can request multiple transaction certificates. The transaction certificate is independent of user's identity information, thus preserving user's privacy.

- ACA: It is responsible for certifying users' owernship of attributes and maintaining a database for all the users and their assigned attributes, including ID, Affiliation, Attribute Name/Value, and the valid time period.

Besides the aforementioned three certificate authorities, in the proposed framework I also introduce two components for attribute-based encryption, *i.e.*, Key Generation Center (KGC) and Attribute Validation Center (AVC).

- KGC: this component is responsible for generating and publishing crypto algorithm related parameters, and creating attribute private key for a user.

- AVC: the Attribute Authority component takes the responsibility of users' attribute validation and endorsement and distribute certificate for users to prove their ownership of attributes.

7.3 Attribute-Based Access Control in Hyperledger Fabric

Figure 7.4 summarizes the workflow of how the identity and attribute management system distributes certificates and private attribute keys to members. This procedure can be split into three phases, *i.e.*, enrollment certificate distribution, transaction certificate distribution and private attribute key distribution.

Attribute Certificate Distribution

- **v1**: the user sends request for attribute certificates. The request includes the materials that could be used to prove the identity, attributes of the user.
- **v2**: the attribute validation centers generate attribute certificates for the user and send them back the user.

Enrollment Certificate Distribution

- **e1**: User → ECA, a user sends an enrollment certificate request to the enrollment certificate authority.
- **e2**: the enrollment certificate authority generates an enrollment certificate for the user.
- **e3**: ECA → User, the ECA sends the created certificate to the user.
- **a1**: ECA → ACA, after distributing the enrollment certificate to the user, ECA will send the certificate to the attribute certificate authority.
- **a2**: ACA will update the database of user identity and attributes storage.

Transaction Certificate Distribution

- **t1**: User → TCA, the user requests a batch of transaction certificates based on its application requirements. The request will include the number of desired transaction certificates, the assigned enrollment certificate, the attribute names to be used.
- **t2**: TCA → ACA, upon receiving the request from the user, the transaction certificate authority will generate a signed request which includes the user's

enrollment certificate and attribute names and send the request to the attribute certificate authority.

- **t3**: ACA will query the database and check whether the user is assigned the requested attributes. If the user has part or all of the requested attributes, the attribute certificate authority will generate an attribute certificate which includes the user's attributes name and corresponding value as well as the enrollment certificate. Otherwise, ACA will generate an error message.

- **t4**: ACA $\rightarrow$ TCA, based on the query result of the user's attributes, ACA sends either an attribute certificate or an error message to the transaction certificate.

- **t5**: TCA $\rightarrow$ User, the transaction certificate authority checks whether the attributes contained in the attribute certificate is valid or not and sends the generated transaction certificates to the user.

Private Attribute Key Distribution

- **k1**: User $\rightarrow$ KGC, the user with the transaction certificate sends the private attribute key request to the key generation center. The request includes both the user's identity, attributes, as well as the attribute certificates obtained from the AVC.

- **k2**: KGC $\rightarrow$ User, the key generation center verifies the user's attribute certificate and generates the private attribute keys for the user and sends them back to the user.

Note 1. I will not change the workflow marked by a_i, e_i and t_i as designed by the Hyperledger Fabric.

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            37:3e:ca:f7:da:34:25:40:bd:ab:84:f1:46:ee:b8:78:37:67:e8:68
    Signature Algorithm: ecdsa-with-SHA256
        Issuer: C=US, ST=California, L=San Francisco, O=org1.example.com, CN=ca.org1.example.com
        Validity
            Not Before: Oct  6 22:57:00 2019 GMT
            Not After : Oct  5 23:02:00 2020 GMT
        Subject: OU=client, OU=org1, OU=department1, CN=user1
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:59:fa:22:9c:7b:72:28:a4:06:7c:7c:0e:a4:d0:
                    0d:03:42:87:d6:02:c3:8c:a1:1e:b8:47:2b:c1:a7:
                    40:7a:58:6a:89:c6:47:b9:72:1b:97:17:df:91:73:
                    58:8e:cc:19:9b:e3:b8:3d:81:6a:75:e9:f6:c4:f2:
                    fc:6d:ce:9f:16
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature
            X509v3 Basic Constraints: critical
                CA:FALSE
            X509v3 Subject Key Identifier:
                35:1F:A7:86:75:59:B5:6B:E8:16:33:D0:69:0F:0D:90:A4:A2:07:33
            X509v3 Authority Key Identifier:
                keyid:42:39:AA:0D:CD:76:DA:EE:B8:BA:0C:DA:70:18:51:D1:45:04:D3:1A:AD:1B:2D:DD:DB:AC:6A:57:36:5E:49:7C

            1.2.3.4.5.6.7.8.1:
                {"attrs":{"carmake":"Toyota"}}
    Signature Algorithm: ecdsa-with-SHA256
         30:44:02:20:6f:10:e0:05:1b:28:90:bf:c0:b3:1d:e0:d2:4f:
         d1:1f:d8:d4:9a:49:a4:67:41:84:2c:ba:bc:77:5d:b7:c6:2b:
         02:20:3d:09:d7:0c:97:d0:6a:43:bd:4f:98:86:5d:29:63:9d:
         f2:12:3d:c2:97:71:82:a1:84:3e:7f:26:bf:f5:15:ab
-----BEGIN CERTIFICATE-----
MIICUDCCAfegAwIBAgIUNz7K99o0JUC9q4TxRu64eDdn6GgwCgYIKoZIzj0EAwIw
czELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNh
biBGcmFuY2lzY28xGTAXBgNVBAoTEG9yZzEuZXhhbXBsZS5jb20xHDAaBgNVBAMT
E2NhLm9yZzEuZXhhbXBsZS5jb20wHhcNMTkxMDA2MjI1NzAwWhcNMjAxMDA1MjMw
MjAwWjBCMTAwDQYDVQQLEwZjbGllbnQwCwYDVQQLEwRvcmcxMBIGA1UECxMLZGVw
YXJ0bWVudDExDjAMBgNVBAMTBXVzZXIxMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcD
QgAEWfoinHtyKKQGfHwOpNANA0KH1qLDjKEeuEcrwadAelhqicZHuXlblxffkXNY
jswZm+O4PYFqden2xPL8bc6fFqOBmTCBljAOBgNVHQ8BAf8EBAMCB4AwDAYDVR0T
AQH/BAIwADAdBgNVHQ4EFgQUNR+nhnVZtWvoFjPQaQ8NkKSiBzMwKwYDVR0jBCQw
IoAgQjmqDc122u64ugzacBhR0UUE0xqtGv3d26xqVzZeSXwwKgYIKgMEBQYHCAEE
HnsiYXR0cnMiOnsiY2FybWFrZSI6IlRveW90YSJ9fTAKBggqhkjOPQQDAgNHADBE
AiBvEOAFGyiQv8CzHeDST9Ef2NSaSaRnQYQsurx3XbfGKwIgPQnXDJfQakO9T5iG
XSljnfISPcKXcYKhhD5/Jr/1Fas=
-----END CERTIFICATE-----
```

Figure 7.5: An Example Enrollment Certificate

The blockchain platform itself provides a way to implement attribute-based access control [1]. The access control decisions are made by the chaincode on the basis of an

[1]https://tinyurl.com/y6a5uyju

identity's attributes. To enable this, the platform proposed to add one or more attribute name and value into an identity's enrollment certificate. When this identity is used to call functions in the chaincode, the attribute's value will be extracted to make the attribute-based access control decision. As shown in Figure 7.5, the common name of the certificate is the example user "user1". It was assigned an attribute "carmake" with the value "Toyota". Within the chaincode, the following Figure 7.6 shows how access control logic is added.

```
func (s *SmartContract) queryAllCars(APIstub shim.ChaincodeStubInterface) sc.Response {

    // Check GetAttributeValue
    carmake, makefound, err := cid.GetAttributeValue(APIstub, "carmake")
    if err != nil {
        return shim.Error(err.Error())
    }

    fmt.Println("Car Make found : ", makefound)
    fmt.Println("Car Make is : ", carmake)

    // Check AssertAttibuteValue
    makeErr := cid.AssertAttributeValue(APIstub, "carmake", "Toyota")
    if makeErr != nil {
        return shim.Error("Assert error")
    }
```

Figure 7.6: Attribute-Based Access Control in Hyperledger Fabric

The mechanism implemented in Hyperledger Fabric can only achieve attribute-based access control on the chaincode level. All the car owners or data uploaders will use the same access policy. In this example is that it is only the identity with the value of the attribute "carmaker" to be "Toyota". The example user "user1" could query the API in the chaincode. What if a new user joins the system and wants a new access policy, such as the value of the attribute "carmaker" is "Ford"? Within the current attribute-based access control mechanism in Hyperledger Fabric, it is very difficult to change if not impossible.

7.4 ABE-based ABAC in Hyperledger Fabric

To enable discretionary ABAC in Hyperledger Fabric, I propose to integerate ABE into the blockchain platform. The following summarizes the implementations.

7.4.1 ABE Toolkit

I implement the ABE scheme proposed in Chapter 5. The provided APIs include: global parameter generation, federated setup, federated key generation, delegated key generation, encryption, and decryption. Since the chaincode is written in multiple programming languages, I implement a local agent for dealing with the programming language gap issue.

7.4.2 Hybrid Encryption

To enable hyprid encryption, I provided APIs for encrypting any string, which includes the symmetric key of AES, password, or any secret messages. To enable file encryption, I implement a file encryption/decryption toolkit. The ciphertext format would be, the ABE will be used to encrypt the password or symmetric key, and then the password or symmetric key will be used to encrypt a longer message or files.

7.4.3 Attribute Validation Authority

I selected the signature scheme in Boneh *et al.* (2001, 2003) to implement the attribute authority. The following is the description of the scheme.

- **Key Generate**: Run on the attribute authority end. For a particular attribute authority, select randomly $x \leftarrow \mathbb{Z}_p$, and compute $v \leftarrow g^x$. The AA's public key is $v \in G$. The attribute authority's secret key is $x \in \mathbb{Z}_p$.

- **Sign**: For a particular attribute authority, given the secret key $x \in \mathbb{Z}_p$ and a message $M \in \{0,1\}^*$, compute $h \leftarrow H(M)$, where $h \in G$, and $\sigma \leftarrow h^x$. The signature is $\sigma \in G$. H denotes a hash function which maps an arbitrary binary message to an element in G.

- **Verify**: Given an attribute authority's public key v, a message M, and a signature σ, compute $h \leftarrow H(M)$; the verification succeeds if the equation $e(g, \sigma) = e(v, h)$ holds;

The implemented attribute authority provides services in the way of restful API. An example request is in the following format.

{
"aaID":"1",
"userGID":"qdong11",
"attributes":"attr3,attr4"
}

7.4.4 ABE in Chaincode

Figure 7.7 describes the workflow of using attribute-based encryption to protect sensitive data stored in the blockchain applications.

- Step 1: The user request attribute certificates from multiple attribute authorities. The certificates associates the user's identifier with its attributes.

- Step 2: The user sends the private attribute request to the key generator. The request includes the attribute certificates of the user. Note that the key generation center is just an abstract description. It could be in a hierarchical structure as described in Chapter 5.

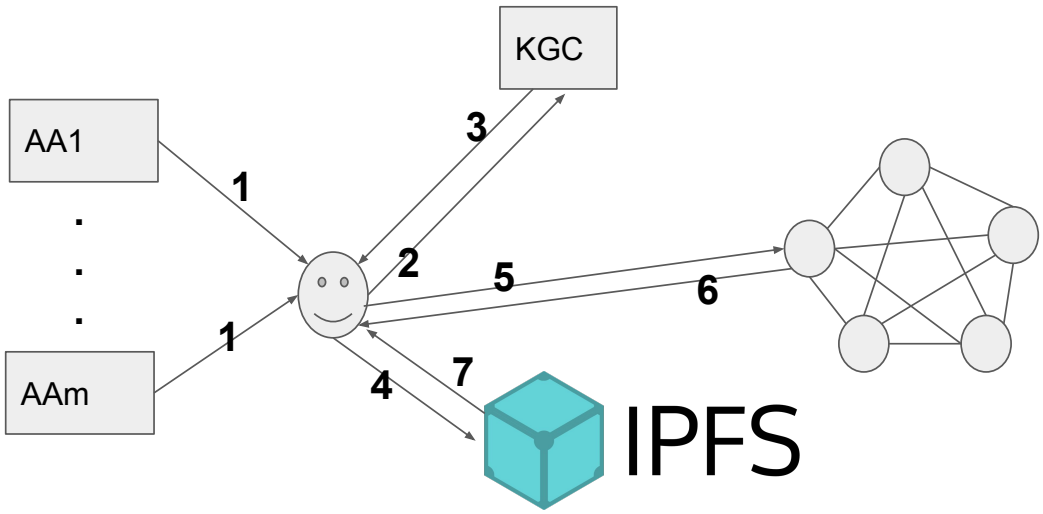

Figure 7.7: Workflow of the Demo System

- Step 3: The KGC will verify the use's attributes. If the verification succeeded, the corresponding private attribute key will be distributed to the user. Otherwise, the request will fail.

- Step 4: As a data owner, to upload the sensitive file. At first, the file needs to be encrypted and uploaded to the IPFS Benet (2014) to get the hash value which could be used to download the file. Then the data owner will run the ABE encryption scheme to encrypt both the symmetric key or password used to encrypt the file and the hash value.

- Step 5: upload the ABE ciphertexts and the other metadata related with the file to the blockchain application.

- Step 6: Whenever a user searched the blockchain to find interested file, the chaincode will return the non-sensitive metadata as well as the ABE ciphertext. The user will at first check whether its attributes, identity satisfy the access policy enforced over the data, and then decrypt the ABE ciphertext to get the hash value and password or symmetric key.

- Step 7: The user will download the file from the IPFS file system by using the decrypted hash value. Then the user decrypts the file by running file decryption toolkit with the encrypted file and the decrypted password or symmetric key as inputs. Finally, the user gets access to the sensitive data.

I modified the chaincode Fabcar used in the Hyperledger Fabric to demonstrate attribute-based access control to show how ABE is used. The Fabcar application uses Hyperledger Fabric to store users' car information, including car color, car making company, *etc.*. I change the the field owner to include sensitive information about the car's owner or the car itself. I assume the sensitive information is stored in a file system named IPFS Benet (2014). When uploading the file to the system, a hash value will be generated. In the later time, a data user with the hash value could download the file.

To protect the sensitive file, the ABE scheme needs to provide protection over both the file's hash value and the symmetric key or passed used to encrypt the file. Therefore, when the user call the chaincode API to upload sensitive data, before uploading, the data would be encrypted. The following is an example file encryption request and the generated ciphertext.

```
const enc_request_data =
{
   "password": "$ASU+ENC_FILE_PASSWORD",
   "ipfs_hash": "Qmd286K6pohQcTKYqnS1YhWrCiS4gz7Xi34sdwMe9USZ7u",
   "access_policy": "((attr1 and attr2) or (attr2 and attr3) or (attr3 and attr4) or (attr5 and attr6))",
   "revoked_users": ["ru1", "ru2", "ru3"],
   "non_revoked_authorities": ["org1", "org2", "org3"],
   "abe_ciphertext_file": "ctxt.json"
}
```

Figure 7.8: Example File Encryption Request

Compared with the previous data uploading, this time it is the ABE ciphertext being uploaded but not the plaintext message.

Figure 7.9: Example File Encryption ABE Ciphertext (Some Information Hidden for Space Limitation)

```
// Submit the specified transaction.
// createCar transaction - requires 5 argument, ex: ('createCar', 'CAR12', 'Honda', 'Accord', 'Black', 'Tom')
await contract.submitTransaction('createCar', 'CAR12', 'Honda', 'Accord', 'Black', JSON.stringify(abe_ciphertext));
```

Figure 7.10: Upload the ABE Protected Sensitive Message to Blockchain

When the user queries the data, it will be capable to download the data and then decrypt locally by calling the file decryption API.

7.5 Summary

In this chapter, I investigate the workflow of Hyperledger Fabric, how it achieves the goal of permissioned blockchain, how it performs the attribute-based access control model within chaincode. I find the access control is chaincode-level and is too coarse. Therefore, I propose to add new modules into the trust framework of Hyperledger Fabric to enable ABE-based ABAC. I do implementation of attribute au-

thorities, message/file encryption/decryption toolkit, ABE toolkit and also show the whole workflow by an example chaincode application Fabcar.

Chapter 8

CONCLUSION AND FUTURE WORK

In this thesis, I propose three CP-ABE schemes. The first CP-ABE scheme is named DUR-CP-ABE, that is discretionary user revocation ciphertext-policy attribute-based encryption scheme. I identify each user by way of a hierarchical identity, which includes information of the hierarchical organization. In this way, to revoke a group of users affiliated with an organization, I could revoke the organization directly without revoking all the users individually, thus reducing both computation and communication overheads. The second scheme expands the DUR-CP-ABE in terms of key management. It provides new features including federation, delegation and inter-operability. The federation solves the problem of single point of trust and failure issue. No single trusted authority owns the master secret key, while multiple trusted authorities have to work together in a federated way to generate the public parameters. Delegation enables the key generation to follow the organization's layered management structure. Inter-operability enables a user to get private key from different subdivisions.

To enable attribute expiration, I propose an ABE scheme that supports automatic attribute expiration. Different from the existing solutions where interactions are needed, the proposed scheme is non-interactive. The scheme provides the features including decentralization, automatic attribute expiration and temporal delegation. By decentralization, a user could get attributes from multiple authorities. By automatic attribute expiration, I embed a time period for each attribute assigned to the user, thus only when the attributes satisfy the access policy and do not expire when the encrypted data is shared can the user decrypt the ciphertext.

Considering that most existing work on applying attribute-based encryption is on cloud computing, I propose a framework for using attribute-based encryption in hyperledger fabric. I implement an attribute authority for distributing attributes for users. I implement the attribute-based encryption toolkit for the whole system setup, private key generation, data encryption and data decryption. I implement a file encryption/decryption toolkit which could work together with attribute-based encryption to perform hybrid encryption.

With the proposed attribute-based encryption enabled attribute-based access control model, fine-grained discretionary access control could be achieved but not the chaincode-level coarse access control. To demonstrate the whole workflow of using attribute-based encryption in hyperledger fabric, I set up a simulated hyperledger fabric network, modified a Fabcar chaincode which is used in fabric's documentation to show attribute-based access control. The demo showed the workflow of how a user gets attribute certificates from multiple attribute authorities, how the key generator verifies the attributes and generates the private key for the user when the validation succeeded, how a data owner uploaded encrypted secrets (which will be used to decrypt the sensitive file), how a user query the desired data and get the ciphertext, perform decryption to get the secret, and finally get access to the sensitive file.

There are some very interesting future work. ABE schemes are constructed based on pairing, which makes it less efficient compared with other public key encryption schemes used in practice. When more functionalities are added into the scheme, efficiency becomes even worse. Therefore, it is very important to further investigate how to optimize the scheme either in terms of implementation or algorithm design.

Attribute-based encryption is a special category of functional encryption Boneh *et al.* (2011). The function in attribute-based encryption is to evaluate whether a set of attributes associated with a user's private key satisfy the access policy described

by a boolean formula associated with the ciphertext. If the attributes satisfy the access policy, the output will be the plaintext message. With functional encryption, the user will be assigned a private key based on the function, say f that he wants to evaluate over the message. The decrypted plaintext is not the message but $f(x)$. If f is a predicate function, one application could be intrusion detection over encrypted data. The network payloads might be very sensitive and could not be shared with other entities, functional encryption enables data user to get known of the checking result but not the data themselves. This might also be used in medical applications, such as gene testing.

With attribute-based encryption, any user is allowed to be a data sender. This makes it possible for some users to send malicious data to the receiver. Therefore, it is significant to verify the identity of a data sender. Existing Attribute-Based Signature Li *et al.* (2010) could be used to perform attribute-based authentication where a user who was assigned a set of attributes is capable to prove the ownership of these attributes. In some application scenarios, not only a user's attributes but also its identity needs to be verified before decrypting the data sent by this user. It is interesting to construct a scheme supporting attribute-based identification, authentication and authorization all in one.

REFERENCES

Akinyele, J. A., C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green and A. D. Rubin, "Charm: a framework for rapidly prototyping cryptosystems", Journal of Cryptographic Engineering **3**, 2, 111–128 (2013).

Akinyele, J. A., M. W. Pagano, M. D. Green, C. U. Lehmann, Z. N. Peterson and A. D. Rubin, "Securing electronic medical records using attribute-based encryption on mobile devices", in "Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices", pp. 75–86 (ACM, 2011).

Attrapadung, N. and H. Imai, "Attribute-based encryption supporting direct/indirect revocation modes", in "IMA International Conference on Cryptography and Coding", pp. 278–300 (Springer, 2009).

Beimel, A., *Secure schemes for secret sharing and key distribution* (Technion-Israel Institute of technology, Faculty of computer science, 1996).

Benet, J., "Ipfs-content addressed, versioned, p2p file system", arXiv preprint arXiv:1407.3561 (2014).

Bethencourt, J., A. Sahai and B. Waters, "Ciphertext-policy attribute-based encryption", in "2007 IEEE symposium on security and privacy (SP'07)", pp. 321–334 (IEEE, 2007a).

Bethencourt, J., A. Sahai and B. Waters, "Ciphertext-policy attribute-based encryption", in "IEEE SP'07", pp. 321–334 (Oakland, CA, 2007b).

Boldyreva, A., V. Goyal and V. Kumar, "Identity-based encryption with efficient revocation", in "Proceedings of the 15th ACM conference on Computer and communications security", pp. 417–426 (ACM, 2008).

Boneh, D. and X. Boyen, "Efficient selective-id secure identity-based encryption without random oracles", in "International conference on the theory and applications of cryptographic techniques", pp. 223–238 (Springer, 2004).

Boneh, D. and M. Franklin, "Identity-based encryption from the weil pairing", in "Annual international cryptology conference", pp. 213–229 (Springer, 2001).

Boneh, D., C. Gentry, B. Lynn and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps", in "International Conference on the Theory and Applications of Cryptographic Techniques", pp. 416–432 (Springer, 2003).

Boneh, D., B. Lynn and H. Shacham, "Short signatures from the weil pairing", in "International Conference on the Theory and Application of Cryptology and Information Security", pp. 514–532 (Springer, 2001).

Boneh, D., A. Sahai and B. Waters, "Functional encryption: Definitions and challenges", in "Theory of Cryptography Conference", pp. 253–273 (Springer, 2011).

Chen, J., H. W. Lim, S. Ling, H. Wang and K. Nguyen, "Revocable identity-based encryption from lattices", in "Information Security and Privacy", pp. 390–403 (Springer, 2012).

Cocks, C., "An identity based encryption scheme based on quadratic residues", in "IMA International Conference on Cryptography and Coding", pp. 360–363 (Springer, 2001).

Dong, Q., D. Huang, J. Luo and M. Kang, "Achieving fine-grained access control with discretionary user revocation over cloud data", in "2018 IEEE Conference on Communications and Network Security (CNS)", pp. 1–9 (IEEE, 2018).

Fiat, A. and M. Naor, "Broadcast encryption", in "Annual International Cryptology Conference", pp. 480–491 (Springer, 1993).

Gervais, A., G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf and S. Capkun, "On the security and performance of proof of work blockchains", in "Proceedings of the 2016 ACM SIGSAC conference on computer and communications security", pp. 3–16 (ACM, 2016).

Goyal, V., O. Pandey, A. Sahai and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data", in "ACM Conference on Computer and Communications Security", pp. 89–98 (Alexandria, Virginia, USA, 2006).

Harris, S. and F. Maymi, *CISSP All-in-One Exam Guide, Seventh Edition* (McGraw-Hill Education Group, 2016), 7th edn.

HIPAA, "Individuals' Right under HIPAA to Access their Health Information 45 CFR § 164.524", `https://tinyurl.com/jhhyk6n` (2017).

Hu, V. C., D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, K. Scarfone *et al.*, "Guide to attribute based access control (abac) definition and considerations (draft)", NIST special publication **800**, 162 (2013).

Hu, V. C., D. R. Kuhn, D. F. Ferraiolo and J. Voas, "Attribute-based access control", Computer **48**, 2, 85–88 (2015).

Hunkeler, U., H. L. Truong and A. Stanford-Clark, "Mqtt-s—a publish/subscribe protocol for wireless sensor networks", in "Communication systems software and middleware and workshops, 2008. comsware 2008. 3rd international conference on", pp. 791–798 (IEEE, 2008).

Hur, J. and D. K. Noh, "Attribute-based access control with efficient revocation in data outsourcing systems", IEEE Transactions on Parallel and Distributed Systems **22**, 7, 1214–1221 (2011).

Jahid, S., P. Mittal and N. Borisov, "Easier: Encryption-based access control in social networks with efficient revocation", in "Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security", pp. 411–415 (ACM, 2011).

Lai, J., R. H. Deng, C. Guan and J. Weng, "Attribute-based encryption with verifiable outsourced decryption", IEEE Transactions on Information Forensics and Security **8**, 8, 1343–1354 (2013).

Lewko, A., A. Sahai and B. Waters, "Revocation systems with very small private keys", in "Security and Privacy (SP), 2010 IEEE Symposium on", pp. 273–285 (IEEE, 2010).

Lewko, A. and B. Waters, "Decentralizing attribute-based encryption", in "Annual international conference on the theory and applications of cryptographic techniques", pp. 568–588 (Springer, 2011).

Li, B., A. P. Verleker, D. Huang, Z. Wang and Y. Zhu, "Attribute-based access control for icn naming scheme", in "Communications and Network Security (CNS), 2014 IEEE Conference on", pp. 391–399 (2014).

Li, B., Z. Wang and D. Huang, "An efficient and anonymous attribute-based group setup scheme", in "2013 IEEE Global Communications Conference (GLOBECOM)", pp. 861–866 (2013).

Li, J., M. H. Au, W. Susilo, D. Xie and K. Ren, "Attribute-based signature and its applications", in "Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security", pp. 60–69 (ACM, 2010).

Li, J., J. Li, X. Chen, C. Jia and W. Lou, "Identity-based encryption with outsourced revocation in cloud computing", Computers, IEEE Transactions on **64**, 2, 425–437 (2015).

Libert, B. and D. Vergnaud, "Adaptive-id secure revocable identity-based encryption", in "Topics in Cryptology–CT-RSA 2009", pp. 1–15 (Springer, 2009).

Lynn, B., "The pairing-based cryptography library", URL `https://crypto.stanford.edu/pbc/` (2006).

Nakamoto, S., "Bitcoin: A peer-to-peer electronic cash system", (2008).

Naor, D., M. Naor and J. Lotspiech, "Revocation and tracing schemes for stateless receivers", in "Advances in Cryptology—CRYPTO 2001", pp. 41–62 (Springer, 2001).

Sahai, A. and B. Waters, "Fuzzy identity-based encryption", in "Annual International Conference on the Theory and Applications of Cryptographic Techniques", pp. 457–473 (Springer, 2005).

Semiconductors, P., "hyperledger-fabricdocs documentation", available at `https://tinyurl.com/yjj83gf5` (2019).

Shamir, A., "Identity-based cryptosystems and signature schemes", in "Workshop on the theory and application of cryptographic techniques", pp. 47–53 (Springer, 1984).

Waters, B., "Efficient identity-based encryption without random oracles", in "Annual International Conference on the Theory and Applications of Cryptographic Techniques", pp. 114–127 (Springer, 2005).

Waters, B., "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization", in "Public Key Cryptography - PKC 2011", pp. 53–70 (Springer-Verlag, 2011).

Wood, G., "Ethereum: A secure decentralised generalised transaction ledger", Ethereum project yellow paper **151**, 1–32 (2014).

Yamada, S., N. Attrapadung, G. Hanaoka and N. Kunihiro, "A framework and compact constructions for non-monotonic attribute-based encryption", in "International Workshop on Public Key Cryptography", pp. 275–292 (Springer, 2014).

Yang, K., X. Jia and K. Ren, "Attribute-based fine-grained access control with efficient revocation in cloud storage systems", in "Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security", pp. 523–528 (ACM, 2013).

Yu, S., K. Ren and W. Lou, "Attribute-based on-demand multicast group setup with membership anonymity", Computer Networks **54**, 3, 377–386 (2010).

Zhu, Y., H. Hu, G.-J. Ahn, M. Yu and H. Zhao, "Comparison-based encryption for fine-grained access control in clouds", in "ACM conference on Data and Application Security and Privacy", CODASPY '12, pp. 105–116 (San Antonio, Texas, USA, 2012).