Efficient Ciphertext-policy Attribute Based Encryption for Cloud-Based Access Control

Ruqayah Rabeea Hashim Al-Dahhan

A Thesis Submitted in Partial Fulfilment of the Requirements of Liverpool John Moores University for the Degree of Doctor of Philosophy

May 2019

Supervisors Certification

We certify that the thesis entitled “**Efficient Ciphertext-policy Attribute Based Encryption for Cloud-Based Access Control**" was prepared under our supervision at the Department of Computer Science, Faculty of Engineering and Technology, Liverpool John Moores University as a partial of fulfilment of the requirements of Liverpool John Moores University for the degree of Doctor of Philosophy.

Name	**Prof. Qi Shi**
Title	Director of study
Address	James Parsons Building, 3 Byrom St, Liverpool L3 3AF, UK.

Name	**Dr. Gyu Myoung Lee**
Title	Second Supervisor
Address	James Parsons Building, 3 Byrom St, Liverpool L3 3AF, UK.

Name	**Dr. Kashif Kifaya**
Title	Third Supervisor
Address	Computer Science Department, Air University, Islamabad, Pakistan.

Signed (Director of Studies):

Date:

Abstract

Outsourcing data to some cloud servers enables a massive, flexible usage of cloud computing resources and it is typically held by different organizations and data owners. However, various security concerns have been raised due to hosting sensitive data on an untrusted cloud environment, and the control over such data by their owners is lost after uploading to the cloud. Access control is the first defensive line that forbids unauthorized access to the stored data. Moreover, fine-grained access control on the untrusted cloud can be enforced using advanced cryptographic mechanisms. Some schemes have been proposed to deliver such access control using Ciphertext-policy attribute based encryption (CP-ABE) that can enforce data owners' access policies to achieve such cryptographic access control and tackle the majority of those concerns. However, some challenges are still outstanding due to the complexity of frequently changing the cryptographic enforcements of the owners' access policies in the hosted cloud data files, which poses computational and communicational overheads to data owners. These challenges are: 1) making dynamic decisions to grant access rights to the cloud resources, 2) solving the issue of the revocation process that is considered as a performance killer, and 3) building a collusion resistant system. The aim of our work is to construct an access control scheme that provides secure storing and sharing sensitive data on the cloud and suits limited-resources devices.

In this thesis, we analyse some of the existing, related issues and propose a scheme that extends the relevant existing techniques to resolve the inherent problems in CP-ABE without incurring heavy computation overhead. In particular, most existing revocation techniques require re-issuing many private keys for all non-revoked users as well as re-encrypting the related ciphertexts. Our proposed scheme offers a solution to perform a novel technique that dynamically changes the access privileges of legitimate users. The scheme drives the access privileges in a specific way by updating the access policy and activating a user revocation property. Our technique assigns processing-intensive tasks to cloud servers without any information leakage to reduce the computation cost on resource-limited computing devices. Our analytical theoretical and experimental findings and comparisons of our work with related existing systems indicate that our scheme is efficient, secure and more practical compared to the current related systems, particularly in terms of policy updating and ciphertext re-encryption. Therefore, our proposed scheme is suited to Internet of Things (IoT) applications that need a practical, secure access control scheme.

Moreover, to achieve secure, public cloud storage and minimise the limitations of CP-ABE which mainly supports storing data only on a private cloud storage system managed by only one single authority, our proposed access control scheme is extended to a secure, critical access control scheme with multiple authorities. This scheme ought to be carefully designed to achieve fine-grained access control and support outsourced-data confidentiality. In addition, most existing multi-authority access control schemes do not properly consider the revocation issue due to the difficulty of addressing it in distributed settings. Therefore, building a multi-authority CP-ABE scheme along with addressing changes to policy attributes and users, have motivated many researchers to develop more suitable schemes with limited success. By leveraging the existing work, in this thesis, we propose a second CP-ABE scheme that tackles most of the existing work's limitations and allows storing data securely on a public cloud storage system by employing multiple authorities which manage a joint set of attributes. Furthermore, the proposed scheme efficiently maintains the revocation by adapting the two techniques used in the first proposed single authority access control scheme to allow dynamic policy update and invalidate a revoked user's secret key that eliminates collusion attacks. In terms of computation overhead, the proposed multi-authority scheme outsources expensive operations of encryption and decryption to a cloud server to mitigate the burden on a data owner and data users, respectively. Our scheme analysis and the theoretical and implemented results demonstrate that our scheme is scalable and efficient.

Acknowledgement

I would like to express my deepest thankfulness and gratitude to the Almighty Allah for granting me this opportunity and for blessing this project.

I acknowledge my sincere appreciation and gratitude to my Director of study Prof. Qi Shi for his invaluable feedback on my research, for his support of my PhD, for his guidance and encouragement. I also would like to thank my second and third supervisors Dr. Gyu Myoung Lee, and Dr. Kashif Kifayat.

My sincere gratitude goes to my sponsor the Iraqi government, Ministry of Higher Education and Scientific Research of Iraq, Iraqi Cultural Attaché in London, University of Anbar and College of Computer and Information Technology for the support and the funding received towards my PhD.

Also, I really appreciate the support that I received from Liverpool John Moores University, particularly, the Department of Computer Science to improve my skills and to earn this research degree. I am especially grateful to Tricia, Elizabeth, Tracy, Carol, Natasha, Laura, Alexia and all Faculty Research Administrators for going far beyond their duty.

Thanks go out to my supportive friends Ruba, Hiba, Samar, Shaymaa, Ruqayah, Rana, Afaf, Libyan Ruqayah, Islam, Aseel, Raghad, Nashwa, Toey and Basma for encouraging and helping me during the PhD period in whatever way they could.

My heartfelt thanks to my family; Mum (Anaam), Dad (Rabeea), Sisters (Fatima, Zainab, and Aaisha) and Brothers (Omar, Yaseen, and Mohammed) for always believing in me and encouraging me to achieve my goals and dreams.

Publications and Contributions

Conference Papers:

1. Ruqayah R. Al-Dahhan, Qi Shi, Gyu Myoung Lee, Kashif Kifayat, “Revocable, Decentralized Multi-Authority Access Control System,” 2018 IEEE/ACM International Conference on Utility and Cloud Computing (UCC Companion), December 2018. (Zurich, Switzerland)
2. Ruqayah R. Al-Dahhan, Qi Shi, Gyu Myoung Lee, Kashif Kifayat, “Access Privilege Elevation and Revocation in Collusion-Resistant Cloud Access Control,” IEEE Word Conference on Smart Trends in Systems, Security & Sustainability (WORLDS4), October 2018. (London, UK)
3. Ruqayah R. Al-Dahhan, Qi Shi, Gyu Myoung Lee, Kashif Kifayat, “Context-Aware Cloud-Based Access Control for the Internet of Things – Refined Work,” Faculty Research Conference: Liverpool John Moores University, 2018, Liverpool, UK.
4. Ruqayah R. Al-Dahhan, Qi Shi, Gyu Myoung Lee, Kashif Kifayat, “Context-Aware Cloud-Based Access Control for the Internet of Things – Extended Work,” Faculty Research Conference: Liverpool John Moores University, 2017, Liverpool, UK.
5. Ruqayah Rabeea Al Dahhan and Qi Shi, “Context-Aware Cloud-Based Access Control for the Internet of Things – Initial Work,” Faculty Research Conference: Liverpool John Moores University, 2016, Liverpool, UK.

Journal Papers:

1. Ruqayah R. Al-Dahhan, Qi Shi, Gyu Myoung Lee, Kashif Kifayat, "Survey on Revocation in Ciphertext-Policy Attribute Based Encryption,” *Sensors*, vol. 19, no. 7, p. 1695, 2019.

Table of Contents

List of Abbreviations

AA	Attribute Authority
ABE	Attribute-based Encryption (ABE)
CAA	Central Authority
CDHA	Computational Diffie-Hellman Assumption
CP-ABE	Cipher text-Policy Attribute based Encryption
CS	Cloud Server
DBDHA	Decisional Bilinear Diffie-Hellman Assumption
DDHA	Decisional Diffie-Hellman Assumption
DHP	Diffie-Hellman Problem
DLP	Discrete Logarithm Problem
DO	Data Owner
ECC	Elliptic Curve Cryptography
EHR	Electronic Health Record
IaaS	Infrastructure as a Service
IoT	The Internet of Things
KP-ABE	Key-Policy Attribute Based Encryption
LSSS	Linear Secret Sharing Scheme
NIST	The National Institute of Standards and Technology
PaaS	Platform as a Service
PBC	Pairing-Based Cryptography
SaaS	Software as a Service
SSS	Secret Sharing Scheme
SU	System User
W-CP-ABE	Waters' CP-ABE Scheme

List of Figures

List of Tables

Chapter 1

Introduction

Chapter 1: Introduction

1 Introduction

An innovative technology has been introduced, developed and adopted in 1999. This technology is the Internet of Things (IoT) which has changed human life to a huge extent by optimising the techniques of intelligent, big data analysis, expanding the productivity of systems, and increasing the flexibility. Such services shape the industry and are adopted by the academic sector. However, the limited resources of the IoT devices such as battery life, computing power, and storage are the main issues that prevent consumers from successfully leveraging the IoT services and applications.

On the other hand, the method of storing, sharing, retrieving, and processing data has changed with the recent advancements of technology (e.g. cloud technologies). Organisations and individuals are increasingly encouraged to utilise services that enable data sharing, availability and accessibility from everywhere and at any time. The cloud is a well-established technology providing this type of service. However, the expanding use of such services raises critical security issues, particularly in such untrusted environments like the cloud which has the ability to directly access the data that is stored on it, and process it.

To protect sensitive data, data confidentiality needs to be achieved to add a restriction on the range of cloud functions. However, thanks to the fact that the cloud server is the only entity that directly interacts with service-users, it is responsible for deciding who accesses the data. Therefore, an access control mechanism is needed to regulate access to data and expressive access policies are required to be enforced to limit the trust boundaries of the cloud provider.

Therefore, to leverage the merits of the above-mentioned technologies, a new, popular architecture has emerged which combines IoT and cloud, where IoT services and applications are offered and built on the top of the cloud services. In this thesis, we focus on the security of this integrated technology and authorization, mainly access control.

In this chapter, Sections 1.1 presents the research motivation. The challenges, the methodology, requirements, research aims and objectives and the contributions of this thesis are introduced in Sections 1.2, 1.3, 1.4, 1.5 and 1.6, respectively. The outline of the thesis is presented in Section 1.7.

1.1 Research Motivation

The factor that motivated us to undertake this study will be expressed in this section. Where, we are interested in some applications that control children's internet activities by running rating systems. These applications help parents to protect their children from accessing unsuitable online materials.

1.2 Research Problems and Challenges

In untrusted cloud environments, many challenges are outlined as a result of widely leveraging storage and sharing services. These challenges are shown as follows:

1- The cloud storage and sharing services are provided by multiple, unknown platforms to serve a great, unlimited number of users (whose identities are unknown) and who outsource a huge amount of their sensitive data to these untrusted servers to store on them. To avoid any sort of information leak, and grant limited access privileges to these untrusted servers which run critical works on these data, a one-to-many encryption scenario is needed to let an untrusted cloud server share encrypted data with the served users without knowing their identities.

2- Some careful measurements have to be taken due to moving data away from owners' control. Where, making appropriate, required decisions to access the stored data with granting restricted privileges to an untrusted server to participate with this decision-making process, is a major challenge. To establish trust with an untrusted server and tackle the above-mentioned issue, access policies that are identified by data owners as a set of attributes, ought to be enforced. Therefore, granting rights to data owners to enforce their policies to access data and compensate the lack of control, is another critical challenge.

3- Dynamically managing, driving and customising the users' privileges that are changed in response to the frequent change in the values of users' attributes are the key issues that have motivated a considerable number of researchers to tackle these challenges.

4- In the context of encrypting data that is stored on the cloud, constructing a collusion resistant system is still an outstanding issue. In particular, it is essential to prevent a cloud server to collude with the system users who are no longer allowed to access data and already know the decryption key.

5- Distributing the keys securely to a large, dynamic number of authorized system users whose identities are unknown and overcoming problems with the traditional cryptographic approaches are essential challenging tasks. That needs to happen in a way that prevents malicious, authorized users to collude with each other to access a high level of data.

1.3.Research Methodology

Identifying the key cause of the problems described in section 1.2 is the first step in this research. Then, it followed by creating new solutions to rectify the problems. More specifically, the methodology for this research consists of four distinct phases: (1) literature review, (2) requirement analysis and specification, (3) two schemes design, and (4) implementation and evaluation. The scheme design phase also involves several stages as shown below.

1- **Literature review:** to fill the knowledge gap and solve the problems identified, we performed a thorough literature review of cloud based access control and current schemes for Attribute Based Encryption (ABE). In addition, we analysed their limitations and tried to find appropriate solutions to overcome these restrictions.

2- **Requirements analysis and specification:** in this phase, the relevant existing work critically analysed to determine the needs or conditions for the newly proposed schemes to meet, by taking account of the possibly conflicting requirements of the two different technologies which are cloud and IoT. Since requirements analysis is critical to the success of the proposed schemes, the requirements derived must be practical and related to identified schemes needs. Based on these considerations, the initial requirements selected and specified.

3- **Scheme design**: based on the analysis from the previous phase, the essential requirements for the design of our two proposed schemes extracted. The design divided into two main stages: 1) designing a single-authority cloud access control scheme including seven algorithms to support storing data on a private cloud storage system, solve the revocation problem stated later, 2) designing a mature scheme of multi-authority cloud access control with eight algorithms to support storing data on a public cloud model and resolve the single-authority scheme limitations.

4- **Implementation and evaluation**: the final phase of this proposed scheme consists of two stages that are the implementation and evaluation of the designed schemes.

In the implementation stage, we implemented the designed schemes including the algorithms and methods of each scheme and tested them according to the requirements specified. In the second stage of this research, the implemented schemes evaluated and compared against the relevant existing work to determine any benefits our schemes offer and any problems it may experience.

1.4 Research Requirements

In this section, some critical requirements that need to be met for sharing data over the cloud, are presented to distinguish the risks. These requirements are as follows:

1. **Data Confidentiality and Privacy**: It is a set of rules that protects a certain type of information by placing some restrictions on it. It is an essential requirement for cloud storage since the cloud service provider which stores the data, is normally unauthorized to access its content. Thus, the data accessibility ought to be only for explicitly legitimate users. That is satisfied by using cryptographic techniques to enable the legitimate data access even when the data encrypter is offline and preserving the privacy of the users' identities.
2. **Fine-grained Access Control**: It is a key mechanism that grants different access privileges to different users even if they are in the same group according to their credentials given by the associated system, and flexibly specifies individual users' access rights.
3. **Expressive access structure**: It is important for the access policies specified by a data owner to be expressive to realize fine-grained access control. Moreover, an encryption technique is required to support the expressiveness of these policies. This requirement makes the access control scheme similar to a real-life access control.
4. **Collusion Resistance:** The system has to prevent any collusion attacks from combining their information together to illegitimately gain unauthorized data through collaboration [30]. In the cloud environment, this type of attack can be either a group of misbehaving system users who collude with each other, combine their information and gain higher access rights or a combination of a cloud server and malicious, revoked users who try to gain the original data.
5. **Forward and Backward Security:** Forward security means any revoked user ought to be prevented from accessing data and decrypting any new published ciphertext after leaving the system. In terms of backward security, several types of application

need to achieve it. Such applications need a mechanism in which the ciphertexts published previously cannot be decrypted by any user who newly joins the system [31]. On the other hand, other applications do not need to achieve backward security in the sense that a user who newly joins the system can also be able to decrypt the data published previously [32-34].

6. **Revocation:** When a user is degraded or leaves the system, its access rights need to be reduced or revoked, respectively, by the related access control scheme without incurring significant computational cost. In addition, attribute updating is not a straightforward process in ABE and it is hard to address, as updating a single attribute could impact a large number of users accessing the same attribute.
7. **Scalability:** The performance of the system should not to be affected by the increase in the number of the system users.
8. **Computation overhead:** It is essential to fulfil all the above requirements with minimal computation cost.

1.5 Research Aims and objectives

In this part of the thesis, the critical aims and objectives of the project are presented.

1.5.1 Aims

Based on the challenges and problems identified earlier, the general aim of this proposed project is to construct a scheme that supports a spontaneous coalition between the IoT and the cloud, and provides secure storing and sharing sensitive data on the cloud in a specific way that improves the decision-making process to access its resources. This aim will be useful to apply to some IoT applications that have digital contents such as e-books, videos, patient health records and so on. These applications are becoming pervasive in the era of the cloud and need a mechanism to prevent them from being obtained by inappropriate users.

Offering a novel model for dynamic access to cloud resources in response to access policies changes will provide data owners more flexibility and security by allowing them to share their resources with others for easy access to them. The aim of this project will be accomplished by considering the current research in cloud access control strategies and then developing the model for novel, flexible, secure access control. Although many studies have been done in this field, little work has considered all the mentioned problems in one scheme to enhance the decision-making process of regulating access to cloud resources.

1.5.2 Objectives

To accomplish the aim of this proposed project, the following objectives are set out to:

1. Examine the current approaches to cloud access control by performing a thorough literature review and identify their strengths and shortcomings.
    - This objective is achieved in Chapters 2 and 3 that present the existing, relevant techniques, their limitations and issues, as well as some related fundamentals and principles.
2. Develop an attribute-based encryption technique in a specific way and construct a proposed scheme that is able to handle the frequent attributes changes with reference to solving the user revocation problem and stores data in private storage cloud environments.
    - In Chapter 4, the proposed single authority scheme is designed, constructed and implemented. In addition, the security requirements are identified. Based on these requirements and the implementation results, the scheme is analysed and evaluated.
3. Extend the proposed scheme to build an advanced access control scheme that deals with storing data on public storage cloud environments and adjust the proposed revocation techniques to be adapted with the modified scheme, where these two schemes ought to be feasible for IoT technologies.
    - The proposed decentralised multi-authority scheme is designed, constructed, implemented and analysed in Chapter 5.
4. Illustrate the effectiveness of the constructed schemes.
    - The results and discussions in Chapters 4, 5 and 6 show that our proposed schemes address most of the existing limitations (mentioned above in Section 1.5) and support storing sensitive data in an untrusted cloud environment with dynamic privilege management.

1.6 Research Contributions

In this thesis, the design, theoretical details and implementation of our proposed access control schemes are presented. The novelty of our collusion-resistant schemes are to drive the access privileges in a specific way by updating the access policy as well as user revocation. In this part, our core research contributions and advancements in an untrusted, outsourced environment are discussed as follows:

1- Reformulating the scheme of CP-ABE and presenting a new scheme that constructs our novel cloud access control scheme and achieves data confidentiality, fine-grained access control and supports expressive access policies to be secure in untrusted environments. In particular, our proposed scheme is constructed by rebuilding the most popular existing systems that have many merits but lack dynamicity in a dynamic storage environment.

2- Resolving the issue of frequent attribute changes which has not sufficiently been addressed by the existing systems, by providing a novel technique to efficiently handle users' attributes by policy updating, leading to managing and customising users' privileges. This technique helps to elevate, eliminate, or even revoke users' access rights by efficiently dealing with a monotone access structure without incurring heavy computations.

3- Manipulating the user revocation problem by enforcing constraints and a specific formula into users' secret keys which are also known to revoked-users for data decryption to invalidate them after the revocation event occurred.

4- Outsourcing a computational part of the expensive encryption operation to a cloud server without any information leakage about the plaintext leads to minimizing the long-standing time that is required for encryption and policy update. That happens due to merging the CP-ABE with the traditional encryption techniques to efficiently manage a monotone access structure without incurring heavy computation. In addition, part of the policy update and the whole ciphertext re-encryption process are delegated to cloud.

5- Adding a proxy server that activates the user revocation property, makes the scheme robust against collusion attacks to be a collusion resistant scheme.

6- Further extending the proposed scheme and enhancing its security and performance by developing the scheme from a single attribute authority to multi-attribute authorities, which is suitable for more complicated cloud applications. This extension leverages the power of the proposed single-authority scheme such as dynamic privilege management that is considered as a significant, complicated issue to be achieved in a multi-authority scheme.

7- In the proposed multi-authority scheme, secure outsourced decryption is used to mitigate the burden from users.

1.7 Organisation of the Thesis

Chapter 2 introduces related definitions, fundamentals, and principles of ABE schemes. Furthermore, we identify some issues and limitations as well as some related work.

Chapter 3 briefly presents some mathematical background, relevant principles and basic concepts related to advanced cryptographic techniques.

Chapter 4 proposes a new Ciphertext-Policy Attribute Based Encryption (CP-ABE) scheme with a single authority that can practically manage users' access privileges. Moreover, the theoretical model is presented. At the end of this chapter, based on the scheme evaluation and experimental results, we analyse the scheme in terms of security and performance and show that our proposed scheme is secure against collusion attacks.

Chapter 5 extends the proposed algorithm in Chapter4 to build a multi-authority access control scheme that provides higher capabilities compared with the existing work. Furthermore, this chapter presents the implementation of the scheme and discusses the experimental results. In addition, a comparison between the proposed scheme and the most relevant one is carried out to check the practicality of the scheme.

Chapter 6 summarises the chapters presented and concludes the thesis. In addition, it points out some directions of future research extracted from this work.

Chapter 2

CP-ABE for Outsourcing Data to Cloud

Chapter 2: CP-ABE for Outsourcing Data to Cloud

2 Introduction

Unlimited cloud storage and data outsourcing services provide data owners and enterprises with capacities for storing and processing a massive amount of data. These high quality services enable easy accessibility, high scalability and availability [35]. Despite all the advantages, serious concerns about the data security and confidentiality in such a cloud environment have been raised [36]. To preserve data confidentiality, many techniques support encrypting the outsourced data stored in the cloud environment. However, these techniques cannot regulate access to specific stored data or enforce access policies [3].

In an untrusted cloud environment, preserving data confidentiality, making an appropriate decision on data and enforcing access policies are core challenges. Therefore, many system models and techniques for access control based on cryptographic operations have been characterized and described by researchers to provide secure and efficient cloud access control. Although such techniques enable sharing data with a large number of users, some open issues still need to be addressed, particularly for Internet of Things (IoT) applications. Firstly, access control policies should be expressive enough to incorporate relevant contextual information on the properties, features or characteristics that are associated with users, objects, or the environment (e.g. age, position, time etc.), reflecting the frequently changing conditions and correlating with ongoing activities in the environment concerned [37].

Secondly, since IoT applications are varying widely (e.g., e-health including patients' medical records management and remote diagnoses, military systems including soldiers' data management and monitoring, smart vehicles including traffic jam management, and smart cities), securing such applications by building a collusion resistant system and providing fine-grained access control is a key challenge. The reasons for this are the sensitivity of such application data and the consequences of attacking such applications cause great damage to systems and their users. Thirdly, the most important issue from the user point of view is how to flexibly join and leave a system with a low computational cost (i.e. efficient user revocation).

To address the above challenges, a considerable amount of research has been conducted to develop necessary cryptographic techniques. This chapter will critically examine and compare these techniques to demonstrate their effectiveness and efficiency as well as limitations.

In a large-scale distributed environment, particularly the cloud environment, traditional symmetric cryptographic techniques with the same key for both encryption and decryption operations suffer from a key distribution and management problem. On the other hand, traditional asymmetric cryptographic approaches, which utilise a public key for encryption and a private key for decryption, lack computational efficiency. These one-to-one schemes are not desirable to encrypt data and send it to a group of recipients. The reason for this is that a data owner needs to know who the recipients are, their identities, or their public keys, before encrypting its data and sending it to the corresponding authorized users separately.

However, to eliminate the enormous computational costs of the traditional cryptographic operations, and to achieve the mentioned requirements (such as preserving data confidentiality, regulating access to stored data and using expressive policies), attribute based access control has been introduced, which grants access privileges to users based on their attributes. To perform attribute-based access control while preserving data confidentiality, ABE has been proposed, which is an advanced asymmetric cryptographic technique invented to leverage the merits of the symmetric encryption (e.g. high efficiency) and solve its key management and distribution problems.

Two variants of ABE were discussed in [38]. These variants are ciphertext policy attribute based-encryption (CP-ABE) and key policy attribute-based encryption (KP-ABE). The major difference between them lies in how to associate a secret key and an access policy with relevant data and attributes. In KP-ABE, an access policy is associated with a secret key and a set of attributes are associated with the data encrypted with the key. Conversely, in CP-ABE, each encrypted data item is assigned with a specific access policy and a user's secret key for the data decryption is assigned with a set of attributes. As a consequence of embedding an access policy into a user's secret key in KP-ABE, a data owner (who encrypts the data) can only select a set of attributes but will not be able to decide which user can access its ciphertext. To decrypt the ciphertext, a key generator (i.e. an attribute authority), which generates the decryption key for authorized users, will be responsible for granting or denying access to the key [39]. Due to this property, we pay little attention to KP-ABE and focus on CP-ABE in the rest of this thesis, as our objective is to give data owners full control of their sensitive data and CP-ABE is more suitable for this purpose.

The criteria used to choose the related works, in this chapter, are based on selecting various CP-ABE schemes using different methods to solve the same problem (i.e. the revocation problem). In addition, covering all the existing techniques is under our consideration. This chapter is structured as follows: Section2.1 gives a basic description of IoT. The cloud computing technology is described in Section 2.2. The aspects of access control are discussed in Section 2.3. Section 2.4 presents the principle of context awareness. The background of ABE and the main problems of CP-ABE are described in Section 2.5. Existing single authority and multi-authority CP-ABE schemes are introduced in Section 2.6. Finally, Section 2.7 summarises the chapter.

2.1 The Internet of Things (IoT)

The Internet of Things (IoT) refers to a system of smart devices which are intelligently connected to the physical world collecting data by embedded sensors and then leveraging such data. The architecture of the IoT consists of three layers [1]. These layers are the recognition layer, network layer and application layer shown in **Figure 2.1**.

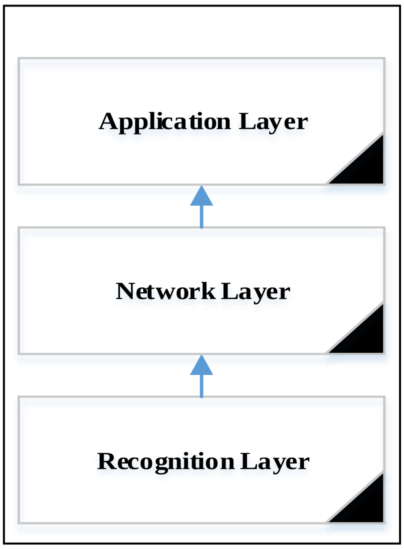


Figure 2.1: The Internet of Things architecture

In the recognition (or perception) layer, data is collected from the environment or things, to be transformed into a digital form. The network layer is the main layer of the IoT which connects the other two layers together and transmits the data from the recognition layer to the application layer which is the top layer of the architecture. The application layer provides the relevant services that meet users' needs.

Hence, this technology offers many services that enhance the quality of consumers' life and increase enterprises' productivity by improving the efficiency of education, health, decision making, and so on [2]. However, the IoT devices are resource–limited things. Therefore, one of the main IoT challenges is the huge amount of data that has to be dealt with. Storing and processing these data needs powerful computing and storage abilities exceeding that found in the IoT. Therefore, integrating the IoT with the cloud that has unlimited storage and computing power, is a critical issue.

To leverage the cloud resources, IoT applications deliver their data to the cloud service provider which provides considerable storage and computational resources for the IoT devices (e.g., in e-healthcare networks). On the other hand, delivering data to the cloud means losing the control and the management of these data, raising serious security issues which form open challenges. To sum up, the combination of cloud computing and IoT (as illustrated in **Figure 2.2)** can provide a universal environment of data collecting services and powerful processing of such data. Whereas IoT produces rich contextual information, the cloud effectively serves as the brain to improve decision-making based on the information provided.

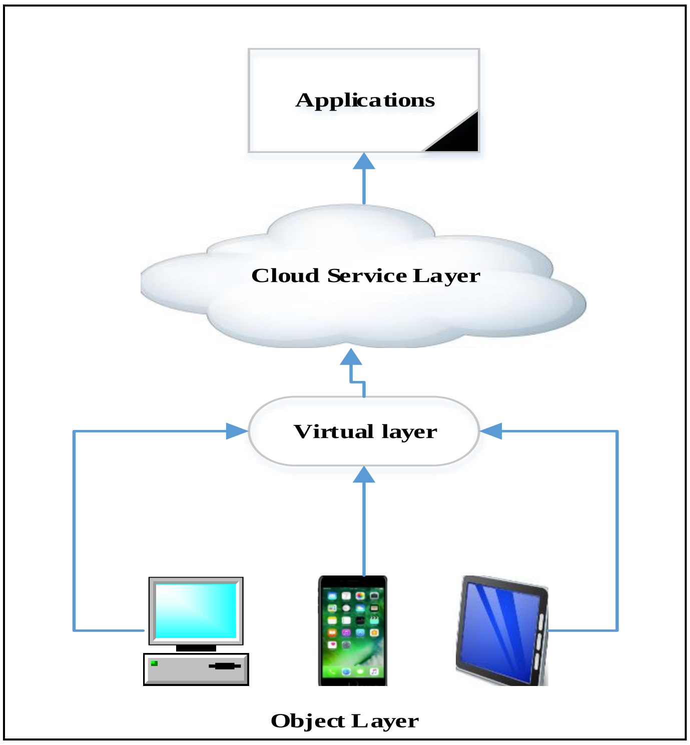


Figure 2.2: The integration of the IoT and cloud.

2.2 Cloud Computing System

Cloud computing is a large-scale computing paradigm for a wide range of functional capabilities, which enables convenient, on-demand network access to a large, shared pool of virtualized, managed computing and storage resources[3]. These resources are delivered effortlessly as services to billions of consumers through the network anywhere and anytime [4]. These services were categorized into three various models by the National Institute of Standards and Technology (NIST)[5] as Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS).

Software as a Service (SaaS) is a software distribution model. In particular, the cloud service provider hosts applications and makes them available to a large number of consumers over the internet. As a result, SaaS provides shared access to the applications running on the cloud infrastructure instead of running these applications on the organization's computers and hardware [6]. That eliminates the expense of hardware acquisition, maintenance and the need for software installation. In addition, SaaS offers high scalability by giving consumers the option to access more or fewer features or services.

The second model is Platform as a Service (PaaS), in which, hardware and software tools which are needed for application development such as java development or application hosting, are delivered by the provider to users through the network, which in turn allows shared access to tools and programming languages demanded to deploy an application. This is beneficial to companies and consumers by allowing them to focus on creating and developing applications instead of maintaining the infrastructure. The access to PaaS will be charged by a provider on a pay-per-use-basis that many enterprises prefer.

The last model is Infrastructure as a Service (IaaS). It provides a virtual machine to the cloud users, which is overcommitted physical servers to expand profits from investing hardware. Where, it allows shared access to underlying hardware resources such as network, storage and other computing resources. The need for more physical machines is reduced by Server virtualization. That helps to avoid the complexity and expense of maintaining and managing their own physical resources.

Generally speaking, cloud computing can be categorised into three deployment models based on the type of data that are dealt with and the required levels of security and management. These types are public, private and hybrid clouds. In a private cloud, only exclusive use is offered for the cloud infrastructure to a single organisation. Such infrastructure and services may be managed by the organisation or a third party. While in a public cloud, public use is provided for the cloud infrastructure which is managed by a government or an academic organisation or combined organisations. The composition of these two models is considered as a hybrid cloud.

Many applications which use intensive data and depend on SaaS and PaaS models, require logical data storage as these applications need to operate simultaneously on contiguous data. Cloud storage is a service in which managing, maintaining and backing up data are made remotely available to users over the internet. It maximises the benefits by hosting users' data on the cloud servers from anywhere at any time. Recently, many companies have started to offer the cloud storage service.

Due to being provided with the accessibility, flexibility and data recovery with low cost by the cloud storage service [7], many companies are motivated to outsource their data to cloud storage servers. In that way, the companies' need for housing special equipment for storing their data is minimised with a guarantee that those data are protected against any natural disaster, stealing or system crash. In addition, scalability is one of the major benefits that the cloud provides. That enables the company to be enlarged to accommodate future needs.

However, storing data on remote external servers and assigning a variety of essential responsibilities of managing and maintaining those data to the cloud service provider without any intervention from the data owners is a critical issue. Once the data is outsourced to the cloud servers, the data owners will lose the control of their data. Since the cloud service provider is not totally trusted by the cloud consumers and a large amount of the stored data is highly sensitive, integrity, security and privacy are major issues in cloud computing.

Therefore, to maximise the adoption of the cloud storage service, some appropriate cryptographic techniques ought to be undertaken to satisfy two key features [8]. The first one is the *confidentiality* which ensures that the cloud service provider has no knowledge about the customers' data. The second feature is the *integrity* which means the ability of the cloud customers to detect any illegitimate modification carried out on their data by the cloud service provider or any attackers.

In terms of cloud users who need to access to data which is stored on the cloud, one of the main mechanisms used in the cloud environment to manage authorized access, is access control whose main responsibility is to manage users' access rights. It grants access to authorized users and forbids others access to data [9]. Due to the distributed environment with untrusted cloud servers, efficient mechanisms for regulating access over encrypted data are required. Therefore, many system models and algorithms for access control have been characterized and described by researchers to provide secure and efficient cloud access control.

2.3 Access Control

In the cloud environment, the sensitive, large, scalable, stored data requires a secure manner to protect and preserve its integrity and confidentiality without affecting the scalability and the performance of the system. One of the critical security mechanisms for data protection is access control that permits, restricts or denies access to system files by setting some conditions and rules which are combined together to make and enforce an access control decision [10]. In this way, the access control technique can ensure only authorized users who need to access certain data, have the ability to do that.

Some core requirements need to be achieved in any effective cloud access control system. The first one is fine-grained access control [11]. In particular, each user in a system has their own access right which may differ from others in the same group. Due to lack of control, the second requirement is to assign control to the data owners after residing their data on the cloud without computation overhead which is the third requirement. To keep data safe and guarantee the security, the data has to be encrypted. That will keep data away from being illegitimately accessed by a cloud server or any unauthorized users. Therefore, the fourth requirement is confidentiality [12].

To meet the above requirements, attribute based access control has been introduced [13]. However, to hide the data from a storage server, encrypting data is essential before storing them on such servers. Thus, data encryption with attribute-based access control is known as an Attribute-Based Encryption (ABE) technique. An attribute is a piece of information that describes the properties, features or characteristics of an object [14]. This information can be recognised by either automated or human approaches. For example, an attribute could be a department (e.g. engineering, computer science, etc.), an occupation (e.g. teacher, student, researcher, etc.), and experience years (e.g. two-years, five-years, etc.). In general, attributes are classified into two types [15]: 1) non-temporal attributes with discrete attribute values (e.g. age, address, email, etc.), and 2) temporal attributes with continuous values (e.g. interval, time, etc.).

Many studies have been carried out on the cloud access control using ABE with discrete attribute values [13, 16, 17]. These studies have a lot of problems which remain unsolved. For example, Yang et al. [17] proposed an access control model with low computational costs for decryption. However, the work incorporated weak privacy and security considerations. On the other hand, some schemes require intensive computations in return for stronger privacy and security protection, meaning that they are unsuitable for mobile devices with limited computation power [18, 19].

In addition, some work has been carried out using ABE with continuous attribute values, which is known as temporal access control [20, 21]. The access structure can be in the form of time (e.g. between 8 am and 12 pm). These temporal attributes are familiar in the cloud. For instance, only during a particular period of time, can users access certain data. However, such schemes have their shortcomings. For example, the scheme of Zhu et al. [21] does not address user revocation, and the scheme of Yang et al. [20] manages the revocation problem inefficiently by refreshing an update key and sending it to all users at every time slot with a valid set of attributes which represent the revocation.

Achieving data confidentiality and access control for the cloud data is a core challenge that needs to be taken into account. Addressing this challenge supports data security management, and allows data owners to regulate their data and enforce restrictions on accessing data. Traditional cryptographic techniques can keep data confidentiality. However, the other requirements (mentioned above) are hard to achieve with these types of techniques.

2.4 Context awareness

Due to rapid changes in users' context, the use of the context information is crucial in interactive applications, particularly for ubiquitous computing applications [22]. Context can be defined as "any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves" [23]. If a system uses the context to supply relevant information to a user who uses it in a specific task, it is considered as context-aware [24].

Some critical issues have to be considered when the systems and applications intend to use context information gathered from the environment [25]. The first one is organizing the gathered information in an effective way that is compatible with the system requirements. The second one is how systems can utilize contextual information to provide improved services to the system users [26].

Although different researchers have developed context-aware access control models, the use of context has been targeted at detecting devices and network environments that were used to request an access to the cloud data. This information can be useful to detect the computational power of the devices or to check if the requested data will be transmitted through insecure channels [27, 28].

However, an active access control model requires a much broader scope of context and is centred on the context which consists of all the characterizing information considered relevant to it. This context can be identified as identities, locations, times or activities which are collected and labelled in some meaningful way and represented in terms of attributes [29].

2.5 Attribute-based Encryption (ABE)

ABE is one of the advanced cryptographic techniques for one-to-many encryption that overcomes the limited functionalities of the traditional public-key cryptographic techniques. This scheme was proposed by Sahai and Waters [40] as an application of fuzzy identity-based encryption which uses human-intelligible identities (such as unique name, IP address, email address, etc.) as public keys, where a data sender directly encrypts its data with the receivers' identity. Later, Goyal et.al. [38] present a more general construction of ABE in which attributes have been utilized to issue a public key and to generate a logical expression of these attributes called an access policy. Both the public key and access policy are used for encrypting data. In contrast with the traditional cryptographic systems which encrypt data to one particular user or group that knows the decryption key, there is no more need to share the same private key or store several versions of the ciphertext encrypted with different keys [41]. Moreover, ABE has no restriction on the number of users in the system. Based on these considerations, this scheme has been leveraged to regulate users' access to cloud data by using attributes as an access policy.

To apply ABE, a data owner encrypts its data using a symmetric encryption algorithm with a symmetric key and then encrypts the key using an ABE scheme with a public key. The encrypted key is distributed to a group of recipients/users as a ciphertext. Each user obtains the private key sk for the encrypted key decryption from a key generator that calculates the key according to the user's attributes. In this case, the data owner does not need to know the identities of the legitimate users and their dynamicity. **Figure 2.3** illustrates the above operational process. Applying ABE to the two variants (i.e. CP-ABE and KP-ABE) follows the same procedure (as shown in **Figures 2.4 and 2.5**). The main difference is, in KP-ABE, users' secret keys are issued using an access policy that defines the access privileges of the authorised user, and the symmetric key is encrypted over a set of attributes. However, CP-ABE uses access policies to encrypt data (i.e. symmetric key) and secret keys of the legitimate users are generated over a set of attributes.

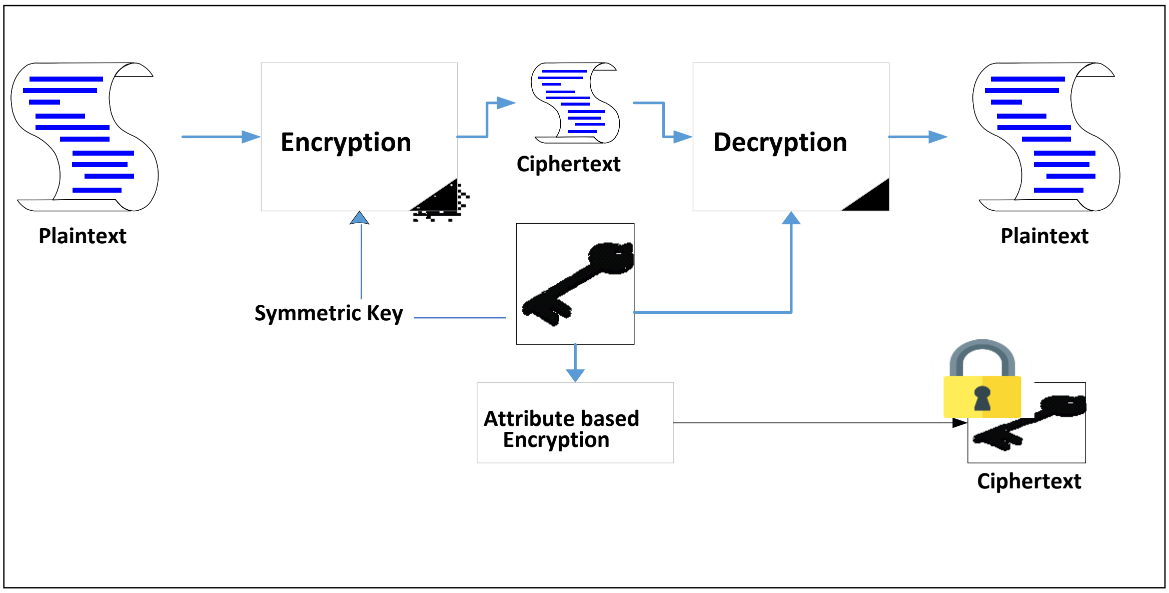


Figure 2.3: Using ABE to encrypt a symmetric key

Many KP-ABE and CP-ABE schemes were proposed with some notable examples listed in **Table 2.1**.

Table 2.1: Summary of the proposed ABE schemes

Scheme	Description	Revocation	Access Policy
Bethencourt et al.[39]	The first CP-ABE scheme using a tree access structure.	Lack of revocation	Less expressive
Waters[42]	The first fully expressive CP-ABE scheme using a linear secret sharing access structure.	Lack of revocation	Full expressive
Wang et al.[43]	The first hierarchical ABE scheme with a disjunctive normal form (DNF) policy.	Addressing revocation	Not expressive

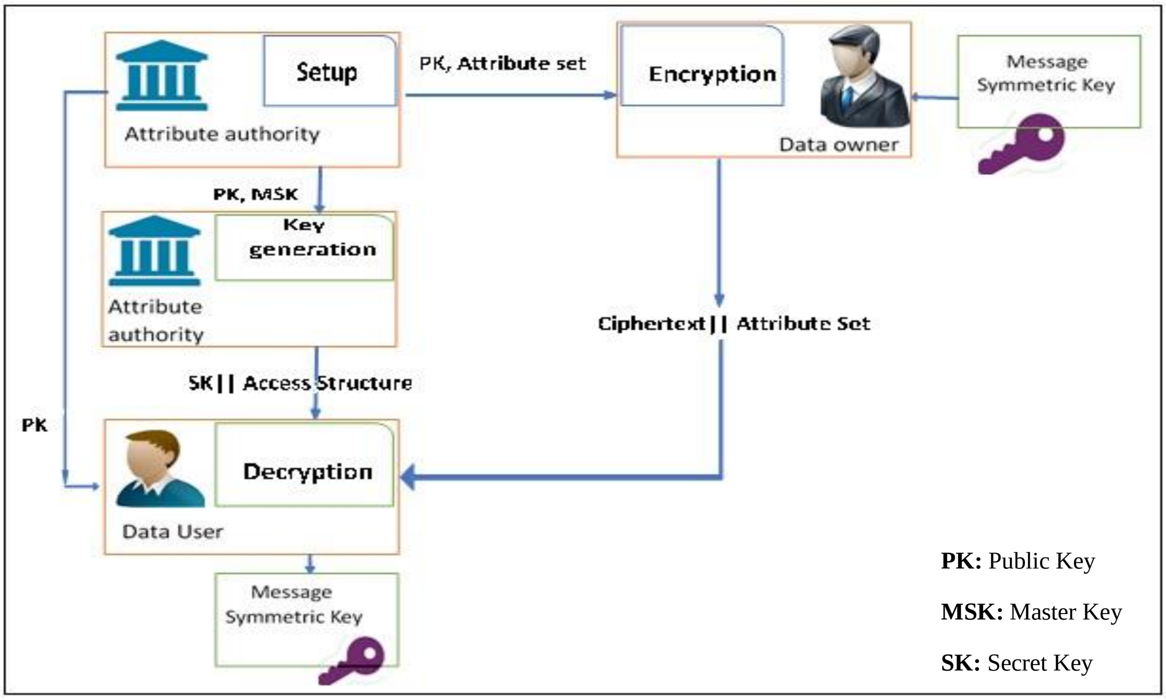


Figure 2.4: The procedure of KP-ABE

2.5.1 Ciphertext–policy Attribute Based Encryption (CP-ABE)

The most popular variant of ABE techniques is CP-ABE. Four entities are responsible for running this scheme. These entities are attribute authority, data owner, data user and cloud server. The role of the attribute authority is to generate secret keys for users according to their attributes to decrypt data. In addition, it is responsible for generating a public key and a master key. A data owner’s role is to define an access policy that describes who can access its data as well as encrypting those data under this access policy. Firstly, the data owner uses a symmetric encryption technique (e.g. AES) to encrypt its data. After that, the owner encrypts the symmetric key under its access policy using CP-ABE by selecting a random value as a secret which is shared using a linear secret sharing (LSS) technique to generate some values associated with each corresponding attribute in the ciphertext according to the owner’s access policy. This policy is determined over a set of attributes by the data owner and can be demonstrated as a Boolean function with (AND, OR) gates between attributes (e.g. (lecturer AND experience >= 2 years) OR Professor). Then the encrypted data is sent to the designated cloud for storage including the data ciphertext, the CP-ABE ciphertext and the access policy. Associating the access policy with the ciphertext means that the ciphertext chooses which key can recover the plaintext, giving the data owner more control of its outsourced data [44]. The eligible users who possess the required attributes in a right combination (i.e. satisfy the access policies) can successfully decrypt the encrypted data. As a result, the main benefit from using CP-ABE is that sensitive data can be stored on an untrusted server without performing authentication checks for the data access [45].

A common framework of a CP-ABE scheme includes four algorithms as demonstrated in **Figure 2.5**: Setup, Encryption, Key Generation and Decryption [42], which are defined below:

- $\boldsymbol{Setup}(\boldsymbol{\lambda}, \boldsymbol{U}) \rightarrow (\boldsymbol{MSK}, \boldsymbol{PK})$: Takes a set of attributes $\boldsymbol{U}$ in the system and an implicit security parameter $\boldsymbol{\lambda}$ (such as the type of the elliptic curve group used and the base finite field) as inputs to generate a public key $\boldsymbol{PK}$ and a master key $\boldsymbol{MSK}$ as outputs.
- $\boldsymbol{Encrypt}(\boldsymbol{PK}, \boldsymbol{A}, \boldsymbol{M}\ ) \rightarrow \boldsymbol{CT}$: Takes as inputs a public key $\boldsymbol{PK}$, an access structure $\boldsymbol{A}$, and a message $\boldsymbol{M}$ to be encrypted. The output will be a ciphertext $\boldsymbol{CT}$.
- $\boldsymbol{KeyGen}(\boldsymbol{MSK}, \boldsymbol{S}) \rightarrow \boldsymbol{SK}$: In this algorithm, a master key $\boldsymbol{MSK}$ and a set of attributes $\boldsymbol{S}$ are taken as inputs. A user's secret key $\boldsymbol{SK}$ is generated as output.
- $\boldsymbol{Decrypt}\ (\boldsymbol{CT}, \boldsymbol{SK}) \rightarrow \boldsymbol{M}$: This algorithm takes as inputs a user's secret key $\boldsymbol{SK}$ and a ciphertext $\boldsymbol{CT}$. It returns a message $\boldsymbol{M}$ when the user's attributes satisfy the access structure.

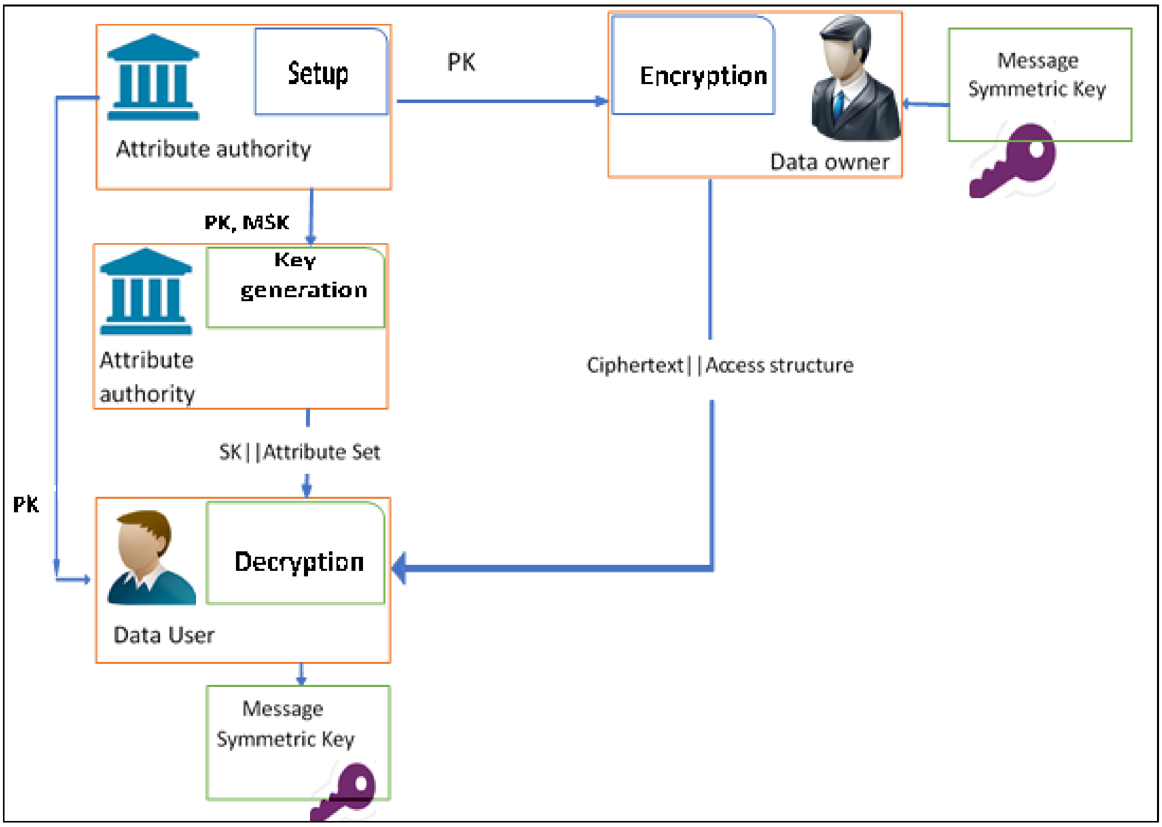


Figure 2.5: The CP-ABE mechanism

There are some appealing merits of the CP-ABE technique over other one-to-one traditional encryption techniques that enable coarse-grained access control. First, CP-ABE is to enable fine-grained access control in an encrypted form. This is desirable for many access control applications that run some cloud services such as storage and sharing services. Secondly, it supports highly expressive policies representing any access structures. Thirdly, it offers a good solution to data confidentiality. As generating a secret key for a user happens only once but it can be used to decrypt all the subsequent ciphertexts, CP-ABE reduces communication overhead [44]. Fourthly, it is ***collusion resistant*** against misbehaving authorized users, which is achieved by associating a random number or polynomial with each attribute of a user's secret key so that only the attributes with the same random value can be used for decryption, leading to preventing different legitimate users from colluding with each other [46]. Finally, it is possible to integrate CP-ABE with a ***proxy re-encryption technique*** in cloud in order to increase security by re-encrypting ciphertexts without disclosing the plaintexts to the cloud [44].

However, there are some weaknesses related to the CP-ABE scheme. These include that CP-ABE only works fine when attributes are descriptive [11]. In other words, temporal attributes are not well handled by CP-ABE. In addition, this technique is difficult to handle the ***attribute/user revocation*** problem [47] without trusting the cloud service provider that already hosted the data, particularly in dynamic environments where users' attributes can change over time. Trusting a cloud server raises another issue which is a ***collusion attack***. This attack involves revoked users colluding with the cloud server to combine their information together to gain access to unauthorized data. Therefore, the adoption of CP-ABE requires additional refinements.

Based on the entities that a user can obtain authorization from, CP-ABE is classified into two different categories [48]. They are single authority CP-ABE, where all attributes are handled by a single authority, and multi-authority CP-ABE in which different authorities manage the attributes in a distributed manner. However, in multi-authority systems, many complicated issues can be experienced when the CP-ABE systems are built. For example, to tie the work of all authorities together, some existing systems use either a central authority that could cause a bottleneck problem and is contradictory to the distributed control principle [49], or coordination between the authorities, which increases communication and computational costs. In addition, each authority needs to be aware of each other, running the risk of collusion by combining their information to figure out unauthorised information.

In addition, the revocation process is more complicated to manage in this type of system [41, 49].

On the other hand, some issues can be encountered in any single authority systems. The first one is the key escrow problem that happens due to the ability of the authority in this type of the system to gain access to all users' keys. This ability is obtained by the authority as it possesses the master key from which the users' keys are derived. The second issue is the limited ability of any single authority system to handle a wide range of different attributes. Moreover, it represents a single-point bottleneck on security. Once an adversary compromises the system, the authority's master key is easy to obtain.

To efficiently exploit the advantages of CP-ABE and avoid most of its drawbacks, we need to construct a new scheme of CP-ABE that mitigates the difficulties in the existing schemes, uses sufficient, alternative solutions that dynamically change users' privileges without entrusting information to a cloud server, and builds a collusion resistant system. The main issues that we are concerned about, are:

- Resolving the revocation problem.
- Covering a wide range of attributes needed by any system, and eliminating a single point failure.
- Reducing the computation overhead.

2.5.2 The Revocation Problem

Revocation is a property to change the access rights of users when unexpected events occur such as malicious behaviour from a user, or an expired service that a user had purchased [41]. There are two scenarios where the revocation can be conducted. The first one is called *attribute revocation* that happens when some of a user's attributes are removed from the current set due to being degraded in the system. For example, degrading a manager of an organization to a normal employee role leads to losing some of its possessed attributes and hence access rights. The second scenario occurs when a user leaves the system; its access rights have to be revoked so that the user is no longer able to access the system and decrypt any stored data on it, which is called *user revocation* [50]. Based on these considerations, designing a mechanism to revoke the user's certain access rights must be embedded in the system from the beginning. Otherwise, the whole system has to be rebuilt with the advent of each revocation event.

In attribute-based access control schemes, the attribute revocation is a severe problem and very costly to apply for two reasons. The first one is the same attribute may be associated with different users' secret keys, causing significant computational overhead throughout the revocation process. This happens due to the need for updating all relevant keys for the non-revoked users and re-encrypting the related ciphertexts [41]. The second reason is most of the existing proposed attribute revocation methods are based on a semi trusted server, but this is an unrealistic assumption [50]. This is because the server could breach the trust and even be compromised, resulting in permitting unauthorized users to access the data stored in the cloud for gaining profits, e.g. when the cloud illegitimately permits a company to access the data of its competitors.

Therefore, the following crucial requirements are needed to handle the revocation problem:

1) Permit instantaneous banning of a malicious user,
2) Resist collusion attacks or invalidate the secret keys of the revoked users [41] (which means the cloud cannot collude with a revoked user to illegally obtain encrypted data),
3) Minimise the computation overhead of a revocation process,
4) Support forward security which means any newly published ciphertext cannot be decrypted by any revoked user with revoked attributes [51].

Although considerable research has been devoted to solving the revocation issue, most of the existing studies lack practicality and the revocation process is considered as the major hindrance (**Table 2.2** illustrates the main existing systems). The current strategies and assumptions utilised for the revocation are either considering that the server used is a trusted entity that can be assigned critical, essential, auxiliary processes of access control or, in the worse-case scenario, assuming that the data owner and a private key generator (attribute authority) stay online all the time [50].

Some studies have been done to handle the revocation problem periodically [52, 53]. Wan et al. [52] propose a hierarchical attribute-set-based encryption scheme with user revocation. To cope with user revocation, they added an attribute expiration time to a user's key. This time indicates the validity period of the user's key. However, this causes serious vulnerabilities due to the uncontrolled period from the revocation time of a user to the expiration time of its key, as well as bringing an extra computational burden to the authority for frequent key updating and maintaining secure channels for all transactions. On the other hand, other schemes have also been proposed with instantaneous attribute revocation [41, 50, 54, 55].

Several researchers have worked to build systems that resist collusion attacks [56, 57]. These schemes [56, 57] use a secret sharing scheme in order to prevent the server from decrypting the ciphertexts or illegitimately granting permissions to revoked users to access the data. In addition, they achieve dynamic, immediate attribute and user revocation without updating keys of non-revoked users. However, these schemes can only revoke a limited number of users. On the other hand, some schemes can revoke an unlimited number of users [45]. In [45], a unique identifier is associated with every user's secret key, which is in turn used to construct the revocation information to be embedded in the ciphertext. However, the ciphertext size increases linearly with the number of revoked users, which has a negative impact on available storage capacities, particularly when the amount of data is large.

Moreover, applying the revocation process consumes a lot of computing resources. The attribute revocable system proposed in [54] needs to update all keys for the non-revoked users and re-encrypt the related ciphertext, which leads to low scalability and high computational overheads. Other recent studies have utilized a refereed delegation of computation models to alleviate the computation overhead of the identity-based encryption during the revocation process [58, 59]. These schemes introduce an aided server which is a Key Update Cloud Service Provider, to outsource most operations of the key generation related to the revocation process. They assign, for each user, a hybrid private key which contains two types of component, identity and time components which are combined together using the AND gate. The users need periodically to contact the Key Update Cloud Service Provider to update their time components according to a revocation list. However, this approach requires the server to be honest.

Moreover, some studies have developed an attribute revocation process using techniques based on issuing versions of users' secret keys [50, 55]. In [55], each user uses the old versions of its secret key to get the newest one, which leads to storing all versions of updated keys in the cloud to avoid a *stateless receiver* problem which happens when users lose their previous keys needed to compute their updated secret keys. However, keeping records of all the previous secret keys leads to a storage overhead. To overcome this problem, another mechanism has been proposed. Only the latest secret key needs to be held by its corresponding users in [50]. Instead of updating all the non-revoked users' secret keys and re-encrypting the ciphertexts, only the components in the secret keys and ciphertexts associated with the revoked attributes need to be updated. The workload of ciphertext update will be delegated to a server. Although this system improves the efficiency of the attribute revocation mechanism and reduces the storage overhead, it requires the cloud server to be semi-trusted in the sense that although the cloud does not have knowledge about the plaintexts, it has to possess parts of secret information and thus has to be trusted to deal with these secrets properly.

A dynamic user revocation scheme was proposed by Xu et al. [60]. In this scheme, the cloud server is in charge of re-encrypting ciphertext by using its assigned delegation key. However, this scheme does not handle the attribute revocation. So, the user will lose its access right of accessing data in the system, when it is put on the revocation list even if it still has other access attributes. However, some studies have enabled CP-ABE with proxy re-encryption which transforms a ciphertext of a message into another ciphertext of the same message by a semi-trusted proxy server using a re-encrypting key without any knowledge of the underlying plaintext [61], to achieve the attribute and user revocation [62]. In the scheme of Zu et al.[62], two master keys are generated by the authority. One of them is sent to the cloud server to deal with the revocation process, and the other is used to derive the secret keys of users. So, when the revocation event occurs, the non-revoked users' access rights would not be affected. Although this scheme does not need to update keys in the case of attribute revocation, there is a need to re-encrypt the ciphertext.

In addition, some studies have been proposed to accelerate the revocation process by applying a mechanism to change just the affected part of data instead of the entire one [63]. In such a scheme, the data is split into a number of slices using the variant of a secret sharing scheme (SSS) which is called All or Nothing, and then it is outsourced to the cloud. When a revocation process happens, only one slice needs to be retrieved by the data owner in order to re-encrypt and then re-upload it. However, the data owner must conduct the revocation process. To overcome the problem of the owner having to stay online all the time, the revocation process may not be executed immediately and also requires additional computation costs.

Recently, some works have been concerned about the attribute revocation issue. In [64], the authors proposed a system which addresses the problem of revoking users and attributes dynamically. The revocation process is executed by the cloud server which re-encrypts the ciphertext according to the revocation list using the proxy re-encryption techniques, responds to the queries of the non-revoked users and partially decrypts the ciphertext for them. Moreover, the cloud server has additional shares of the system attributes that are used for attribute revocation. In this way, revoking one attribute from some users' privileges will not affect the access of other legitimate users. Although the system outsources heavy computational tasks to a cloud server (in particular, re-encryption and a part of decryption operations) and addresses the problem of revoking users and attributes dynamically, the cloud is required to be semi-trusted. Therefore, the system does not resist against collusion attacks and partly grants the cloud server more control over data access. Moreover, the ciphertext size in this system increases linearly with the number of revoked users due to an additional ciphertext header and other components.

Furthermore, adding new attributes to the updated access policy is a critical mission which some of the existing systems do not manage. However, the work in [65] addresses this problem. Although the access policy is enforced cryptographically, it can be changed dynamically without updating users' secret keys. A dynamic policy update process is needed to transform an old LSS matrix to an updated one corresponding to their relevant policies. When the two matrices are compared, the attributes changed by the access policy updating and the corresponding vectors in the matrix will be recognised to change only the ciphertext components associated with those updated attributes. The distribution of the re-encrypted ciphertext after updating the policy is similar to the distribution of the old ciphertext. However, many changes frequently occur in a set of ciphertext components and these changes are done by the data owner. That means an additional computational burden on the data owner. Moreover, the system re-randomizes the ciphertext before updating it. The re-randomization cost is similar to the cost of the whole encryption ciphertext process which leads to communication and computation overhead.

Table 2.2: Classifying main existing systems based on the revocation type**.**

Scheme	Description	Revocation type	The problem
[52, 53]	Add an attribute expiration time to a user's key	User revocation	Periodically
[56, 57]	Resist collusion attacks	Attribute and user revocation	Limited number
[45]	Revoke an unlimited number of users	User revocation	The Ciphertext size increases linearly with the number of revoked users
[54]	Consume a lot of computing resources	Attribute revocation	High computation overhead
[58, 59]	Alleviate the computation overhead	Periodic attribute revocation	Collusion attack
[50, 55]	Issue versions of users' secret keys	Attribute revocation	Collusion attack
[60]	Dynamic revocation	User revocation	No attribute revocation
[61]	Enable CP-ABE with Proxy Re-Encryption	Attribute revocation	Collusion attack
[62]	Use two master keys	Attribute and user revocation	Collusion attack
[63]	Accelerate the revocation	User revocation	Computational burden on a data owner
[64]	Dynamic revocation	Attribute and user revocation	Collusion attack
[65]	Updated access policy	Attribute revocation	Computational burden on a data owner

Moreover, some schemes have been built in a multi-authority cloud environment, where the attribute revocation problem is a more complicated issue. Most of the existing revocation techniques are either not efficient or based on a trusted server. So it is not sufficient to apply them to multi-authority schemes [66]. The scheme of Abraham and Sriramya [48] does not require a server to be totally trusted, because updating keys is carried out by each attribute authority and not by the cloud server. However, in this revocable scheme, the burden of the revocation process is shifted to the authorities which in turn are exposed to corruption due to periodically communicating to system users. On the other hand, the identity-based revocation technique in a multi-authority system is introduced [41] which leads to distributing the computational overhead over a large number of users when they run the encryption and decryption algorithms. However, the computational burden for revocation has negative effects on the users.

In cloud storage systems, granting trust to the server that is curious about a user's privacy, or maintaining the data owner online all the time, is not an appropriate situation. In addition, the shortcomings of the existing schemes are: a) a lazy revocation process implying delay in revocation, b) issuing new secret keys to the non-revoked users, c) re-encrypting ciphertexts during user /attribute revocation, or d) expanding the ciphertext size. All these issues highlight the real need for building a system, which securely outsources expensive computations to a server without any leakage of private information so as to achieve privacy-preserving revocation. This will achieve two essential perquisites. The first one is to prevent the cloud from colluding with the revoked users or gaining any information about the plaintext, and the other is to reduce the computation cost on the data owner.

2.6 The Types of CP-ABE Scheme

In terms of distributed control in an untrusted cloud environment and based on the way of granting authorization to users (i.e. depending on gaining secret keys by users from a single trusted entity or from a group of independent, cooperative entities), the CP-ABE schemes can be classified into two categories that are described below.

2.6.1 The Single Authority Scheme

Most of the existing systems employ one entity to have the power of generating the decryption private keys for all system users. In such schemes, one attribute authority administrates all system attributes. This authority has the master secret key that is used to derive all users' decryption secret keys. These keys are distributed to the system users via secret channels. The inherent issue of this type of scheme is the key escrow problem that occurs due to the ability of an attribute authority to recover any ciphertext using its master key. However, the security assumption that such systems is based on, is that the authority is fully trusted. On the other hand, crashing or corrupting this entity affects the availability of the whole system.

In addition to the key escrow issue, due to the lack of schemes that can efficiently address some issues of CP-ABE, such as revocation and collusion resistance, many researchers have been motivated to construct a practical CP-ABE system using a single authority scheme (**Table 2.3** shows a summary of these studies). Notably, some approaches have been introduced to eliminate the computation cost [67] for lightweight devices. This type of study uses a CP-ABE scheme to offer a constant size for both ciphertexts and secret keys. It uses one-way hash functions and an encryption algorithm to produce a ciphertext and a special polynomial function to generate a secret key by a key generation algorithm. However, it supports the AND-gate access structure. In addition, in this proposed scheme, the revocation problem is not taken into account.

Furthermore, some recent work has been proposed to alleviate the computations to be appropriate for resources-limited mobile devices [68-70]. In [68], the system is introduced with a large attribute universe-based access control, when the space and number of system attributes are flexible and not limited in the setup phase, and outsources decryption to the cloud. This single-authority system uses the LSSS access structure. While in [69, 70] the online-offline technique is used to eliminate most computations. The online/offline CP-ABE scheme in [69] is proposed to mitigate the online-encryption computation burden on an e-healthcare record (EHR) owner by splitting these computations into offline computations which are performed before knowing the data and specifying the access policy, and few online computations which are required to keep the battery life long-lasting. In this scheme, LSS is used to encode the access policy. However, these systems [68-70] do not address the revocation problem.

In the scenario of encrypting medical records where a data owner (patient) ought to generate a secret key to system users, outsourcing the operation of key generation is desirable. Therefore, some researchers have proposed a fully outsourced ABE scheme [71] that achieves outsourced key generation, encryption and decryption. The system supports the LSSS access structure. In terms of outsourcing the key generation and reducing the communication cost (e.g. battery consumption), the server will generate an intermediate secret key with only knowing the public key which can be downloaded later by the user after charging its mobile without draining the battery (i.e. the outsourcing operation is offline). To protect the master secret key and the private keys, the data owners hire two different servers to generate secret keys. However, the two servers colluding with each other means the whole system is under collusion attacks. Furthermore, the revocation problem is not considered in this system.

Moreover, outsourcing the heavy operations of CP-ABE to fog computing has also attracted a lot of researchers' attention [34]. Fog computing is a paradigm extended from the cloud computing. Such a scheme [34] uses the access tree as an access structure. The system proposes an approach to outsource part of the encryption and decryption operations to fog nodes in order to minimise the computational burden on the data owners and system users, respectively. In addition, the system addresses the attribute change by updating the secret keys for all affected-system users who share the updated attribute. However, sending the updated key to those users via a secure channel causes communication and computation overheads. Furthermore, the system assumes that the cloud service provider, fog nodes and the attribute authority are semi trusted.

On the other hand, some studies have been carried out to reduce the computation cost of the ABE by using a pairing-free ABE system [72, 73]. In [72], the system also eliminates the transmission overhead on the secure channel by sending the large part of a secret key on a public channel to the users while sending only the blinding factor via a secret channel. However, using the pairing instead is more reliable and secure. In addition, this scheme does not use the LSSS to distribute the attributes in the users' secret keys, which is a more expressive access structure than the threshold scheme that this scheme used. Moreover, the revocation problem is not taken into account by the authors.

In [73], beside reducing the complexity of ABE by using pairing-free ABE, invalidating the leaked keys of non-revoked users and the revoked keys is considered. When a key of a non-revoked user is accidently leaked or revoked upon attribute revocation, a key-insulation technique is utilised to divide the system lifetime into several periods. At each period of time, only one part of the secret key can be updated by the authority which computes the updating components and sends them to the authorized users. The authority uses a random number for each period of time. The system supports the tree access structure. However, the system does not support an instant key invalidation operation. Alternatively, the revocation happens periodically. Furthermore, a heavy computation cost is imposed on data owners thanks to re-encrypting a plaintext at each period of time.

Furthermore, work has been done not only to outsource the decryption operation to a cloud server but also go further to verify that the outsourced decryption carried out by the cloud, is correct [74]. The verification technique transforms a ciphertext using some processes with the blinded secret key. Then before computing the plaintext, a user compares each transformed ciphertext component with the corresponding, original ciphertext component to retrieve the blinded value. In this case, the outsourced decryption is verified. The tree access structure is used in this system. However, the system uses a big size of ciphertext as well as incurring a heavy computation cost. In addition, the revocation problem facing any access control system is not addressed.

As a result of the importance of protecting health information which may be revealed by access policies, some studies have been proposed to hide an access policy in CP-ABE schemes [75]. This system [75] uses a large attribute universe and partially hides an access policy. The system handles any expressive access policies represented as LSSS. However, some additional computational operations are added before the decryption phase for testing whether a user's attributes satisfy the access policy, imposing more burdens on a user. Furthermore, the scheme does not consider the revocation problem.

In addition, recent studies have been proposed to achieve more security by hiding the access policy [76, 77]. In [77], the authors present a CP-ABE scheme that provides two features. The first one is to hide the attribute values from the attribute authority. That happens by utilising the 1-out-of-n oblivious transfer technique that can send attributes in a fuzzy selection manner to the authority; in this case, the authority can generate the secret key without knowing the attribute value. The second one is to protect the type of attribute in the access policy that is embedded in the ciphertext using the attribute bloom filter approach to check whether an attribute belongs to the hidden access policy without revealing it. The LSSS access structure is supported in this system. However, more computational operations are incurred by a data owner and users. These operations increase linearly with the complexity increase of the access structure and the number of the users' attributes. Furthermore, in terms of communication overhead, more information needs to be sent to the attribute authority. In addition, the revocation problem is not managed.

Although most of the existing systems can hide access structures and support restricted access structures with a composite order group of which the order is a product of two large primes, the scheme in [76] introduces a mechanism to partially hide access structures with enabling the expressive LSSS access structure in a prime-order group which is a cyclic group with a prime-number order. Pairing performance in a scheme with a composite order group is about 50 times lower than the same pairing in the prime order group [78]. In general, each attribute consists of a name and a value. In this scheme [76], the attributes' values in the access policy are hidden by the data owner due to their sensitivity. The authors use the randomness splitting mechanism to protect the values of the attributes by hiding them in the ciphertext. However, the revocation problem is not dealt with. Furthermore, expensive operations are needed to compute the ciphertext and each user's private key as well as increasing size of the ciphertext.

Table 2.3: Summary of main existing systems and their limitations

Scheme	Description	Access Structure	The problem
Odelu et al [67]	Eliminate the computation cost for lightweight devices	AND-gate	Lack of revocation
Fu et al [68]	Provide large attribute universe-based access control	LSSS	Lack of revocation
Liu et al and Li et al [69, 70]	Offer online-offline techniques to eliminate most computations	LSSS	Lack of revocation
Zhang et al. [71]	Propose a fully outsourced ABE scheme	LSSS	Lack of revocation
Zhan et al. [34]	Outsource the heavy operations of CP-ABE to fog computing	Access Tree	Inefficient revocation
Karati et al.[72]	Reduce the computation cost of ABE by using pairing-free ABE	Threshold	Lack of revocation
Hong and Sun [73]	Reduce the computation cost of ABE by using pairing-free ABE	Access Tree	Periodical revocation
Kumar et al. [74]	Outsource and verify the decryption operation	Access Tree	Lack of revocation
Zhang et al. [75]	Hide an access policy in CP-ABE schemes	LSSS	Lack of revocation
Cui et al. and Han et al. [76, 77]	Hide an access policy in CP-ABE schemes	LSSS	Lack of revocation

However, it is essential to consider the problems that single authority schemes create. These include 1) the diversity of attributes that are hard to manage by only one authority, 2) the key escrow problem that occurs when the single authority is not totally trustworthy and has the ability to gain access to all users' keys, and 3) the security failure that creates a serious problem when the authority is compromised by an adversary that gains the system's master key. A multi-authority scheme is suitable for resolving these weaknesses.

2.6.2 The Multi-Authority Attribute based Access Control System

To tackle the single authority schemes' problems, an effective way is introduced to minimize the trust level of the single authority, strengthen the privacy of user data and enhance the system security and performance by replacing the single authority with multiple ones for disjoint attribute management that becomes much harder for an adversary to compromise. Therefore, in this section, the type of scheme that allows securely storing data on a public cloud storage system and employs multiple authorities which manage sets of attributes, is presented.

A critical challenge of current multi-authority access control systems (also all single authority schemes) is the inefficiency of the key generation process. This issue occurs in the single authority systems when one authority manages all attributes in a system and issues secret keys for all system users. Therefore, compromising or crashing this authority makes the whole system unavailable. The same issue happens in multiple authorities schemes, when each authority in the system administrates a disjoint attribute set (i.e. each authority administrates a different set of attributes), which presents a performance bottleneck. To mitigate the effects of this issue, all attribute sets ought to be managed by all system attribute authorities individually (i.e. joint attribute sets).

Although many recent multi-authority CP-ABE schemes have been proposed [79, 80], some limitations are still not considered. The existing multi-authority access control systems can be classified into three categories with their limitations summarised below:

- The first type of scheme (e.g. a scheme by Han et al. [81]) contains many authorities that have to work together, resulting in a high communication cost and lack of scalability since it is hard for authorities to join or leave freely. Furthermore, these authorities might collude with each other and combine their information to gain unauthorized data about the users.

- The second type needs a central authority to tie the work of all authorities together, and to be involved in issuing users' secret keys besides having the master key (e.g. the work by Liu et al. [82]). The drawbacks of this type of scheme are that the concept seems contradictory to distributed control and it incurs low performance and a security bottleneck.
- Decentralized systems are the third type of the multi-authority access control system, which remove any central authority and employ independent attribute authorities, where the systems are scalable (e.g. the system proposed by Ruj et al. [83]). For this type of system, user revocation is hard to address, which incurs a heavy computational cost.

The first multi-authority access control system was proposed by Chase et al. [84]. This system uses a central authority as an active entity, which generates users' secret keys, co-operates with the system attribute authorities that manage disjoint attribute sets and distribute the secret keys. The problem with this system is the central authority has the master key that can be used to decrypt all ciphertexts. This means that the central authority represents a performance and security bottleneck. The system also does not address the revocation problem.

Yang et al. [17] proposed a decentralized access control model by using multi authorities with a semi-active central authority which is only in charge of initialising the system. In their system, part of the decryption operation is outsourced to a cloud server to mitigate the burden of decryption on a user. Moreover, the system supports the revocation process. However, in the revocation phase, heavy computation is put on attribute authorities (AAs) for computing an update key for each non-revoked user. Since the attributes change frequently, this approach becomes a performance killer and not practical in cloud access control systems. In addition, the attribute set in this system is divided into various disjoint subsets where each one is driven by one authority. Once an authority is compromised, the adversary can gain the corresponding private keys of its attributes, which in turn affects the performance of the whole system.

Another multi-authority scheme has been proposed [85] to advance the system in [17] by jointly managing a system attribute set. In this work, a verifiable threshold multi-authority access control model with a semi-active central authority is introduced using a secret sharing approach to generate a shared master key among multiple authorities, where all the attribute authorities collaborate with each other to create the key. In this scheme, users' secret keys can be generated by contacting a threshold number of attribute authorities. However, this system does not address the revocation problem. In addition, some communication and computation are needed among the authorities to exchange their shares and reconstruct the master key. Furthermore, a heavy computational workload is placed on users.

Moreover, some researchers have claimed that they propose a revocable threshold multi-authority access control system with the management of joint attribute sets [86] to advance the system in [85]. However, the theoretical model that is presented in this work, uses an access tree as an access structure, unlike the scheme in [85], which uses LSSS as an access structure. Moreover, the theoretical model shows that the system is performed as a single authority access control system, not as the authors have claimed. Furthermore, the system does not address the attribute revocation problem. In addition, the experimental results are vague. The same issue occurs with another study [87] which is not as its authors have claimed.

Administrating joint attribute sets is advocated in [88]. This system adopts the technique in [85] and employs a framework to eliminate communication costs by efficiently assigning a part of the secret key generation task to the central authority. Since this assigning operation is based on receiving intermediate keys from t attribute authorities where these keys are associated with attributes, this operation does not affect negatively on solving the single-point performance bottleneck problem with the other systems. However, the system does not address the revocation problem and assumes that the central authority is fully trusted and has the master key, meaning that the compromised central authority with one corrupted attribute authority can break the system security.

Although CP-ABE schemes give data owners more control over their data, a decentralised system with multiple, uncoordinated authorities has been proposed to increase data owners' control over the data by giving them more privileges to restrict access to a fraction of data [89]. In this scheme, even if the user's attribute set fulfils a data owner's access policy, the user can decrypt a fraction of a related ciphertext according to how many fractions are specified by the data owner. The data owner encrypts its whole data once using one policy and different symmetric keys. The approach utilized is chunk based encryption which divides data into several chunks and a different symmetric key is used to encrypt each chunk. This scheme uses LSSS as an access structure. However, although the scheme improves the encryption process, it makes the key generation process more complicated. In addition, it does not manage the revocation problem.

Recently, some studies have been carried out to devise a decentralized multi-authority system with no central authority and without any interactions among the authorities involved [90], where the attribute sets are disjoint. The authors proposed an approach to hide the access policy and resolve the revocation problem. However, upon each revocation event, expensive computational operations are needed. These include updating the secret keys for all non-revoked users after generating an updated key (containing the new version of the revoked attributes) by the authorities and re-encrypting the components of the ciphertext already stored on the cloud server and associated with the revoked attributes. Furthermore, heavy computations are put onto the data owner due to its heavy responsibilities. These include the data owner’s responsibilities for encrypting the ciphertext, hiding the access policy by replacing each attribute in the access policy by a value of pairing between the hash value of that attribute and the public key of the authority that drives this attribute, re-encrypting the cipher-text, and sending it again to the cloud server.

Another recent work is proposed in [91]. It uses a decentralized multi-authority scheme with accountability to trace the misbehaving users who leak their decryption keys. In the system, each attribute authority deals with a disjoint attribute set. Although there is no central authority in the system, there are some interactions among the authorities to share a secret function. In addition, the system hides the attribute information in the ciphertext. However, the system uses an AND-gate access policy in an inflexible way and also managing the accountability leads to an increased ciphertext size because part of the ciphertext that deals with an access policy, is specified to include authorized users’ identities with ‘*’ used if no specific identity is required. Furthermore, the number of authorities is defined in the system initialization phase, which means the system is not scalable afterwards. Moreover, in this system, the most critical revocation issue is not addressed.

Some recent studies have claimed to solve the problem of key escrow, prevent the key-abuse attack and minimise the level of single authority trust by proposing an accountable authority [92, 93] instead of using a multi-authority scheme. The work in [92] proposes two accountable, revocable systems that manage the problem of accountability, traceability and privilege revocation of malicious users. However, in these two systems, the researchers use two techniques for revocation. The first one uses a revocation list and embeds it in the ciphertext leading to an increase in the ciphertext size, while the second approach implements periodical key updating that comes with an additional cost of communications and computations by issuing an updated key for non-revoked users. Moreover, it is not an effective solution to use an accountable authority as an alternative of multiple authorities due to the lack of administrating a wide range of attributes. The work uses a composite order group, which needs complicated processes and makes the system less efficient than a prime order group.

In addition, some researchers proposed a multi-authority scheme with a hidden-structure attribute based encryption [94]. They use a tree access structure and disjoint attribute set. In this system, a central authority plays a main role of creating the system master key sent to all attribute authorities. This concept works similarly to a single authority approach, which lacks decentralisation and represents a security bottleneck. Moreover, the system inefficiently addresses the revocation problem. After each revocation process, the central authority has to update the revocation list and re-issue a new master secret key and send it to all authorities in the system in a secure manner. These authorities regenerate new secret keys for all non-revoked users.

To sum up, a multi-authority CP-ABE scheme is an appropriate solution to addressing security and privacy issues as well as enhancing the performance in the cloud environment. However, the current work has several limitations with notable ones listed in **Table 2.4**, including inefficient revocation, high communication and computation costs, and inefficient key generation. These challenges highlight an urgent need to propose a multi-authority scheme that can not only securely outsource expensive computations to cloud without revealing private information but also efficiently control user access privileges. Outsourcing computations to the cloud reduces the computation costs on data owners and users while allowing user access privileges to be efficiently elevated or revoked according to a policy update process.

2.7 Summary

Some mentioned benefits, requirements and weaknesses in this section are summarized as follows:

1- Traditional cryptographic techniques suffer some problems. These include a key distribution and management problem, lack of efficiency of the computational operations and lack of proper usability in cloud environments (where there is a group of recipients).

2- The CP-ABE technique is chosen to be a promising technique for access control due to its benefits. The main benefits are to enable fine-grained access control in an encrypted form, support highly expressive policies, offer a good solution to data confidentiality, and provide ***collusion resistance*** between misbehaving authorized users.

Table 2.4: Summary of main of the existing systems and their limitations

Scheme	Description	The problem	Attribute set
Han et al.[81]	The authorities have to work with each other	High communication cost	Disjoint
Liu et al.[82]	Using an active central authority to administrate attributes	Security bottleneck	Disjoint
Lin et al.[95]	Decentralized threshold authorities work together without a central authority	Lack of revocation	Disjoint
Ruj et al. [83]	Decentralized system without a central authority	Lack of revocation	Disjoint
Li et al.[96]	The central authority is not involved in generating secret keys	Support AND access structure which is not expressive	Disjoint

3- There are some issues related to the CP-ABE scheme. The first issue is that it is difficult to handle the ***attribute/user revocation*** problem. The second one is that the ***collusion attack*** issue may arise due to trusting a cloud server that may collude with the revoked users and combine their information together to access unauthorized data.

4- Crucial requirements are needed to handle the revocation problem. These requirements are to a) permit instantaneous banning of a malicious user, b) invalidate the secret keys of the revoked users to prevent collusion attacks, c) remove or even reduce the computation overhead of a revocation process, and d) support forward security.

5- The shortcomings of dealing with the revocation issue in the existing schemes are: a) a lazy revocation process implying delay in revocation, b) high computation overhead due to updating all secret keys of the non-revoked users and re-encrypting ciphertexts, or d) expanding the ciphertext size.

6- In the existing work, to avoid the collusion attack, the orientation is to maintain the data owner online all the time or to grant trust to the server that is curious about a user's privacy, or.

7- Based on the distributed control, CP-ABE schemes can be classified into two different schemes. The first type is a ***single authority*** *CP-ABE* scheme, where all attributes are handled by a single authority. The second one is a *multi-authority CP-ABE* scheme in which different authorities manage the system attributes.

8- The security assumption that single-authority systems are based on, is that the authority is fully trusted. The issues, which single authority schemes create, include a) the difficulty of handling the diversity of attributes by only one authority, b) the key escrow problem, and 3) the security failure that creates a serious problem when the authority is compromised by an adversary.

9- To minimize the trust level of the single authority, strengthen the privacy of user data and enhance the system security and performance, the single authority is replaced with multiple ones to make it harder for an adversary to compromise a system.

10- A multi-authority CP-ABE scheme is an appropriate solution for addressing security and privacy issues as well as enhancing the performance in the cloud environment. However, the current work has several limitations including the difficulty of efficiently addressing the revocation problem, high communication and computation costs, and the inefficiency of the key generation process.

Chapter 3

Preliminaries, Basics and Technical Approaches

Chapter 3: Preliminaries, Basics and Technical Approaches

3 Introduction

The high-level mathematical background of pairings and some basic principles of pairing-based cryptography, which CP-ABE is built on, are reviewed in this chapter. In addition, a brief study of techniques that deal with access-control policies and the ways to embed these policies in ciphertexts is provided, especially, those that are relevant to our proposed work.

This chapter is organised as follows. Section 3.1 introduces some relevant mathematical tools. Pairing-based cryptography is presented in Section 3.2. Section 3.3 describes Shamir's secret sharing scheme (SSSS). Section 3.4 provides the types of access structure, its representation approaches and an example of how to represent an access policy using a linear secret sharing scheme (LSSS). The complexity assumptions, the selective security model, the Waters' system that our proposed scheme in Chapter 4 is based on, and the summary of this chapter are discussed in Sections 3.5, 3.6, 3.7 and 3.8, respectively.

3.1 Mathematical Tool

To understand the cryptographic algorithms that will be presented in the subsequent chapters, some mathematical methods need to be introduced. In this section, these methods are briefly described. For a more extensive introduction of the methods, please refer to the relevant references given throughout the chapter for details.

3.1.1 Group

A group $(G,*)$ is a set of elements G which is associated to a binary operation (*) which takes any two elements in the group, and combines them to form a third element in that group [97]. If the set and the operation satisfy the four group properties, it will qualify as a group. These properties are described as follows:

1. Closure: $\forall a, b \in G$, then $a * b \in G$
2. Associativity: $\forall a, b, c \in G, then\ a * (b * c) = (a * b) * c$
3. Identity element: There exists one identity element e which has the property such that $\forall a \in G,\ a * e = a$
4. Inverse: Every element has an inverse, that is: $\forall a \in G, \exists b \in G$ such that $a * b = e$

A group is **finite** when it has a finite number of elements. $|G|$ or $\#G$ is the order of a finite group, which is the number of elements in its set. A group is called an **abelian group**, if it has an additional property as follows:

- Commutative: $\forall a, b \in G, (a * b) = (b * a)$

In most cases in cryptography, this property is used because it makes the groups cryptographically useful (for example,$g^{xy} = g^{yx}$).

An abelian group is called a **cyclic group** if there is a single element $\boldsymbol{g}$**,** from which all other elements in the group can be obtained by frequently applying the group operation to $\boldsymbol{g}$. Such an element $\boldsymbol{g}$ is called the generator of the group and is mathematically denoted as $\langle g \rangle$.

3.1.2 Finite Fields

A finite field (F) is a mathematical group with a finite number of elements and two binary operations and satisfies the usual arithmetic properties:

1. $(F, +)$ is an abelian group with additive identity denoted by 0.
2. $(F \backslash \{0\}, .)$ is an abelian group with its multiplicative identity denoted by 1.
3. The distributive law holds: $\forall a, b, c \in F \quad (a + b).c = (a.c) + (b.c)$

An example of a finite field is all the integers modulo a prime number p. This finite field is denoted by Z_p (e.g $Z_5 \backslash \{0\} = \{1,2,3,4\}$ is a finite field).

Any two fields of the same number of elements (order) are said to be **isomorphic**, meaning that they are structurally the same. It is possible to map between two isomorphic fields (for example, F_1 and F_2) using a field isomorphism Φ:

$$\Phi: F_1 \rightarrow F_2$$

3.1.3 Discrete Logarithm Problem (DLP)

The discrete logarithm problem (DLP) [98] is the basis for a one–way function. DLP is a logarithm defined with regard to cyclic groups. If G is a cyclic group of order n and g is a generator of G, then from the definition of cyclic groups, any element h in G can be calculated as g^x for some x. The discrete logarithm of h to the base g in the group G is defined to be x. We denote that as $x = log_g$ h. For example, if the group is Z_5 , and the generator is 2, then the discrete logarithm of 1 is 4 because $2^4 \equiv 1 \; mod \; 5$. Discrete logarithm problem is not always hard. The hardness of finding discrete logarithms depends on the groups. In the conventional cryptographic schemes, the order of the group n must be prime and very large (usually at least 1024-bit) to make the cryptographic systems safe.

3.1.4 The Diffie-Hellman Protocol (DHP)

DHP is one of the existing standard protocols that uses the discrete logarithm problem to work in finite fields and elliptic curves. The Diffie-Helman problem is closely related to the hardness of computing the discrete logarithm problem over a cyclic group. For instance, Alice and Bob are two people who want to exchange their keys using the Diffie-Hellman protocol. They pick a cyclic group G with order p and a generator g. Then they randomly choose $a, b \in [1, p]$ and start exchanging g^a, g^b. In that case, the secret key equals g^{ab}. The Diffie-Hellman function that is ***hard to be computed*** by any passive attack is defined as:

$$DH(g^a, g^b) = g^{ab}$$

That is what is called Computational Diffie-Hellman Assumption (CDHA). However, this assumption alone cannot provide a sufficient level of security [99] due to the ability of the attacker to collect useful information and predict a big part of the secret key. Therefore, alternatively, most of the existing cryptographic systems capture a stronger assumption, which is the Decisional Diffie-Hellman Assumption (DDHA). In particular, ***it is hard to distinguish*** between two tuples $\langle g^a, g^b, g^{ab} \rangle$ and $\langle g^a, g^b, g^c \rangle$.

3.2 Pairing-Based Cryptography

Pairing-Based Cryptography (PBC) [100] is an area that uses pairings to construct complex cryptographic schemes. This type of cryptography is based on elliptic curve cryptography. The main idea of such schemes is to create a function that takes two points on an elliptic curve group to output an element in a finite field, which is called a pairing e. This mapping allows reducing the Decisional Diffie-Helman problem in one group to an easier, different problem (i.e. Computational Diffie-Helman problem) in another group, where many cryptographic schemes are based on this reduction process.

3.2.1 Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) is a type of public key encryption based on elliptic curve groups over finite fields [101]. An elliptic curve E is the set of points (x, y) with x and y elements of a finite field F_q described by the equation:

$$y^2 = x^3 + ax + b$$

where a and b are parameters which determine the shape of the curve. In addition, it requires that the discriminant $\Delta = 4a^3 + 27b^2$ is nonzero. Equivalently, the polynomial $x^3 + ax + b$ has distinct roots. This ensures that the curve is non-singular. Moreover, there is a need to a point at infinity φ. So E is the set of:

$$E = \{(x, y): y^2 = x^3 + ax + b\} \cup \{\varphi\}$$

Geometry can be used to make the points of an elliptic curve into a group. An elliptic curve group G consists of the elliptic curve points and a group operation called addition, denoted by '+'. Furthermore, the point at infinity serves as the identity element, where adding points on an elliptic curve is closure. The addition law on the elliptic curve group has properties that are shown as follows:

(a) $P + \varphi = \varphi + P = P$ ∀ P∈E.

(b) $P + (-P) = \varphi$ ∀ P∈E

(c) $P + (Q + R) = (P + Q) + R$ ∀ P, Q, R ∈E.

The addition operation of elliptic curve groups has the property of being commutative, i.e. ∀ P,Q ∈G then P + Q = Q + P. Elliptic curve groups could enable shorter keys, while providing a similar level of security to the conventional multiplicative group of a finite field. Due to the small key sizes of Elliptic Curve Cryptography (ECC) and relatively fast computations, ECC becomes the most popular choice for public key encryption at many applications especially those which use sensors (e.g. to achieve a 80-bit security level, there is a need to use 1024-bit key in RSA while a 160-bit curve in ECC is needed).

The difficult issues that most elliptic-curve cryptographic schemes are based on are DLP and CDH problems. These problems can provide a sufficient level of security if the related parameters are chosen properly. While the security assumption that the pairing based cryptography relies on is the decisional Bilinear Diffie-Hellman problem and up to now there are no known attacks breaking this problem.

3.2.2 Bilinear Pairing

Let G_0 and G_T be bilinear, cyclic groups of prime order p, and g be a generator of G_0 [102]. A map $e : G_0 \times G_0 \rightarrow G_T$ denotes a bilinear map if the following properties are satisfied:

1. **Bilinearity**: for all $g \in G_0$ and $a, b \in Z_p$, $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$

2. **Non-degeneracy**: $e(g, g) \neq 1$, where $e(g, g)$ is a generator of G_T.

There are two common types of bilinear pairings, which are Tate pairing and Weil pairing. These pairings become useful due to the bilinearity property. The main difference between them is the speed of computation where Tate pairing is faster.

3.3 Shamir's Secret Sharing Scheme (SSSS)

Shamir's secret sharing scheme [103] is a threshold scheme in which the secret is divided into several parts (shares) and it requires just some of these parts to reconstruct the whole secret. If someone has fewer than the required parts, the secret will not be determined. This scheme is based on polynomial interpolation, where the basic idea of this scheme is to use k points to define a $(k-1)$ degree polynomial (e.g. two points are required to uniquely define a line which is a one-degree polynomial). SSSS consists of the following two protocols:

- The distribution protocol, where a data owner with a secret S generates and distributes the shares of S amongst n users in a $(k\ of\ n)$ threshold fashion with $k < n$. The protocol allows the owner to pick a random $(k-1)$ degree polynomial $f(x) = a_0 + a_1 x + \cdots + a_{k-1} x^{k-1}$ and set $f(0) = S$, where $a_0 = S$ and $(a_1, a_2, \ldots a_{k-1})$ are randomly chosen. Each share $(x_i, f(x_i))$ $(1 \leq i \leq n)$ is then created by computing n points on the polynomial.

- The reconstruction protocol, where an authorized set of k users recover secret S by binding their shares together using the Lagrange interpolation.

3.3.1 Lagrange Polynomial

The Lagrange interpolating polynomial is the unique polynomial $P(x)$ which passes through a set of n given points $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ where $y_i = P(x_i)$ $\{\forall i\colon 1 \leq i \leq n\}$. The degree of this polynomial is the least degree that assumes these points (i.e. $degree \leq (n-1)$). The Lagrange polynomial $P(x)$ is computed (as in **Equation 3.1**):

$$P(x) = \sum_{j=1}^{n} P_j(x), \text{ where} \qquad (3.1)$$

$$P_j(x) = y_j \prod_{k=1, k\neq j}^{n} \frac{x - x_k}{x_{j-} x_k}$$

3.4 The Access Structure and its Representation Approaches

The CP-ABE scheme is almost like a real access control scheme due to its expressiveness. This technique allows a data owner to formulate its policies over a set of attributes and credentials for different groups of users. These policies that can be represented by an access structure are mandatory to be applied in a cryptographic manner by the owner who uses these policies to encrypt its data and store it on the cloud.

An attribute is a piece of information that describes the properties, features or characteristics of an object [14]. This information can be recognised by either automated or human approaches. For example, an attribute could be a department (e.g. engineering, computer science, etc.), an occupation (e.g. teacher, student, researcher, etc.), and experience years (e.g. two-years, five-years, etc.).

To illustrate the approach of CP-ABE, for example, suppose a data owner encrypts a message under a ***policy*** indicated as (Physics AND (Master Student OR PhD Student)), where “Physics”, “Master Student”, and “PhD Student” are ***attributes***. In this way, the owner is able to gain more control over the outsourced encrypted data without needing to know the identities of the eligible users. Once a user’s attributes satisfy the access policy associated with the encrypted data, it is able to recover the associated plaintext.

There are two types of access structure, which are the monotone and non-monotone access structures [104]. The monotone access structure is defined as:

Definition 3.1: “Suppose a set of parties $\mathcal{P} = \{P_1, P_2, \ldots., P_n\}$. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots., P_n\}}$ is monotone if for any B and C: if B $\in$ $\mathbb{A}$ and B $\subseteq$ C then $C \in$ $\mathbb{A}$. A monotone access structure is a collection $\mathbb{A}$ of non-empty subsets $\{P_1, P_2, \ldots., P_n\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called unauthorized sets” [105].

The non-monotone access structure is to indicate that the absence of attributes in the ciphertext ought to be included explicitly, where the negative word ”NOT” is used to describe every such attribute. For example, in a teaching hospital, to allow only doctors and medical students to access data, we need to explicitly add the attribute “**NOT: nurses AND NOT: staff members AND NOT: optometrist**”, and any negative attributes to the access policy express that such users are not allowed to decrypt the ciphertext, without mentioning doctors and medical students, because all attributes in this type of access structure should be negative. However, using this type of access structure has some drawbacks. The most important ones are storage and computation overheads due to the increase in the sizes of the ciphertext, an access policy and the secret key.

Therefore, in our thesis, we suppose that attributes are similar to parties and only monotone access structures are considered. Three common, monotone access structures are used for representing any access policy in CP-ABE systems. Where any monotone Boolean formulas involve AND, OR and threshold operations, an access policy can be transformed into one of these methods: 1) a monotone AND-gate access structure, 2) a (t, n)-threshold access tree and 3) LSS matrices. **Table 3.1** shows a brief summary of various schemes with different access structure representations.

Table 3.1: Summary of the complexity of the access structures in various CP-ABE schemes

Scheme	Access structure representation approach
Phuong et al.[106]	AND-gate
Nishide et al.[107]	AND-gate
Li et al.[108]	Tree
Lai et al.[109]	LSSS
Cui et al.[110]	LSSS

To describe the methods of access structures realization, the AND-gate access structure is a restricted, inexpressive form of the access structure due to supporting only policies with logical conjunction. On the other hand, although the tree access structure is a more flexible, expressive form than the AND-gate one because it supports “AND”, “OR”, and “threshold” operations (as illustrated in **Figure 3.1**), it is hard to be applied to a multi-authority system. The LSSS access structure is a more flexible, expressive, popular and efficient tool because each attribute (that is already connected with other attributes in the access structure) can be dealt with independently and it is easier to apply some mathematical operations to a matrix than a tree structure.

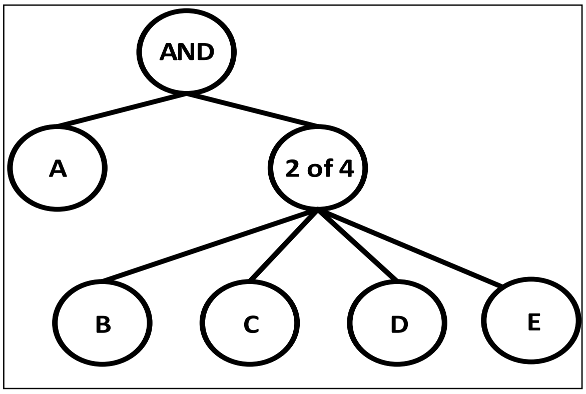


Figure 3.1: Threshold access tree with five attributes located in the leaf nodes and the Boolean AND gate and the threshold gate in the non-leaf nodes

These realization methods are used for describing the access policy and enforcing it into the ciphertext. In our scheme, the ciphertext-policy attribute based encryption scheme uses LSSS matrices to implement monotone access structures, which are included in each ciphertext. The reasons for that are when the access policy is represented as an LSSS matrix, it is difficult to be comprehended by anyone who is not an expert (as shown in **Equation 3.2**). In addition, this tool is highly expressive.

$$W = \begin{bmatrix} 1\ 1\ 0 \\ 1\ 2\ 1 \\ 1\ 2\ 2 \\ 1\ 2\ 3 \\ 1\ 2\ 4 \end{bmatrix} \begin{matrix} \rho(1) = A \\ \rho(2) = B \\ \rho(3) = C \\ \rho(4) = D \\ \rho(5) = E \end{matrix} \qquad (3.2)$$

Where W is a matrix and the function ρ maps the rows of the matrix W to the corresponding attributes (A, B, C, D, E). The LSSS matrix size equals to the number of the attributes in the access tree (i.e. leaf-nodes). The Boolean formula of the access policy that specifies an authorized user who has an attribute A and two other attributes in $\{B, C, D, E\}$, is:

$$(\boldsymbol{A\ AND\ ((B\ AND\ C)\ OR\ (B\ AND\ D)\ OR\ (B\ AND\ E)\ OR\ (C\ AND\ D)\ \ OR\ (C\ AND\ E)\ OR\ (D\ AND\ E))})$$

where it is desirable to transform the access policy to the shortest, equivalent Boolean formula to reduce the size of LSSS.

3.4.1 Linear Secret Sharing Matrix Generation

To understand how we can generate an LSSS matrix that will use it in our proposed schemes in the next chapters, in this section, some principles of LSSS are described. In addition, an example illustrating the process is presented as below:

A) Linear Secret Sharing Scheme (LSSS*)*

A secret sharing scheme Ʊ is linear if the following properties are satisfied [105, 111, 112]:

1- The shares of a secret of each party form a vector over a finite field Z_p.

2- The scheme includes a matrix $W(l \times n)$ with l rows and n columns. For each row W_i of matrix W $(1 \leq i \leq l)$, there is a function $\rho(i)$ mapping this row to the corresponding party (e.g. in the case that a party is an attribute "Student", $\rho(1) = "Student"$ maps the first row of W to "Student"). A vector $\overrightarrow{v}$ is defined as $\vec{v} = (s, r_2, \ldots, r_n)$ where s is the secret to be shared, $s \in Z_p$, and $r_2, \ldots, r_n \in Z_p$ are randomly chosen to hide secret s. The result of $\overrightarrow{W.v} = (\lambda_1, \ldots, \lambda_l)$ is the vector of shares where each party $\rho(i)$ possesses share λ_i.

In every LSSS, for any authorized set S (e.g. authorized attributes) in an access structure A, $I \subseteq \{1, \ldots, l\}$ is defined as $I = \{i: \rho(i) \in S\}$. There is a set of constants, $\{w_i \in Z_p\}_{i \in I}$, which are used with valid shares to reconstruct the secret $\sum_{i \in I} w_i \lambda_i = s$. Here, $\{w_i\}_{i \in I}$ can be computed in polynomial time, satisfying:

$$\sum_{i \in I} w_i W_i = (1,0, \ldots, 0) \qquad (3.3)$$

B) Example:

Let us have an access policy that can be described using the following Boolean formula:

$$(Y\ \boldsymbol{AND}\ \big(((X\ \boldsymbol{AND}\ M)\boldsymbol{OR}(F\ \boldsymbol{AND}\ R)\big)\ \boldsymbol{OR}\ ((N\ \boldsymbol{OR}\ S)\ \boldsymbol{AND}\ (V\ \boldsymbol{OR}\ W)\big))$$

The access tree that can be represented from this formula, is illustrated in **Figure 3.2**.

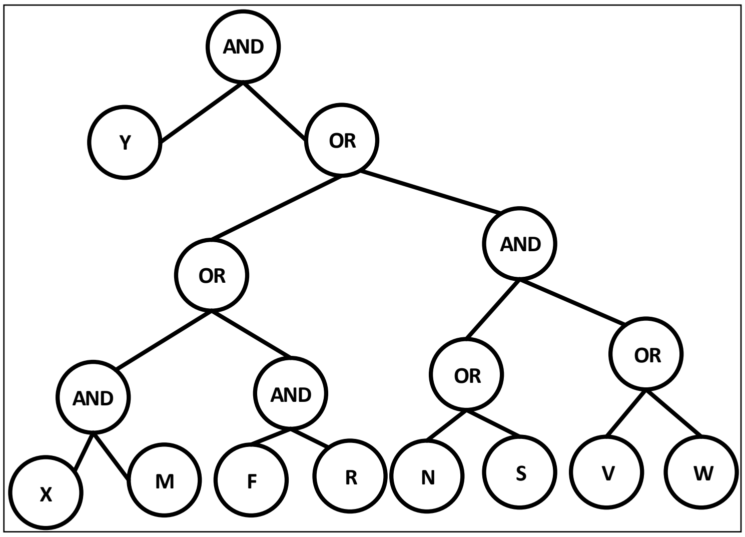


Figure 3.2: A threshold access tree structure

The access tree in **Figure 3.2** is a threshold gate access tree where an interior node is a threshold gate and the leaf nodes are the attributes. An access tree could represent a Boolean formula which contains AND and OR gates instead of threshold gates where a Boolean formula access tree is a special case of a threshold gate access tree (e.g. the AND gate is a (2, 2)-threshold gate while the OR gate is a (1, 2)-threshold gate).

To generate the corresponding LSS matrix with a built-in threshold before enforcing this matrix into the ciphertext in CP-ABE, some steps are needed to derive the LSSS access structure from the above formula using Lewko-Waters Algorithm [113]. This algorithm takes any monotone Boolean formula of an access policy as an input and outputs an LSSS matrix (as illustrated below). First of all, the number of rows in the generated LSS matrix ought to be equal to the number attributes in the formula and equal to the number of leaf nodes in the access tree in ***Figure 3.2***

The steps of the Lewko-Waters algorithm [111, 113] used in this thesis are briefly presented below:

1- The vector $(1,0,\ldots,0)$ is used as the sharing vector. Where the root node of the tree is labelled with vector $\vec{v} = (1)$ and initializes a counter $c = 1$. As shown in ***Figure 3.3.***

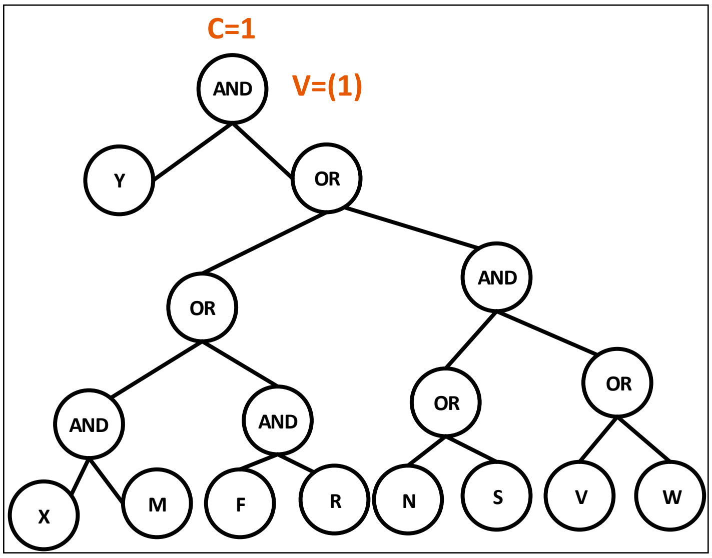


Figure 3.3: The first step of the Lewko-Waters algorithm is to label the root node

2- Then all other nodes are labelled in a specific way. When the labelled node with vector $\vec{v}$ is an OR-gate, its child nodes are labelled with $\vec{v}$ and keeps the counter c without changing.

3- When the labelled node with vector $\vec{v}$ is an AND-gate, the value of c is increased with one, where the value of c represents the length of the vector $\vec{v}$. Therefore, the vector $\vec{v}$ is padded with $0's$ (if it is necessary) at the end to make it of length c. Then its right child node is labelled with $\vec{v}$ concatenated with one. As a result, the right child node is labelled with the vector $\vec{v}|1$ (as illustrated in **Figure 3.4**). While the left child node is labelled with the vector $(0,..,0)|-1$, the length of the vector$(0,..,0)$ depends on the value of c. Therefore, the summation of the right and left children at each level equals to $\vec{v}|0$.

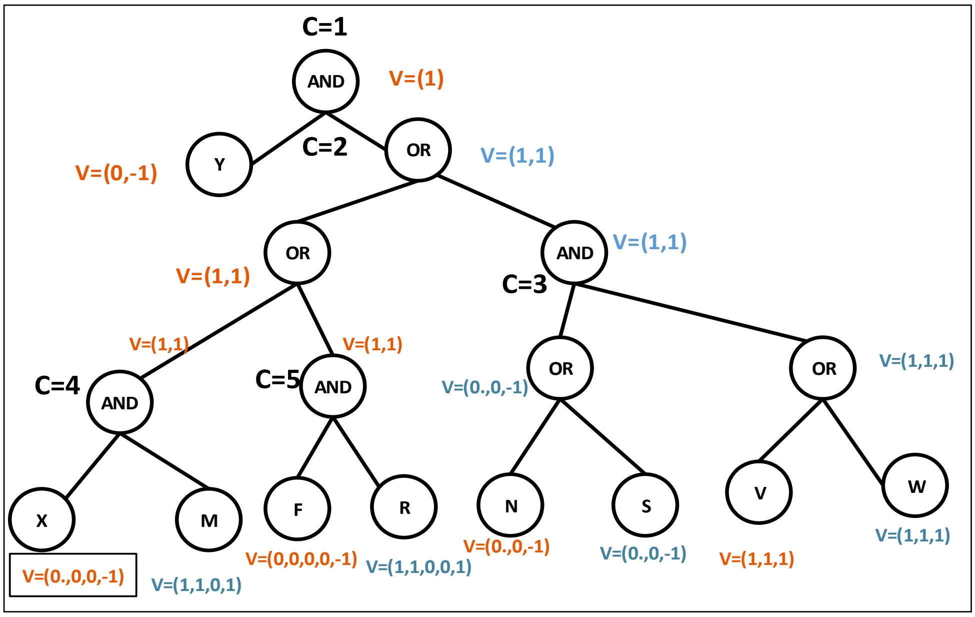


Figure 3.4: Labelling the interior nodes and leaf nodes of the access tree using the Lewko-Waters algorithm

4- After labelling the entire tree, the vectors of the leaf nodes represent the rows of the LSS matrix (as shown in **Equation 3.4**), where the length of all rows must be the same. Therefore, the short rows are padded with 0’s at the end.

$$W = \left[\begin{array}{ccccc} 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{array}\right] \begin{array}{l} \rho(1) = Y \\ \rho(2) = X \\ \rho(3) = M \\ \rho(4) = F \\ \rho(5) = R \\ \rho(6) = N \\ \rho(7) = S \\ \rho(8) = V \\ \rho(9) = W \end{array} \tag{3.4}$$

In this matrix, each authorised subset of the rows (e.g. Y AND (X AND M) includes (1,0,0,0,0) in its span (as in Equation 3.3) on the condition that the below Boolean formula is satisfied by the corresponding attributes.

$$(Y\ AND\ \Big(((X\ AND\ M) OR (F\ AND\ R))\ OR\ ((N\ OR\ S)\ AND\ (V\ OR\ W)\Big))$$

3.5 The Complexity Assumptions

In this section, some assumptions that enable us to build a cryptographic system with advanced security are introduced, particularly those that the CP-ABE schemes are based on. These assumptions are described as below.

3.5.1 Decisional Bilinear Diffie-Helman (BDH) Assumption

Definition 3.2. (Decisional BDH) assumption: A group G denotes a bilinear group of prime order , and a generator of G is g. The parameters a, b, c, z are selected randomly in Z_p. The decisional BDH assumption is that it is still hard for adversary $\mathcal{A}$ to distinguish $(A = g^a, B = g^b, C = g^c, Z = e(g,g)^{abc})$ from $(A = g^a, B = g^b, C = g^c, Z = e(g,g)^z)$ without a non-negligible advantage.

3.5.2 Decisional q-Parallel Bilinear Diffie-Hellman Exponent Assumption

Definition 3.3. (Decisional q-parallel BDHE) assumption [42]: A group G denotes a bilinear group of prime order p, and a generator of G is g. The parameters $a, s, b_1, \dots, b_q$ are selected randomly in Z_p. The decisional q-Parallel BDHE assumption is that once an adversary $\mathcal{A}$ is given:

$$\boldsymbol{y}^{\rightarrow} = g, g^s, g^a, \dots, g^{(a^q)}, g^{(a^{q+2})}, \dots, g^{(a^{2q})},$$

$$g^{s.b_j}, g^{\frac{a}{b_j}}, \dots, g^{\left(\frac{a^q}{b_j}\right)}, g^{\left(\frac{a^{q+2}}{b_j}\right)}, \dots, g^{(a^{2q/b_j})}\ \forall 1 \le j \le q,$$

$$g^{a.s.\frac{b_l}{b_j}}, \dots, g^{\left(a^q.s.\frac{b_l}{b_j}\right)}\ \forall 1 \le j, l \le q, l \ne j \tag{3.5}$$

it is still hard for $\mathcal{A}$ to distinguish $e(g,g)^{a^{q+1}s}$ from a random element $\mathcal{R}$ in G_T. Furthermore, a polynomial time algorithm $\mathcal{B}$ will use the output $z \in \{0,1\}$ of $\mathcal{A}$ to make a guess, and we define the advantage ε of $\mathcal{B}$ to solve the q-Parallel BDHE assumption in G and G_T as:

$$\left|Pr\left[\mathcal{B}\left(\vec{y}, e(g,g)^{a^{q+1}s}\right) = 0\right] - Pr\left[\mathcal{B}(\vec{y}, \mathcal{R}) = 0\right]\right| \geq \varepsilon\,. \qquad (3.6)$$

Once no polynomial time algorithm has a non-negligible advantage to solve the q-Parallel BDHE assumption, we can say that the assumption holds in G.

3.6 Selective Security Model

In this part, the security model in a single-authority access control system is considered, where an attempt has been made to decrypt a ciphertext that is encrypted over an access policy, by the adversary $\mathcal{A}$ with secret keys that have attributes which do not satisfy that access policy. This game includes the roles played by the challenger who is in charge of generating secret keys and hiding the details, and an adversary $\mathcal{A}$ who can ask for any secret keys whose attributes cannot satisfy the access policy that is later embedded into a challenged ciphertext. The security game is defined as below:

Init. The adversary sends its challenge access policy W^* to the challenger.

Setup. The challenger generates the system parameters by running the Setup algorithm. Then, the challenger sends the public parameters to the adversary and keeps the master key secret.

Query 1. Many secret keys queries are made by the adversary using an attribute set that does not match the access policy W^*

Challenge. The challenger receives two equal-length messages ($M_0\ and\ M_1$) from the adversary. One of the submitted messages is chosen randomly by the challenger who encrypts it under W^*. This message is denoted as M_b where $b \in \{0,1\}$. Then the encrypted message is sent to the adversary.

Query 2. More secret keys queries are made by the adversary with the same restriction of the Query 1 phase where none of these queries match the challenged access policy examined in the Challenge phase.

Guess. The adversary outputs its guess $\bar{b}$ for b. In this game, the advantage of an adversary is described as: $\left[\,\bar{b} = b\right] - \frac{1}{2}$.

3.7 Waters' CP-ABE Scheme (W-CP-ABE)

Based on Waters' scheme [42], the algorithms of CP-ABE are defined below:

$\boldsymbol{Setup(U)} \rightarrow \boldsymbol{(PK, MSK)}$. The setup algorithm is run by the attribute authority which takes the number of attributes in the system as input. The authority generates public and master keys (i.e. PK and MSK respectively):

$$PK = g, e(g,g)^{\alpha} , g^{a} , h_1, \ldots, h_U .$$

$$MSK = a, g^{\alpha}$$

Where, it chooses G as a group of prime order p, g is set as a generator, $e : G \times G \rightarrow G_T$ is a bilinear map and U is the number of group elements $h_1, \ldots, h_{\mathrm{U}} \in G$ that are randomly chosen and associated with the U attributes in the system. In addition, $\alpha, a \in Z_p$ are randomly chosen exponents.

$\boldsymbol{Encrypt(PK, (W, \rho), M)} \rightarrow$ **CT**. This algorithm is run by a data owner. The public key PK, a message M and LSSS access structure (W, ρ) are taken as inputs. Where, W is a matrix and ρ is the function that maps rows of W to attributes. A random vector $\vec{v} = (s, y_1, \ldots, y_n) \in Z_p^n$ is chosen randomly. The values of this vector's elements will be used to share the secret s. The value $\lambda i = \vec{v} \cdot W_i$ is calculated for $\forall i = 1 \; to \; l$, where W_i is the i^{th} row vector of W. $r_1, \ldots, r_l \in Z_p$ are blinded random numbers. The ciphertext is:

$$CT = ((W, \rho), C = Me(g,g)^{\alpha s}, C_0 = g^{s}, (C_1 = g^{a\lambda_1} h_{\rho(1)}^{-r_1} ,$$

$$D_1 = g^{r_1}), \ldots, (C_l = g^{a\lambda_l} h_{\rho(l)}^{-r_l}, D_l = g^{r_l})$$

$\boldsymbol{KeyGen(MSK, S)} \rightarrow$ **SK**. The user secret key is generated using this algorithm according to a set of attributes S which is taken together with master secret key MSK as inputs. The user's private key is:

$$SK = (K = g^{\alpha} g^{at}, L = g^{t}, \forall x \in S \quad K_x = h_x^{t}) .$$

Where, $t \in Z_p$is a random number. This algorithm is executed by the attribute authority.

$\boldsymbol{Decrypt\ (CT, SK)} \rightarrow \boldsymbol{M.}$ A ciphertext CT and the user's private key are the algorithm's inputs. Once the user's attributes in its secret key satisfy the access structure in the ciphertext, the output will be the recovered message M. $\{\omega_i \in Zp\}_{i \in I}$ are supposed to be a set of constants such that when $\{\lambda_i\}$ are valid shares of a secret s which corresponds to W_i, then $\sum_{i \in I} \omega_i \lambda_i = s$. Firstly, the decryption algorithm computes B_1 as:

$$B_1 = \frac{e(C_0\,, K)}{(\prod_{i\in I}\left(e(C_i, L)e\left(D_i, K_{\rho(i)}\,\right)\right)^{\omega_i}}$$

$$= \frac{e(g,g)^{\alpha s}e(g,g)^{\,ast}}{\prod_{i\in I}\left(e(g,g)^{ta\lambda_i}e(g,h_{\rho(i)}\,)^{-r_i t}e(g,h_{\rho(i)})^{r_i t}\right)^{\omega_i}}$$

$$= \frac{e(g,g)^{\alpha s}e(g,g)^{\,ast}}{\prod_{i\in I}(e(g,g)^{ta\omega_i\lambda_i})}$$

$$B_1 = e(g,g)^{\alpha s}$$

Then the decryption algorithm can compute the message M as:

$$M = \frac{C}{B_1} = \frac{Me(g,g)^{\alpha s}}{e(g,g)^{\alpha s}}$$

Notably, Waters' scheme [42] does not support the revocation process. In this thesis, we adopt this CP-ABE scheme and extend it to resolve many of its limitations.

3.7.1 Security Proof of BW-CP-ABE

To prove that Waters' scheme [42] is secure, the author encounters one obstacle. This obstacle is indicated when the same attribute is duplicated in the challenge access matrix W^*. That means multiple rows in the access matrix represent the same attribute. To resolve this issue, the author uses the term of the decisional q-Parallel Bilinear Diffie-Hellman assumption instead of the term of the parallel BDHE assumption in order to assign multiple rows in W^* to a one-element group that corresponds to an attribute. Where, q is a polynomial degree which is embedded into a single group element as in Gentry's reduction [114]. The author proves:

Theorem 1: Once the decisional q-Parallel BDHE assumption holds, all polynomial time adversaries have negligible time to selectively break the proposed CP-ABE scheme (i.e. there is no adversary that can break the system), where the challenge LSSS matrix is $W^*(l^* \times n^*)$ with $l^*, n^* \le q$.

In the selective security game, let $\mathcal{A}$ be an adversary with non-negligible advantage against Waters' scheme [42]. The adversary $\mathcal{A}$ selects W^* as a challenge matrix where each of its row number l^* and column number n^* is less than or equal to q. The decisional q-Parallel BDHE problem is played by a simulator $\mathcal{B}$ as follows:

Init. As in (**Section 3.5.2**), $\vec{y}$, T are taken by the simulator $\mathcal{B}$, while the adversary $\mathcal{A}$ sends (W^*, ρ^*) to $\mathcal{B}$ where W^* has l^* rows and n^* columns.

Setup. An element $\acute{\alpha}$ is randomly chosen by the simulator $\mathcal{B}$ where $\alpha' \in Z_p$. The simulator $\mathcal{B}$ sets $e(g,g)^{\alpha} = e(g,g)^{\acute{\alpha}}.e\left(g^{a}, g^{a^q}\right) = e(g,g)^{\acute{\alpha}}.e(g,g)^{a^{q+1}}$ that means implicitly $\alpha = \acute{\alpha} + a^{q+1}$.

In terms of computing each group element h_x that corresponds to an attribute x where $1 \leq x \leq U$, a number z_x is selected randomly for each x. Let $I^*$is a set where $I^* = \{i: \rho^*(i) = x\}$:

$$h_x = g^{z_x} \prod_{i \in I} g^{aW^*_{i,1}/b_i}.g^{a^2W^*_{i,2}/b_i} \dots g^{a^{n^*}W^*_{i,n^*}/b_i}$$

Notably, when $I^*$is an empty set, then $h_x = g^{z_x}$.

Query 1. Many secret key queries are made by the adversary $\mathcal{A}$ in which attribute set S does not satisfy the matrix W^*.

In the context of LSSS, as a result of querying an unauthorized set of attributes by the adversary $\mathcal{A}$, the simulator $\mathcal{B}$ finds a vector $\vec{w} = (w_1, w_2, \dots, w_{n^*}) \in Z_p$ where $w_1 = -1$, and $\forall i, \rho^*(i) \in S$, $\vec{w}.W_i{}^* = 0$. Moreover, the simulator $\mathcal{B}$ randomly selects $r \in Z_p$.

Implicitly, the simulator $\mathcal{B}$ defines t as:

$$t = r + w_1.a^q + w_2.a^{q-1} + \dots + w_{n^*}.a^{q-n^*+1}$$

Therefore, to compute L:

$$L = g^t = g^r \prod_{i=1,..,n^*} (g^{a^{q-i+1}})^{w_i}$$

To generate K:

$$K = g^{\acute{\alpha}}.g^{ar} \prod_{i=2,..,n^*} \left(g^{a^{q-i+2}}\right)^{w_i}$$

At this step, for each $x \in S$ that is not used in the access structure, K_x is calculated as:

$$K_x = L^{z_x}$$

When $x \in S$ and x is used in the access structure, computing K_x becomes a hard task. As a restriction, the terms in the form $g^{a^{q+1}}/_{b_i}$ must not exist. On the other hand, since $\vec{w}.{W_i}^* = 0$, all these terms will be cancelled. Therefore, the simulator generates K_x as:

$$K_x = L^{z_x} \prod_{i \in I} \prod_{j=1,..,n^*} \left(g^{(a_j/b_i)}{}^{r} \prod_{k=1,..,n^*, k \neq j} (g^{a^{q+1+j-k}/b_i})^{w_k} \right)^{{W_{i,j}}^*}$$

Challenge. To build the ciphertext components, two equal-length messages M_0, M_1 are prepared by the adversary $\mathcal{A}$ and sent to the simulator $\mathcal{B}$. The simulator $\mathcal{B}$ selects one of them M_c where $c \in \{0,1\}$. The simulator $\mathcal{B}$ generates the ciphertext components $C = M_c T.e(g^s, g^{\acute{\alpha}})$ and $\acute{C} = g^s$.

To generate the component C_i, the simulator $\mathcal{B}$ selects a vector $\vec{v}$ where the first element in this vector will be the secret s that needs to be shared. So, $\vec{v} = (s, sa + \acute{y}_2, sa^2 + \acute{y}_3, \ldots, sa^{n-1} + \acute{y}_{n^*}) \in Z_p^{n^*}$, where $\acute{y}_2, \ldots, \acute{y}_{n^*}$ are randomly chosen. Moreover, the simulator randomly chooses $\acute{r}_1, \ldots, \acute{r}_l$.

The simulator generates a set R_i, where $i = 1, \ldots, n^*$. This set contains all indices of rows that are assigned to similar attributes as row i (i.e. $\rho^*(i) = \rho^*(k)\ where\ i \neq k$). Therefore, the generated ciphertext components are as follows:

$$D_i = g^{-\acute{r}_\iota}.g^{-sb_i}$$

$$C_i = h_{\rho^*(i)}^{\acute{r}_\iota} \left(\prod_{j=2,\ldots,n^*} (g^a)^{W_{i,j}^* \acute{y}_J} \right).(g^{s.b_i})^{-z_{\rho^*(i)}}.\left(\prod_{k \in R_i} \prod_{j=1,\ldots,n^*} (g^{a^j s.(b_i/b_k)})^{W_{k,j}^*} \right)$$

Query 2. It is the same procedure as Query 1.

Guess. At this phase, the adversary $\mathcal{A}$ outputs its guess $\acute{c}$ of c. If $\acute{c} = c$, then the simulator $\mathcal{B}$ outputs 0 that means that $T = e(g,g)^{a^{q+1}s}$. In this case, we have:

$$Pr\left[\mathcal{B}\left(\vec{y}, T = e(g,g)^{a^{q+1}s}\right) = 0\right] = \frac{1}{2} + \mathrm{Adv}_{\mathcal{A}}$$

Otherwise, the simulator outputs 1 which means that T is a random group element in G_T and M_c is totally concealed from the adversary. In this case, we have:

$$Pr\ [\mathcal{B}(\vec{y}, \mathrm{T} = \mathcal{R}) = 0] = \frac{1}{2}$$

As a result, the decisional q-Parallel BDHE game can be played by a simulator $\mathcal{B}$ with non-negligible advantage.

3.8 Summary

In this chapter, some mathematical principles required in our project have been briefly described. The following explanation elaborates the main points:

- In terms of elliptic cryptography, the discrete logarithm problem has not been solved yet by any known sub-exponential type algorithm.
- The Computational Diffie-Hellman Assumption (CDHA) is when the Diffie-Hellman function is ***hard to be computed*** by any passive attack as defined below: $DH(g^a, g^b) = g^{ab}$ where it is hard to compute g^{ab}.
- The Decisional Diffie-Hellman Assumption (DDHA) is ***hard to distinguish*** between two tuples $\langle g^a, g^b, g^{ab}\rangle$ and $\langle g^a, g^b, g^c\rangle$.
- The difficult issues that most elliptic-curve cryptographic schemes are based on are DLP and CDH problems. These problems can provide a sufficient level of security if the related parameters are chosen properly. While the security assumption that the pairing based cryptography relies on is the decisional Bilinear Diffie-Hellman problem and up to now there are no known attacks breaking this problem.
- Any access policy can be transformed into one of these methods: 1) a monotone AND-gate access structure, 2) a (t, n)-threshold access tree and 3) LSSS matrices.
- The access policy that is represented as an LSSS matrix, is difficult to be comprehended by anyone who is not an expert. In addition, this tool is highly expressive.

Chapter 4

The Proposed Single Authority Access Control Scheme

Chapter 4: The Proposed Single Authority Access Control Scheme

4 Introduction

The design of our proposed single authority CP-ABE scheme is discussed in this chapter, where our proposed scheme extends the relevant existing techniques to resolve the inherent problems in CP-ABE, which is users' credential management according to access privilege customization. The novelty of our collusion-resistant scheme is to drive the access privileges in a specific way by updating the access policy as well as user revocation.

Therefore, as a first step towards solving the attribute revocation problem besides tackling the mentioned issues, we present a technique to assign heavy tasks to a cloud service provider. Once the attribute revocation process needs to be enabled, updating the access policy will be carried out by the data owner, while the cloud server will be responsible for re-encrypting the ciphertext components that the attributes are embedded in. In this case, the cloud server will be in charge of re-encrypting the ciphertext without any information leaking to the server. Finally, in this scheme, security and theoretical performance analysis is carried out showing that our scheme can securely and efficiently offload the computational burden from the attribute authority and the data owner.

The remainder of this chapter is structured as follows. Section 4.1 presents the requirements and the security assumptions that our scheme is based on. Section 4.2 discusses the general explanation of our proposed scheme, the scheme entities, the relationship between these entities and the scheme algorithms. Section 4.3 describes the scheme analysis in terms of security and performance. The experimental results are discussed in Section 4.4. Finally, Section 4.5 outlines our conclusions.

4.1 The Scheme Security Requirements and Assumptions

In this section, the security assumption of each party's role and requirements in our proposed single-authority CP-ABE system is defined. The main security assumptions are discussed below:

1- The cloud server is honest to carry out the tasks that are assigned to it, but curious to find out as much unauthorized information as possible. Moreover, we apply the strong security assumption where it is possible that the cloud server colludes with revoked users. Furthermore, the access control is enforced cryptographically by embedding access policies in the ciphertext without cloud intervention.

2- The attribute authority is assumed to be a fully trusted entity, but it can be attacked by an adversary.

3- Any user can gain the encrypted data stored on the cloud server. However, only the users whose attributes satisfy the access policy and whose identities are not in the revocation list, can properly decrypt the corresponding ciphertexts. Furthermore, the system assumes that there are misbehaving users who try to collude with other entities in the system, excluding the data owner, or with each other to access unauthorized data.

4- The data owner is a fully trusted entity.

5- A proxy server is a minimal-trusted entity which receives a different proxy key upon each user revocation event from the attribute authority to update one of the non-revoked users' secret key components in order to recover the message. Therefore, the potential risk of the proxy server colluding with revoked users is minimised. It is not allowed for this entity to decrypt data because it does not have an attribute decryption key.

Based on the above assumptions, the security requirements that our proposed scheme ought to achieve, are stated as follows:

1- Data confidentiality: Data content is protected against access by any unauthorized users or the cloud server.

2- Fine-grained access control: Different users with various privileges must access different ciphertexts.

3- Collusion resistance: The cloud server is prevented from colluding with revoked or malicious users to gain unauthorized data by combining their information.

4- Forward security: Any revoked user is forbidden to decrypt any new ciphertext after leaving the system.

4.2 Our Proposed Scheme

Our proposed CP-ABE system involves five entities. These entities are responsible for running seven algorithms as shown in **Figure 4.1**. These entities and algorithms are described in detail below in separate sub-sections.

4.2.1 The Scheme Entities

The entities of our proposed scheme and the relationship between them are presented in this section as follow:

Attribute authority. This trusted entity is responsible for generating the system parameters, such as a master key and a public key. In addition, it is in charge of creating a secret key for each user in the system based on the user's attributes. Moreover, upon each revocation event and according to the list of revoked users, the attribute authority issues a proxy key to the proxy server to be introduced later.

Data owners. This party defines an access policy that describes who can access to its data as well as encrypting those data under this access policy. Firstly, a data owner uses a symmetric encryption technique (e.g. AES) to encrypt its data. After that, the owner encrypts the symmetric key under its access policy using CP-ABE by selecting a random value as a secret which is shared using the linear secret sharing scheme (LSSS) technique to generate some values associated with each corresponding attribute in the ciphertext according to the owner's access policy. Finally, the encrypted data is outsourced to the cloud including the data ciphertext, the CP-ABE ciphertext and the access policy. In addition, the data owner is responsible for updating the access policy.

Data users. Each of them receives its secret key which contains its attributes, from the attribute authority. This secret key is used to decrypt any ciphertext uploaded by the data owner whenever the user's attributes satisfy the owner's access policy. Thus, it can recover the plaintext.

Cloud server. This server is an untrusted entity which stores and shares encrypted data that is still useless information to the server even if it colludes with some malicious data users. Those data can be downloaded by any data users. Since the cloud server is untrusted, assigning some tasks to this entity is a critical challenge because the cloud could be curious to extract secret information from the stored data to gain some benefits from the data owner's competitors. However, to leverage the cloud resources, in our model, we assign heavy missions to the cloud to partially encrypt data that is uploaded by a data owner and gives access to that data to various data users. Additionally, the cloud server is able to securely transform the ciphertext components related to the old access policies to the new ones according to the new policies.

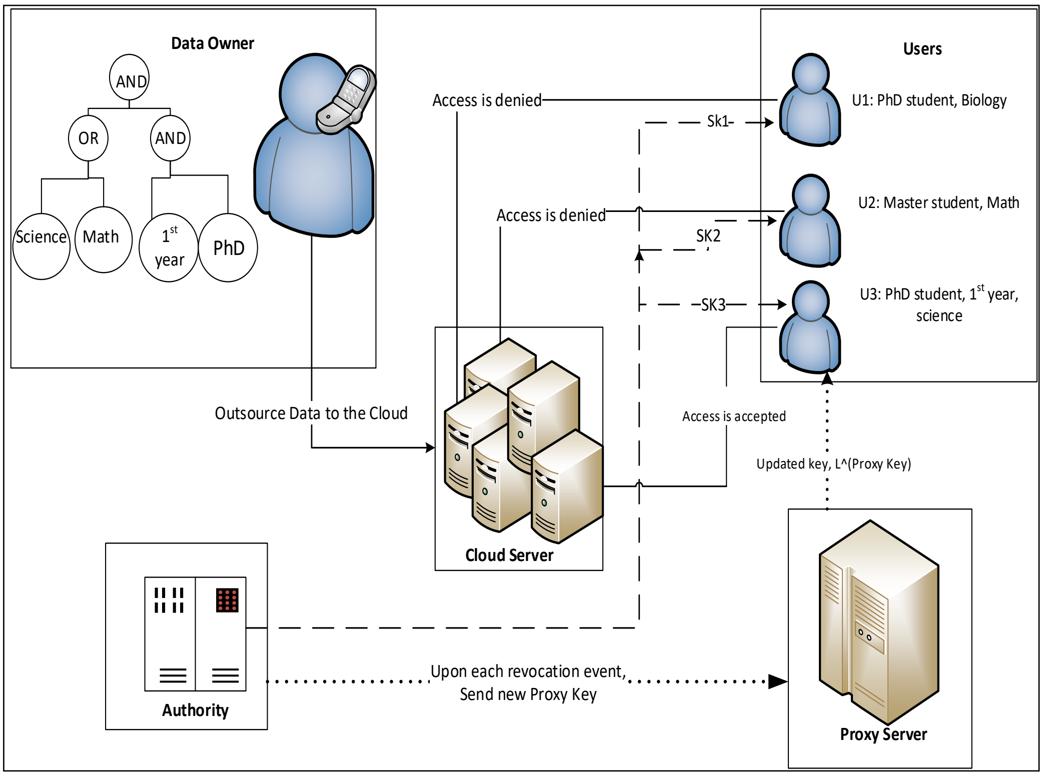


Figure 4.1: The architecture of the proposed system scheme

Proxy server: The proxy server is a minimal-trusted server which is provided with a proxy key by the attribute authority when each revocation process takes place. This key is embedded in each non-revoked user's key which in turn helps these users to recover the message and limits the control privileges of the cloud server to prevent it from colluding with the revoked users.

The relationship and interaction among the entities described above will be explained in the subsequent two sub-sections.

4.2.2 Scheme Entities Relationship

In this subsection, a summary of our proposed work is briefly presented to describe the responsibilities of the system entities and the relationship among them as well as some relevant, existing tools which our scheme has extended. The scheme works as follows:

- First, the system parameters are generated by the attribute authority. These parameters are used in all scheme algorithms. When the data owner decides to outsource his data to the cloud server, he encrypts a message into two separate ciphertext components (C, C_0). These components are encrypted with a message and the secret to be shared, respectively. The other components, which have to be associated with secret encrypted shares, are securely generated by the cloud server, where these encrypted shares have already been distributed over a monotone access structure realized by a LSSS matrix generated over a set of legitimate attributes, before encrypting by the data owner using a traditional cryptographic technique. After outsourcing the resulting encrypted shares by the data owner, these shares are encrypted again to be associated with the authorized attributes by the cloud server using CP-ABE. In that case, only the eligible users can decrypt and recover the message correctly if they meet the following conditions: (a) they already received their secret keys from the attribute authority, (b) their legitimate attributes satisfy the access policy, and (c) they are not in the revocation list, which is indirectly derived by the proxy server. Granting the control privileges of these encrypted shares to the cloud server results in that most operations of our technique are delegated to the cloud server while guaranteeing no information leakage to the server. These operations linearly increase with the number of attributes and the frequency of revocation events.

- Then, some extra layers of security are added where the attribute and user revocation problems are addressed by extracting some ideas from some of the relevant existing techniques. Once a user revocation event occurs, the attribute authority sends a set of the revoked users' identities and the corresponding secret shares as a proxy key to the proxy server, which in turn updates a part of the secret key for only the authorized users. In that case, revoking a user happens by developing and adjusting the technique in [57] to invalidate the key which a revoked user already has. It prevents the revoked users from colluding with the cloud server. Therefore, our scheme resists against any collusion attacks at the same time. It customises users' privileges by updating a policy. In our proposed scheme, this secret is used to generate new shares. This results in expanding the capability of handling the attribute revocation process and adds the ability of elevating user privileges, where the updating process happens in two directions. The first one is to generate a new LSSS matrix which corresponds to the updated policy without changing the value of the data owner's secret. This is carried out by the data owner. The second orientation is to update the ciphertext components which are already stored on the cloud after calculating the new ones. These updating and calculating processes are performed by the cloud server. It leads to exploiting cloud storage and sharing services while mitigating computation and communication overheads.

4.2.3. A Scheme Overview

To build an efficient, trustworthy access control system for storing data on untrusted environments, our single-authority CP-ABE scheme (as shown in **Figure 4.1**) is proposed. In particular, a single, fully trusted attribute authority generates the public system parameters that are later used by all scheme entities to run the system algorithms. Once the data owner needs to outsource its data to a cloud server, encrypting those data over a set of selected, combining attributes known as an access policy is the first procedure considered. In terms of mitigating the encryption burden on the data owner, some expensive operations are securely outsourced to the cloud server. After generating ciphertexts, the data owner stores them on the cloud. Whenever the data owner decides to grant some data users more privileges or withdraw some privileges from others, it can feasibly change its access policy, generate the updated components and send them to the cloud server which in turn re-encrypts the stored ciphertexts.

For other entities such as users and a proxy server, the procedures are often restricted, in which the attribute authority generates different decryption secret keys for all authorised data users depending on their attributes. In addition, once the attribute authority decides to revoke some users who have misbehaved, or their services are expired, it generates a revocation list, which contains a limited number of revoked users, and issues a new proxy key based on such a list. This proxy key is sent to the proxy server that updates only the secret keys of the non-revoked users. In terms of users, a user can be authorised to decrypt a ciphertext if its attributes satisfy the data owner's access policy and its identity is not in the revocation list.

4.2.4. Scheme Algorithms

Our proposed scheme is constructed by seven algorithms which are defined below. The parties who run these algorithms are illustrated in **Figure 4.1** and the main notations used are listed in **Table 4.1**.

$\boldsymbol{Setup(U) \rightarrow (PK, MSK)}$: The attribute authority randomly chooses a polynomial P of a degree c over Z_p to use it for blinding each user's secret key which in turn facilitates revocation of a set of users. Moreover, the authority generates a public key PK (as shown in **Figure 4.2**) which is published for access by all the scheme parties, and a master secret key MSK kept secret to itself, i.e. $PK = (g, e(g,g)^{\alpha} , g^{a} , h_1, \ldots, h_U)$ and $MSK = (a, g^{\alpha}, P)$.

Table 4.1: Main notations used in our proposed single authority CP-ABE scheme.

Symbol	Description
$\boldsymbol{PK}$	The system public key.
$\boldsymbol{MSK}$	The system master key.
$\boldsymbol{SK}$	The user secret key.
$\boldsymbol{P}$	A secret polynomial of degree c which is selected by the attribute authority.
$\boldsymbol{g}$	A generator of an elliptic curve group G.
$\boldsymbol{p}$	A large prime number that represents the order of group G.
$\boldsymbol{\alpha}, \boldsymbol{a}$	Randomly selected exponents belonging to the finite field Z_p.
$\boldsymbol{U}$	The number of attributes in the system.
$\boldsymbol{h_1}, \ldots, \boldsymbol{h_U}$	Random group elements representing the corresponding attributes.
$\boldsymbol{W}$	The $l \times n$-LSS matrix that represents the access structure. Here, the threshold of the access policy is embedded into it. In addition, l is the number of rows which represent the attributes, and n is the number of columns (i.e. it is the same value of the counter c in **pp 48, Section 3.4.1**).
$\boldsymbol{\rho}$	A function mapping each row in $\boldsymbol{W}$ to the corresponding attribute.
$\boldsymbol{M}$	The message $M \in G_T$ which is randomly chosen to represent the key to be encrypted by CP-ABE.
$\boldsymbol{s}$	The secret to be shared, which is selected by the data owner.
$\boldsymbol{R}$	The revocation list created by the attribute authority. This list contains the identities $\{u_i\}$ of the revoked users, with $i \in \{1, \ldots, c\}$ and c as the selected, secret polynomial degree of P.
$\boldsymbol{P(u_i)}$	The random share which is extracted from polynomial $\boldsymbol{P}$ selected by the attribute authority for each user $\boldsymbol{u_i}$**.** This share is used later to de-activate the key when the user $\boldsymbol{u_i}$ is revoked.
$\boldsymbol{\bar{\lambda}_i}$	A plain secret share.
$\boldsymbol{\hat{\lambda}_i}$	The encrypted version of $\bar{\lambda}_i$.

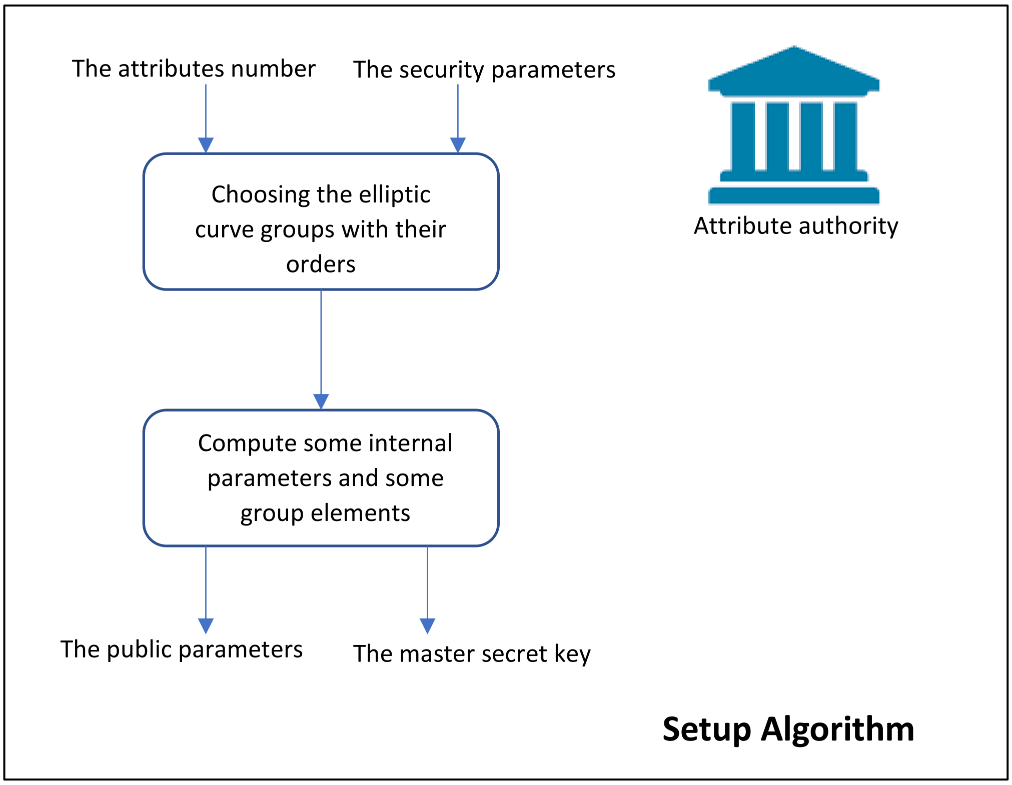


Figure 4.2: The flowchart of the Setup algorithm

Here, the input to this algorithm is U which represents the number of attributes in the system. g is a generator of the selected group G with $|G| = p$, and $h_1, \ldots, h_U \in G$ are random group elements associated with the U attributes in the system. The exponents $\alpha, a \in Z_p$ are randomly selected.

$\boldsymbol{Encrypt}(\boldsymbol{PK}, (\boldsymbol{W}, \boldsymbol{\rho}), \boldsymbol{M}) \rightarrow \boldsymbol{CT}$: A data owner, before migrating its data to the cloud server, encrypts a message M (where $M \in G_T$), using the public key PK and LSSS access structure (W, ρ) specified in **Section 3.7**. To execute the encryption, the owner selects a random vector $\overrightarrow{v} = (s, y_2, \ldots, y_n)$ where $y_2, \ldots, y_n \in Z_p$ are used to share the secret s. The algorithm computes each value $\bar{\lambda}_i$ as $\bar{\lambda}_i = \vec{v} \cdot W_i$, where W_i is the i^{th} row vector of W $(1 \leq i \leq l)$. These shares $\{\bar{\lambda}_i\}_{i \in l}$ will be encrypted by the owner before outsourcing them to the cloud (as shown in **Figure 4.3**). Thus, the ciphertext is outsourced to the cloud server as: $CT = ((W, \rho), C = Me(g, g)^{\alpha s}, C_0 = g^{s})$, together with two other vectors. The first one is $\overrightarrow{\tilde{v}} = (\hat{\lambda}_1, \ldots, \hat{\lambda}_l)$, which represents the vector of the encrypted secret shares that will be embedded by the cloud server in the ciphertext components associated with the attributes (as illustrated in **Figure 4.4**). These components are then published by the cloud as: $\{ \grave{C}_\imath = g^{a\hat{\lambda}_i} h_{\rho(i)}^{-r_i}, \grave{D}_\imath = g^{r_i}\}_{i=1}^{l}$where the blind numbers $r_1, \ldots, r_l \in Z_p$ are chosen randomly by the cloud server.

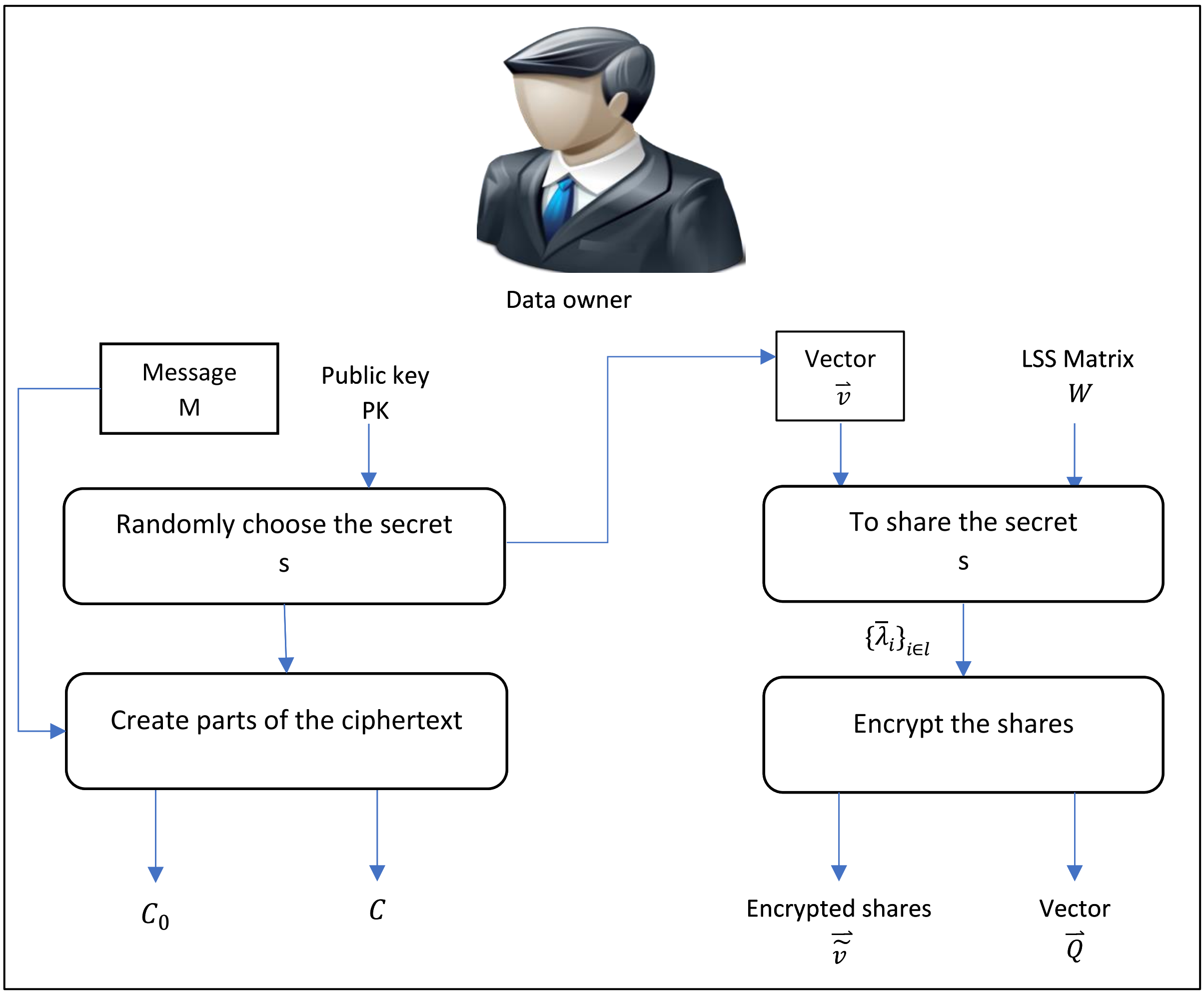


Figure 4.3: The steps of the Encrypt algorithm carried out by the data owner.

The second vector is $\overrightarrow{Q} = (g^{-af_1}, \dots, g^{-af_l})$, in which its elements correspond to the encrypted shares in $\vec{\tilde{v}}$. Both $\vec{\tilde{v}}$ and $\overrightarrow{Q}$ are used to recover the plain-shares $\{\bar{\lambda}_i\}_{i\in I}$ by authorized users. For instance, the first element g^{-af_1} in $\overrightarrow{Q}$ is used with the first encrypted share $\hat{\lambda}_1$ in $\vec{\tilde{v}}$ to recover the first plain share $\bar{\lambda}_1$. Here, $\{f_i \in Z_p\}_{i=1}^{l}$ are randomly selected by the data owner for encrypting the plain-shares $\{\bar{\lambda}_i\}_{i\in I}$ in the vector $\overrightarrow{v.W}$. So, an encrypted share $\hat{\lambda}_i$ is resulted from $\hat{\lambda}_i = \bar{\lambda}_i + f_i \bmod p$ for $i = 1$ to l. The elements of $\overrightarrow{Q}$ are included as ciphertext components. Finally, the ciphertext is published by the cloud server as:

$$CT = (C = Me(g,g)^{\alpha s}, C_0 = g^s, \left\{\grave{C}_\iota = g^{a\hat{\lambda}_i}\, h_{\rho(i)}^{-r_i}\ , \grave{D}_\iota = g^{\,r_i}, T_i = g^{-af_i}\right\}_{i=1}^{l})$$

Since the ciphertext components associated with attributes are partially encrypted by the cloud server, the outsourcing of computation offered by the cloud is efficiently exploited.

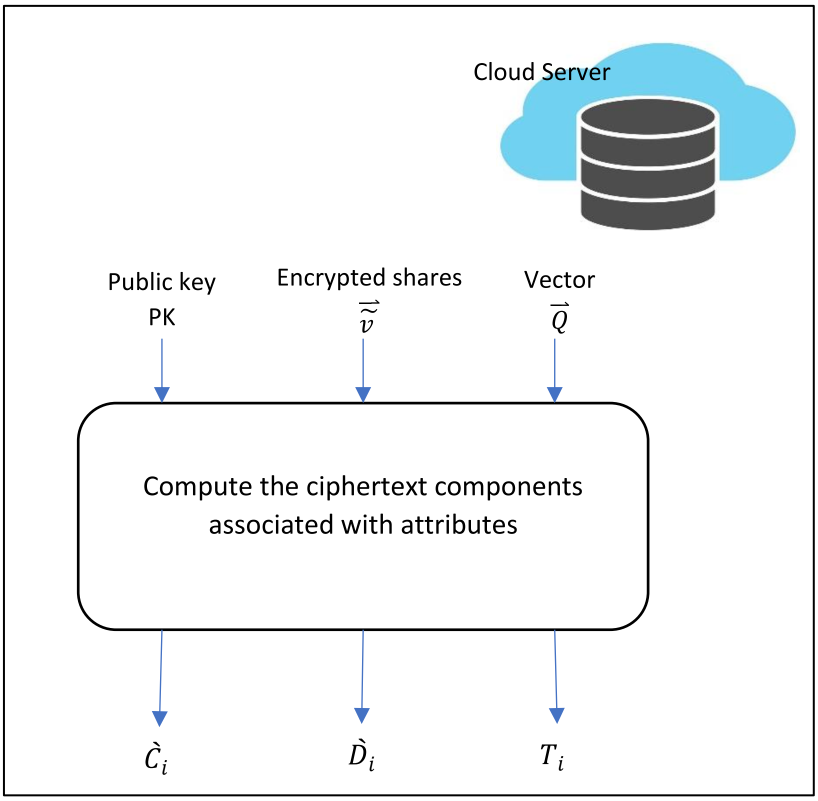


Figure 4.4: The steps of the Encrypt algorithm carried out by the cloud server.

$\boldsymbol{KeyGen(MSK, S) \rightarrow SK}$: The master secret key MSK and attribute set S are used by the attribute authority to generate a secret SK_{u_k} for each user u_k, which is securely received by u_k via a secure channel (as illustrated in **Figure 4.5**). The secret key is defined as:

$$SK_{u_k} = (K = g^{\alpha} g^{atP(0)}, L = g^{at}, \acute{L}_1 = g^{tP(u_k)}, \forall x \in S \quad K_x = h_x^{tP(u_k)})$$

Here, the exponent $t \in Z_p$ is randomly chosen, and the exponent $P(u_k)$ represents a random share selected by the attribute authority for each user u_k to recover the secret $P(0)$ that needs $c + 1$ shares to recover. To ensure the collusion resistance property, upon each user revocation event, the attribute authority generates a proxy key that contains a set of pairs of c secret shares (i.e. points of the secret polynomial) and the corresponding identities of the revoked users $u_i, i \in \{1,2, \dots, c\}$. These shares are embedded by the proxy server in a piece of the user's secret key to transform it for decryption. Therefore, the proxy key will be as follows:

$Proxy_{Key} = \forall u_i \in R: < u_i, P(u_i) >$, where R is the set of revoked users.

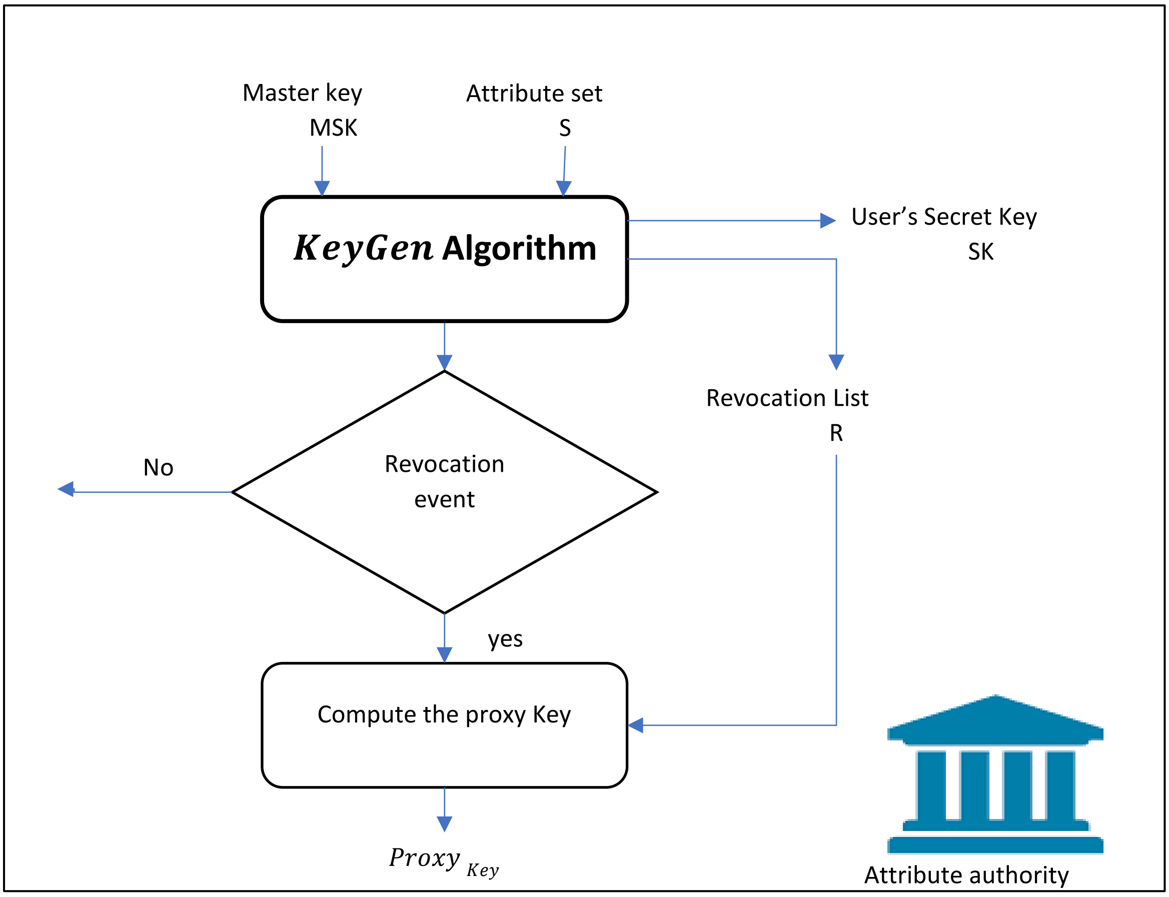


Figure 4.5: The steps of the KeyGen algorithm

$\boldsymbol{KeyTransform(L, Proxy_{Key})} \rightarrow \boldsymbol{L^{Proxy_{key}}}$: The proxy server takes the component L of the user's secret key and then transforms it by embedding the shares of the revoked users in that piece using its $Proxy_{Key}$. Using the revoked shares will prevent them from using these shares to recover the associated plaintext. In addition, the proxy uses its information and the identity of the non-revoked user u_k to compute:

$$\forall i, j \in \{1, \ldots, c\}, k \notin \{1,\ldots,c\}, \qquad \lambda_i = \frac{u_k}{u_k - u_i} . \prod_{j \neq i} \frac{u_j}{u_j - u_i} \qquad (4.1)$$

For every non-revoked user u_k, the proxy calculates λ_k using the equation (4.1) and then sends it to u_k. In this case, the collusion attack happens only when the revoked users collude with the cloud and proxy servers together.

$$\forall u_k \notin R, (L_k)^{Proxy_{key}} = (\, L_k \,)^{\sum_{i=1}^{c} \lambda_i P(u_i)} = (g^{at})^{\sum_{i=1}^{c} \lambda_i P(u_i)} = g^{at \sum_{i=1}^{c} \lambda_i P(u_i)}$$

$\boldsymbol{Decrypt\ (CT, SK, L^{Proxy_{key}})} \rightarrow \boldsymbol{M}$: The decryption algorithm inputs are a ciphertext CT, the user's private key and the output of the $KeyTransform$ algorithm. Once a user's attributes satisfy the access structure in the ciphertext and it is not on the revocation list (as shown in **Figure 4.6**), it can recover the message M as detailed below. The values

$\{\omega_i \in Z_p\}_{i\in I}$ are supposed to be a set of constants such that when $\{\bar{\lambda}_i\}_{i\in I}$ are valid shares of a secret s which correspond to W_i, then $\sum_{i\in I}\omega_i\bar{\lambda}_i = s$. The decryption algorithm computes M as follows:

$$B_1 = \frac{e(C_0\,,K)}{(\prod_{i\in \grave{I}}\left(e\left(\grave{C}_\iota,\acute{L}_1{}^{\lambda_k}\right)e\left(\grave{D}_\iota,K_{\rho(i)}{}^{\lambda_k}\right)e\left(\,T_i,\acute{L}_1{}^{\lambda_k}\right)\right)^{\omega_i}).e(C_0,L^{Proxy_{key}})}$$

$$= \frac{e\left(\,g^s\,,g^{\,\alpha}g^{\,atP(0)}\right)}{(\prod_{i\in \grave{I}}\left(e\left(g^{a\widehat{\lambda}_\iota}.h_{\rho(i)}^{-r_i},g^{tP(u_k)\lambda_k}\right)e\left(g^{\,r_i},\;h_{\rho(i)}{}^{\,tP(u_k)\lambda_k}\right)e(g^{-aF_i},g^{tP(u_k)\lambda_k})\right)^{\omega_i}).e(g^s,g^{at\sum_{i=1}^{c}\lambda_iP(u_i)})}$$

$$= \frac{e(g,g)^{\alpha\,s}e(g,g)^{\,a\,stP(0)}}{(\prod_{i\in \grave{I}}\left(e(g,g)^{a\widehat{\lambda_\iota}tP(u_k)\lambda_k}e(g,h_{\rho(i)})^{-r_itP(u_k)\lambda_k}e(g,\;h_{\rho(i)})^{r_itP(u_k)\lambda_k}e(g,g)^{-aF_itP(u_k)\lambda_k}\right)^{\omega_i}).e(g,g)^{ast\sum_{i=1}^{c}\lambda_iP(u_i)}}$$

$$= \frac{e(g,g)^{\alpha\,s}e(g,g)^{\,a\,stP(0)}}{(\prod_{i\in \grave{I}}\left(e(g,g)^{a\widehat{\lambda}_\iota tP(u_k)\lambda_k}e(g,g)^{-aF_itP(u_k)\lambda_k}\right)^{\omega_i}).e(g,g)^{ast\sum_{i=1}^{c}\lambda_iP(u_i)}}$$

$$= \frac{e(g,g)^{\alpha\,s}e(g,g)^{\,a\,stP(0)}}{(\prod_{i\in \grave{I}}\left(e(g,g)^{ta\left(\widehat{\lambda}_\iota-F_i\right)P(u_k)\lambda_k}\right)^{\omega_i}).e(g,g)^{ast\sum_{i=1}^{c}\lambda_iP(u_i)}}$$

$$= \frac{e(g,g)^{\alpha\,s}e(g,g)^{\,a\,stP(0)}}{e(g,g)^{taP(u_k)\lambda_k\sum_{i\in \grave{I}}\widehat{(\lambda_\iota-F_i)}\omega_i}.e(g,g)^{ast\sum_{i=1}^{c}\lambda_iP(u_i)}}$$

Where $\{\bar{\lambda}_i = \hat{\lambda}_i - F_i\}_{i\in l}$

$$B_1 = \frac{e(g,g)^{\alpha\,s}e(g,g)^{\,a\,stP(0)}}{e(g,g)^{tasP(u_k)\lambda_k}.e(g,g)^{ast\sum_{i=1}^{c}\lambda_iP(u_i)}}$$

$$B_1 = \frac{e(g,g)^{\alpha\,s}e(g,g)^{\,a\,stP(0)}}{e(g,g)^{ast(\sum_{i=1}^{c}\lambda_iP(u_i)+P(u_k)\lambda_k)}}$$

$$B_1 = \frac{e(g,g)^{\alpha\,s}e(g,g)^{\,a\,stP(0)}}{e(g,g)^{astP(0)}}$$

$$B_1 = e(g,g)^{\alpha\,s}$$

Then the decryption algorithm can calculate the message M as:

$$M = \frac{C}{B_1} = \frac{Me(g,g)^{\alpha s}}{e(g,g)^{\alpha s}}$$

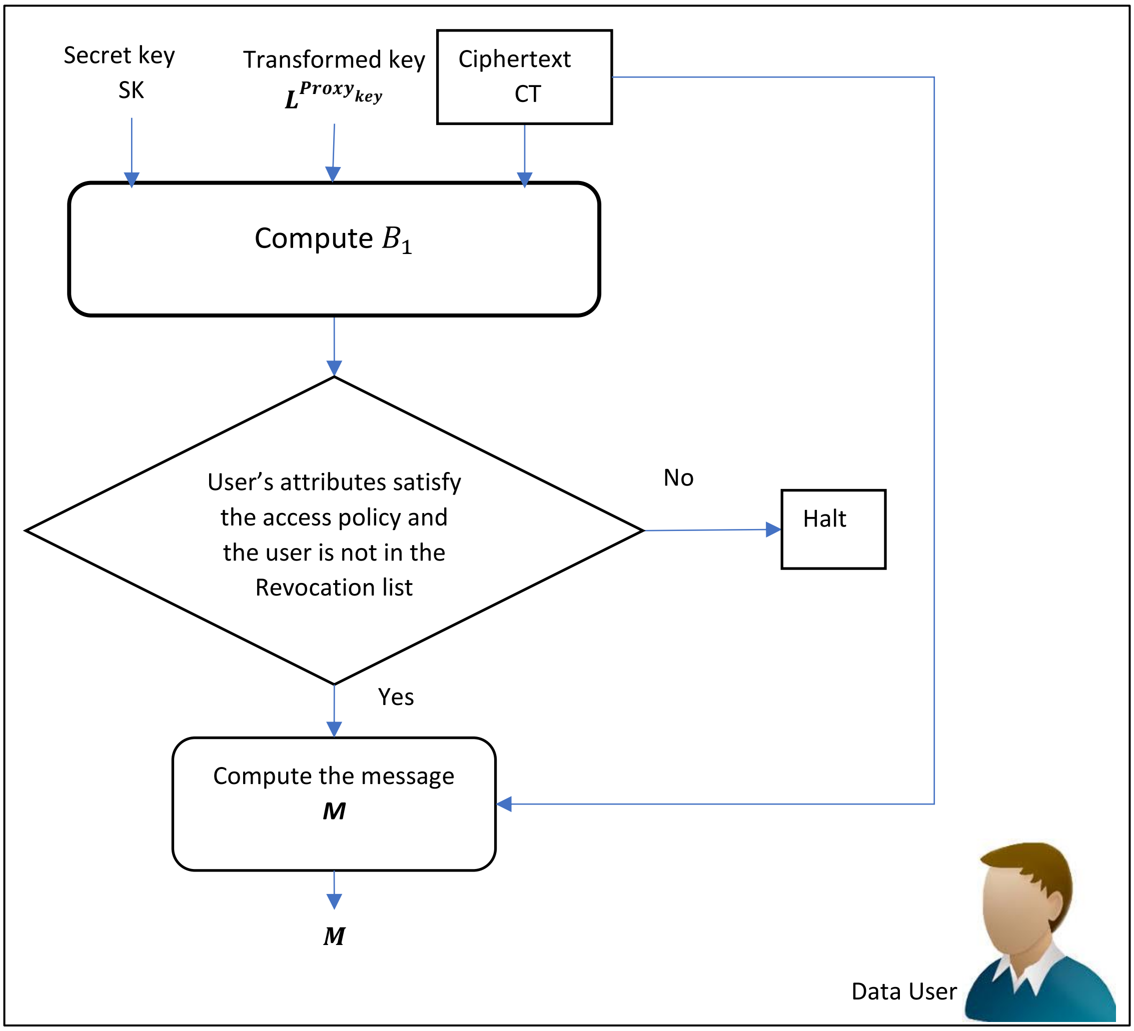


Figure 4.6: The steps of the Decrypt algorithm

$\boldsymbol{PolicyUpdate}(\boldsymbol{PK}, \boldsymbol{KOwner}, \boldsymbol{A}, \boldsymbol{\acute{A}}) \rightarrow (\boldsymbol{ES}, \overrightarrow{\boldsymbol{\grave{Q}}})$: The data owner runs this algorithm (as illustrated in **Figure 4.7**), once there is a need to update the policy. The public parameters PK, the data owner's secret parameter $KOwner$ used in the encryption algorithm, an access policy A (i.e.(W, ρ)) and its updated access policy $\acute{A}$ (i.e.$(\ddot{W}, \ddot{\rho})$) are the inputs of this algorithm. The outputs are a vector of updated, encrypted shares ES according to $\acute{A}$ and its associated formulated vector $\overrightarrow{\grave{Q}}$. Re-sharing the same secret has to be correctly performed using the LSS matrix technique to compute the new shares of the same secret.

As a result, the outcome of this algorithm is the vector $\overrightarrow{\tilde{v}_1}$ that consists of the updated shares. As a consequence of the need to frequently change the shares of the corresponding, updated attributes, the associated ciphertext components that are already stored on the cloud storage server also needs to be updated. Thus, the owner only encrypts the resulted vector $\overrightarrow{\tilde{v}_1}$ in the same way as in the original encryption algorithm to generate *ES*, and then sends it with its associated formulated vector $\overrightarrow{\grave{Q}}$ to the cloud server to re-encrypt the related components. This provides communication and computation offloading without any information leakage to the cloud server.

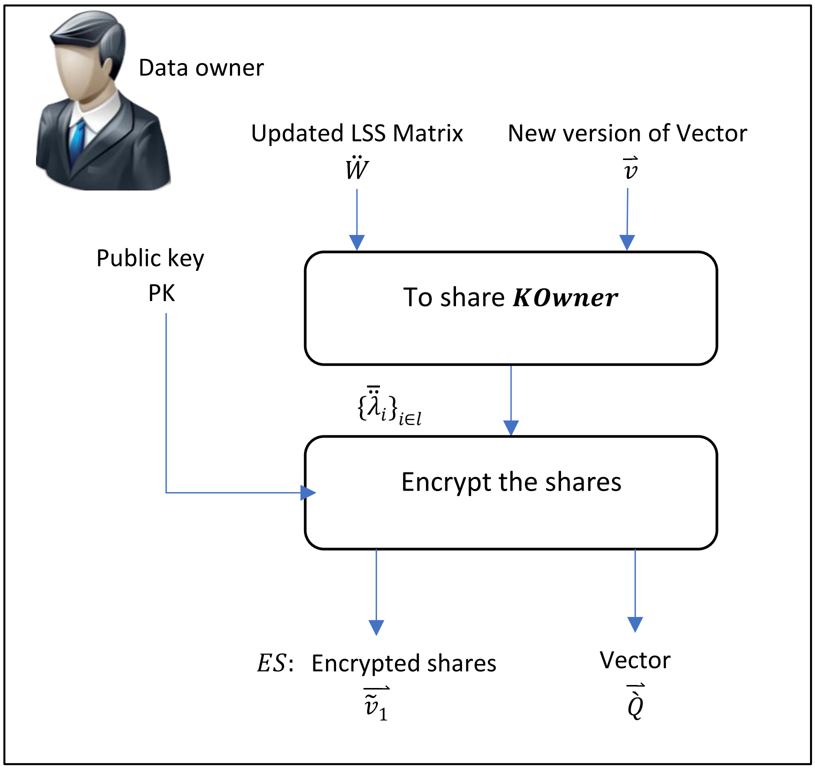


Figure 4.7: The steps of the PolicyUpdate algorithm.

$\boldsymbol{Re-Encryption(PK, ES, \overrightarrow{\grave{Q}}, \acute{A})} \rightarrow \boldsymbol{\acute{C}T}$: The cloud server runs the algorithm that takes the public parameters PK, the encrypted shares ES resulted from the $\boldsymbol{PolicyUpdate}$ algorithm, its associated formulated vector $\overrightarrow{\grave{Q}}$ and the new access policy $\acute{A}$ as inputs (as illustrated in **Figure 4.8**). The cloud server uses these inputs to output the updated ciphertext $\acute{C}T$ using the same criteria in the original encryption algorithm and then publishes $\acute{C}T$ as:

$$\boldsymbol{\acute{C}T} = (C = Me(g,g)^{\alpha s}, C_0 = g^s, \left\{\overline{\overline{C_\iota}} = g^{a\widehat{\lambda_\iota}}\ h_{\rho(i)}^{-r_i}, \overline{\overline{D_\iota}} = g^{\,r_i}, \acute{T}_\iota = g^{-af_i}\right\}_{i=1}^{l})$$

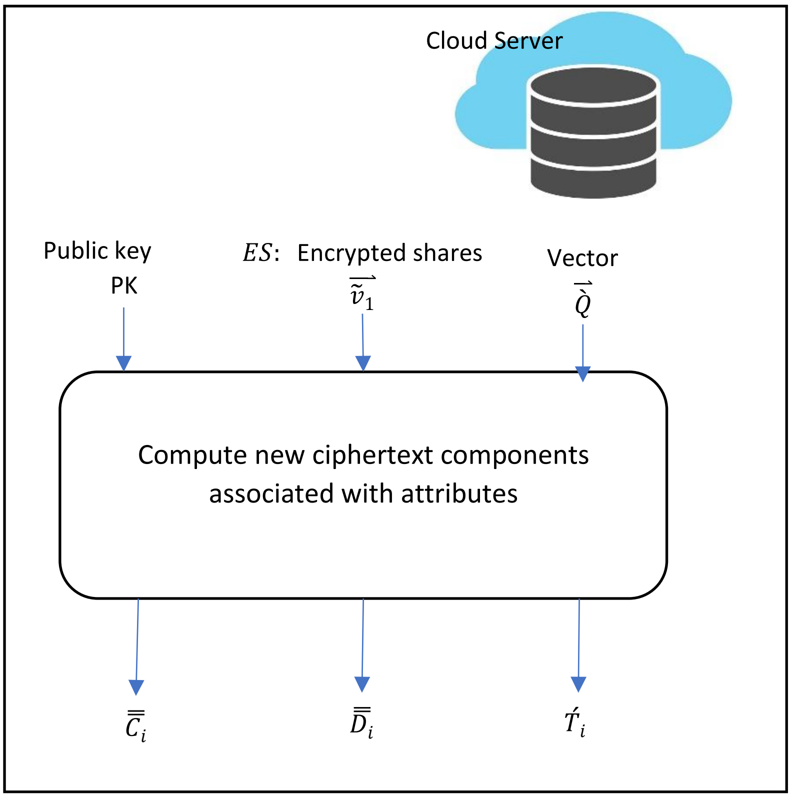


Figure 4.8: The Re-Encryption algorithm step**.**

4.3 Scheme Analysis

The description of our scheme construction in the previous section demonstrates that our scheme manages a user's privileges in accordance with the updating of the data owner's access policy and addresses the user revocation problem. In this section, our construction is analysed in terms of its security and performance.

4.3.1 Security Analysis

This sub-section considers how our proposed scheme meets the security requirements set out in **Section 4.1**, which are data confidentiality, collusion resistance, fine-grained access control and backward and forward security. These requirements are enforced on outsourcing sensitive data to an untrusted environment and handling the attributes with their values changed frequently. The following explanation shows how our scheme fulfils these requirements:

Data Confidentiality. In terms of data confidentiality, our scheme allows the cloud server to re-encrypt ciphertexts without being authorised to decrypt them. Therefore, the data confidentiality is guaranteed against the cloud server. Furthermore, beside the security of Waters' system [42] (see **Section 3.7.1**) that we use as a base of our scheme, we need to prove the security of our policy update, user revocation and proxy operations. Since both the original ciphertext and the updated one have the same distribution, only the security of the original ciphertext with Proxy Revocation is considered.

Definition 4.1. Our proposed single-authority CP-ABE scheme is selectively secure against chosen ciphertext attacks if all polynomial time adversaries have at most a negligible advantage in the same game of **Section 3.6**.

Theorem 4.1. If the security of the basic scheme of Waters' system [42] holds, all polynomial time adversaries have negligible time to selectively break our proposed scheme.

Proof. Suppose that an adversary $\mathcal{A}$ with a non-negligible advantage can break the security of our scheme in a polynomial time. If this adversary wins the selective security game, then a simulator $\mathcal{B}$ can break the security of Waters 'system [42] (i.e. solve the decisional q-Parallel BDHE problem).

Assume two bilinear, cyclic groups of prime order p, G_0 and G_T where $|G| = p$, and g as a generator of G_0. A map $e : G_0 \times G_0 \rightarrow G_T$ denotes a bilinear map. The instance of the decisional q-Parallel BDHE problem $\boldsymbol{y}^{\rightarrow} = (g, g^s, g^a, \ldots, g^{(a^q)}, g^{(a^{q+2})}, \ldots, g^{(a^{2q})}, \ldots)$ is given to the simulator $\mathcal{B}$. When the message is hidden from $\mathcal{A}$, the value of T will be any random element in G_T and otherwise T equals to $e(g,g)^{a^{q+1}s}$ (i.e. $T = e(g,g)^{a^{q+1}s}$).

Init. The adversary $\mathcal{A}$ sends the challenge access policy $({W^*}_{l^* \times n^*}, \rho^*)$ to the simulator $\mathcal{B}$, where $l^*,\ n^* \leq\ q$.

Setup. Some parameters are chosen by the simulator $\mathcal{B}$ to generate the public and master keys. An element $\acute{\alpha}$ is randomly chosen by the simulator $\mathcal{B}$ where $\alpha' \in\ Z_p$. The simulator $\mathcal{B}$ sets $e(g,g)^{\alpha} = e(g,g)^{\acute{\alpha}}. e\left(g^a, g^{a^q}\right) = e(g,g)^{\acute{\alpha}}. e(g,g)^{a^{q+1}}$ that means implicitly $\alpha = \acute{\alpha} + a^{q+1}$.

In terms of computing each group element h_x that corresponds to an attribute x where $1 \leq x \leq U$, a number z_x is selected randomly for each x. Let I^* be a set where $I^* = \{i : \rho^*(i) = x\}$:

$$h_x = g^{z_x} \prod_{i \in I^*} g^{aW^*_{i,1}/b_i}. g^{a^2 W^*_{i,2}/b_i} \ldots g^{a^{n^*} W^*_{i,n^*}/b_i}$$

Notably, when I^* is an empty set, $h_x = g^{z_x}$. Then $\mathcal{B}$ randomly chooses $P(0) \in\ Z_p$.

In addition, the simulator generates the proxy key and the revocation list $R\ ^*$with c random users and their shares (i.e. $R\ ^* = \{\left(u_1^*, P(u_1^*)\right), \ldots, (u_k^*, P(u_k^*))\}$ where $k \in \{1, \ldots, c\}$. Then the simulator sends the public parameters $PK = (g, e(g, g)^{\alpha}\ , g^{a}\ , h_1, \ldots, h_U)$ to the adversary and keeps the master $MSK = (a, g^{\alpha}, P)$ and proxy keys secret.

Query 1. In this phase, the simulator $\mathcal{B}$ responds to all restricted secret keys queries made by the adversary for users $(u_1^*, \ldots, u_n^*)$, where $u_n^* \in Z_p$. Whilst $\mathcal{A}$ sends a new list of the revoked identities $RL^{\,*} = (u_1^*, \ldots, u_k^*)$ where $k \in \{1, \ldots, c\}$ to the simulator, the simulator updates its revocation list $R^{\,*}$ (by computing the shares $P(u_k^*)$ of the corresponding identities in $RL^{\,*}$) and proxy key and sends the generated secret keys to $\mathcal{A}$ unless their attributes satisfy the access policy (W^*, ρ^*) and $u_i^* \notin R^{\,*}$. Therefore, the $\mathcal{B}$'s responses will be as follows:

1- If the attribute set S satisfies the access policy (W^*, ρ^*) and $u_i^* \notin R^{\,*}$, then abort.

2- If the attribute set S does not satisfy the access policy (W^*, ρ^*) and $u_i^* \notin R^{\,*}$, the simulator $\mathcal{B}$ finds a vector $\vec{w} = (w_1, w_2, \ldots, w_{n^*}) \in Z_p$ where $w_1 = -1\ \forall i, \rho^*(i) \in S, \vec{w}.W_i^{\,*} = 0$. Moreover, the simulator $\mathcal{B}$ randomly selects r $\in Z_p$.

   Implicitly, the simulator $\mathcal{B}$ defines t as below:

$$t = r + w_1.a^{q} + w_2.a^{q-1} + \cdots + w_{n^*}.a^{q-n^*+1}$$

   Therefore, to compute L

$$L = g^{at} = g^{ar} \prod_{i=1,..,n^*} (g^{a^{q-i+1}})^{w_i}$$

$$\acute{L}_1 = g^{tP(u_k)} = g^{P(u_k)r} \prod_{i=1,..,n^*} (g^{a^{q-i+1}})^{w_i}$$

   To generate K:

$$K = g^{\acute{\alpha}}.g^{P(0)ar} \prod_{i=2,..,n^*} \left(g^{a^{q-i+2}}\right)^{w_i}$$

   At this step, for each $x \in S$ and x does not use in the access structure, K_x is calculated as:

$$K_x = \acute{L}_1^{\;z_x}$$

   for each $x \in S$ and x is in the access structure, K_x is hard to calculate. Since the term of the form g^{a^{q+1}/b_i} ought not to be computed and hard to be defined.

$$K_x = \acute{L_1}^{z_x} \prod_{i \in I} \prod_{j=1,..,n^*} \left(g^{(a_j/b_i)^r} \prod_{k=1,..,n^*,k \neq j} (g^{a^{q+1+j-k}/b_i})^{w_k} \right)^{W_{i,j}^*}$$

The adversary tries to interact with $\mathcal{B}$ to transform part of its secret keys by calling the Key transform algorithm. Then the simulator runs this algorithm using its proxy key and responses to its request as not belonging to the revocation list.

3- If the attribute set S satisfies or does not satisfy the access policy (W^*, ρ^*) and $u_i^* \in R^*$, then K , L , $\acute{L_1}$, and K_x are computed as the same way in case 2 by $\mathcal{B}$. However, $\mathcal{B}$ does not response to $\mathcal{A}$'s requests to transform its secret keys whose identities belong to the revocation list. In this case, $\mathcal{A}$ cannot recover the value of $P(0)$ even if its secret keys are computed correctly because these keys are invalidated by $\mathcal{B}$.

Challenge. The adversary sends two equal-length messages $(M_0, M_1 \in G_T)$ to $\mathcal{B}$. One of the submitted messages is chosen randomly by the simulator who encrypts it under (W^*, ρ^*). The challenged ciphertext is set as below:

- The simulator selects one of the received messages $M_{\ddot{c}}$ where $\ddot{c} \in \{0,1\}$ and $s \in Z_p$. $\mathcal{B}$ generates the ciphertext components
  $C = M_{\ddot{c}}. e(g^s, g^{\acute{\alpha}}. g^{a^{q+1}}) = M_{\ddot{c}}. e(g,g)^{a^{q+1}s}. e(g^s, g^{\acute{\alpha}}) = M_{\ddot{c}}T. e(g^s, g^{\acute{\alpha}})$
  $\acute{C} = g^s$
- To generate the component C_i, the simulator $\mathcal{B}$ computes a vector $\vec{v}$ where the first element in this vector will be the secret s that needs to be shared. So, $\vec{v} = (s, sa + \acute{y}_2, sa^2 + \acute{y}_3, \dots, sa^{n-1} + \acute{y}_{n^*}) \in Z_p^{n^*}$, where $\acute{y}_2, \dots, \acute{y}_{n^*}$ are randomly chosen. $\mathcal{B}$ encrypts $\vec{v}$ using the formula $\hat{\lambda}_i = W_i^*(sa^{i+1} + \acute{y}_{n^*}) + f_i \; mod \; p$, where $f_i \in Z_p$ is selected randomly and $\hat{\lambda}_i$ represents the encrypted share. Moreover, the simulator randomly chooses $\acute{r}_1, \dots, \acute{r}_l$ and generates a set R_i, where $i = 1, \dots, n^*$. This set contains all indices of rows that are each assigned to a similar attribute as row i (i.e. $\rho^*(i) = \rho^*(k) \; where \; i \neq k$). Therefore, the generated ciphertext components are as follows:

$$D_i = g^{-\acute{r}_i}. g^{-sb_i}$$

$$C_i = h_{\rho^*(i)}^{\acute{r}_\iota} \left(\prod_{j=2,\dots,n^*} (g^a)^{(W_{i,j}^* \acute{y_j}) + f_j} \right) . (g^{s.b_i})^{-z_{\rho^*(i)}} . \left(\prod_{k \in R_i} \prod_{j=1,\dots,n^*} (g^{a^j s.(b_i/b_k)})^{W_{k,j}^*} \right)$$

$$T_i = g^{-af_i}$$

Query 2. It is the same procedure as Query 1.

Guess. At this phase (as in **Section 3.7.1**), the adversary $\mathcal{A}$ outputs its guess $\acute{c}$ of $\ddot{c}$. If $\acute{c} = \ddot{c}$, then the simulator $\mathcal{B}$ outputs 0 which means that $T = e(g,g)^{a^{q+1}s}$. In this case, we have:

$$Pr\left[\mathcal{B}\left(\vec{y}, T = e(g,g)^{a^{q+1}s}\right) = 0\right] = \frac{1}{2} + \text{Adv}_{\mathcal{A}}$$

Otherwise, the simulator outputs 1 which means that T is a random group element in G_T and $M_{\ddot{c}}$ is totally concealed from the adversary. In this case, we have:

$$Pr\left[\mathcal{B}(\vec{y}, \mathrm{T} = \mathcal{R}) = 0\right] = \frac{1}{2}$$

Consequently, the simulator $\mathcal{B}$ would break the security of Waters' system [42], if the adversary wins the game with a non-negligible advantage.

$$Pr\left[\mathcal{B}\left(\vec{y}, T = e(g,g)^{a^{q+1}s}\right) = 0\right] - Pr\left[\mathcal{B}(\vec{y}, \mathrm{T} = \mathcal{R}) = 0\right] = \text{Adv}_{\mathcal{A}}$$

Based on Definition 4.1 and Definition 3.3 **(in Section 3.5.2),** we can observe that the previous hypothesis is contradicted. That means, in the selective security game, there is no adversary $\mathcal{A}$ with a non-negligible advantage. Regarding the theorem, our proposed scheme is secure.

Collusion Resistance. In terms of collusion resistance, most existing systems are based on a common assumption that the cloud server is honest but curious, indicating that the cloud server could collect information correlated to the stored data on it, even if such outsourced data is encrypted. However, the collected information may be used to infer private information by analysing some basic information. Hence, any proposed systems ought to be resistant against any collusion attacks, particularly between the cloud service provider and any revoked users.

To bridge the above-stated gap, our proposed scheme uses the principle of a SSSS to restrict any collusion attacks by embedding an independent random secret share into every user's secret key. This approach ensures that our scheme is collusion resistant and able to securely revoke the main part of the secret key of each revoked user. Therefore, there is no potential of malicious collusion attacks between the revoked users and the cloud server, which is achieved by invalidating the keys of the revoked users. In addition, only limited restricted steps could be taken by the curious untrusted cloud server. Based on these considerations, our scheme is secure.

Consequently, we have compared our scheme with the most relevant existing works [64, 65] and concluded that our scheme is more practical than the existing ones. This is because the authors in [64] assume that the cloud server is semi-trusted and assign essential tasks to it, meaning that their system does not resist against collusion attacks. To avoid such attacks, the system in [65] assigns most of the heavy operations to the data owner, resulting in heavier communication and computation overheads for the data owner. **Table 4.2** summarises the above comparison.

Table 4.2: Comparison of scheme abilities to resist collusion attacks.

Scheme	**Most of Operations Run by**	**Collusion Resistance**
PU-CP_ABE [65]	Data owner	Yes
Guangbo Wang and Jianhua Wang[64]	Cloud server	No
Our scheme	Cloud server	Yes

Fine Grained Access Control. This requirement is enabled by using CP-ABE that cryptographically enforces expressive access policies by data owners. This type of access control grants data owners the ability to choose with fine granularity who can access their data.

Forward and Backward Security. In terms of backward security, our proposed scheme does not need to achieve it, instead it keeps the distribution of the updated ciphertexts in the same form as the distribution of the original ciphertext. In the context of forward security, this security requirement is achieved by activating two processes. First, our technique dynamically updates a data owner's policy, leading to eleviating or revoking some users' privileges. This prevents the users with their attributes revoked, from accessing the same level of the subsequent data. Secondly, a user revocation process revokes all privileges of any revoked user and prevents them from decrypting any newly published data.

4.3.2 Performance Analysis

In this subsection, we compare our scheme with the most relevant schemes in [64, 65], in terms of communication, computation and storage costs. The comparison illustrates that our scheme is more practical as shown in the following discussions.

Communication cost. The communication cost between the key attribute authority (AA) and the system users (SU) is generated by transmitting the secret keys and handling updating events, which is similar in all the compared systems. The communication cost for the policy update and the ciphertext re-encryption operations between the data owner (DO) and the cloud server (CS) is mainly due to transmitting the ciphertext and any changed information after each access privilege setting. In addition, there are extra communicating operations needed in the two other systems, where in [64], the system users need to send their transformation keys to the cloud server to perform partial decryption for them, whereas in the re-randomization operation in [65], the data owner downloads ciphertext, re-randomizes it, and then transmits it back to the cloud server. That means that the communication cost of just the re-randomization operation in [65] equals to the cost of sending the original ciphertext. As illustrated in **Table 4.3**, in [64], upon each revocation event, the revocation list is sent to the cloud server (where the size of this list increases linearly with the number of revoked users), affecting the communication bandwidth. In our scheme, two vectors have to be sent through the channel between the cloud server and the data owner where the number of elements in each vector is equal to the number of the attributes in the access policy (where this number is obviously less than the number of revoked users in [64]).

Computation and storage cost. The two relevant systems [64, 65] introduce almost the same computational cost in all their algorithms except in the ciphertext re-encryption phase. In terms of policy update and ciphertext re-encryption operations, as shown in **Table 4.4**, the computation burden of generating the updated access policy and updating the ciphertext and additional re-randomization operation (that equals to the cost of the original encryption operation) in [65] is put on the data owner. Although the work in [64] delegates part of the decryption operation and revocation process to the cloud server by granting more control privileges to the cloud server (i.e. at the expense of not being a collusion-resistance system), its ciphertext size increases linearly with the revocation list size, leading to the increase in the storage cost. In our scheme, the process for updating the access policy is the data owner's responsibility, which is clearly less than the data owner's responsibilities in [65], while the re-encryption operation corresponding to the updated policy is run by the cloud service provider.

Table 4.3: Comparative summary of communication costs of our scheme against related work.

Scheme	**Communications between the data owner and the cloud upon each revocation event**
PU-CP_ABE [65]	The updated ciphertext components and the whole re-randomized ciphertext
Guangbo Wang and Jianhua Wang [64]	Revocation list
Our scheme	Two vectors, one for the encrypted shares and another for the corresponding formula, where the number of elements in each vector is equal to the number of attributes in the access policy

Table 4.4: Comparative summary of computation and storage costs of our scheme against related work.

Scheme	**Owner**	**Cloud**
PU-CP_ABE [65]	Update the access policy, compute the updated ciphertext components and re-randomize the whole ciphertext	
Guangbo Wang and Jianhua Wang[64]	Generate the revocation list	Generate new shares and headers, re-encrypt the ciphertext with an increased size, and partially decrypt the encrypted data
Our scheme	Update the access policy	Re-encrypt the ciphertext and partially encrypt ciphertexts

The spread of the updated ciphertext. The spread of the updated ciphertext in our scheme and the PU-CP_ABE scheme in [65] is the same as the spread of the original ciphertext which is the core challenge. However, the spread of the two corresponding ciphertexts in [64] is different.

4.4 The Experiment Results and Evaluation

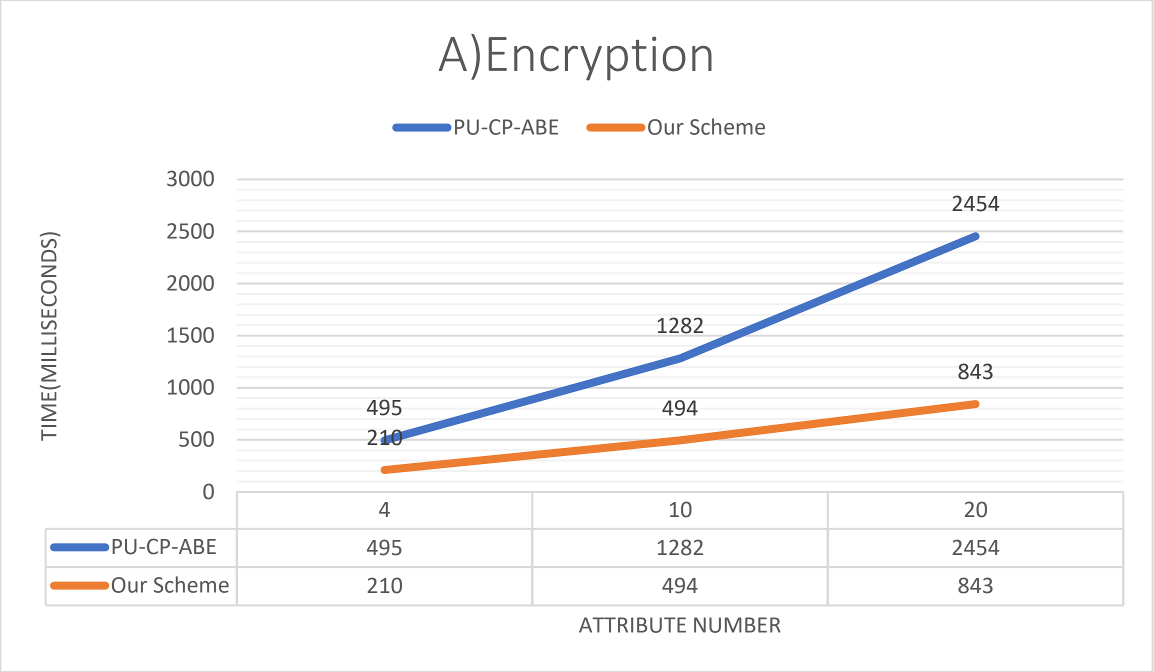
We implemented the proposed scheme specified in **Section 4.2.4**. The CP-ABE system in [42] was taken as the base and developed to adapt to our proposed scheme. The experiment supports dynamic policy update processes, user revocation and relieves the encryption burden on the data owner.

The implementation uses the Java Pairing Based Cryptography (JPBC) library [115] which is built using Java. A 160-bit Elliptic curve group of type (A) curves ($y^2 = x^3 + x$) of JPBC is used with a 512-bit base field. The experiment is executed on Intel(R) Core(TM) i7-2006, 3.40 GH CPU, where pairing takes about 31 milliseconds.

In the CP-ABE scheme, the time consumed in the Encryption process is mainly linear with the number of attributes involved in an access policy. Therefore, the computation overhead results from using a complex access policy. To examine the efficiency of our proposed scheme, we have considered access policies from simple to complex forms with various numbers of attributes (i.e. 4, 10 and 20 attributes in our experiment).

In terms of encryption, comparisons between our scheme and the most relevant scheme PU-CP-ABE in [65] were carried out via simulations. Due to outsourcing some expensive operations (i.e. computing the ciphertext components associated with attributes) to the cloud server, the simulation results match our expectations and show that our scheme is more scalable and efficient than the compared one. Also, the efficiency gain of our scheme increases when the number of attributes in the access policy increases, as evidenced in **Figure 4.9 A**. For example, the compared PU-CP-ABE system takes 136% more time than our solution in the case of 4 attributes and 191% more time when the number of attributes increases to 20.

Similarly, the time consumption of the policy update process is compared between our scheme and PU-CP-ABE in [65]. After the first implementation of the experiment, the time of the policy update process is computed by changing the number of attributes in the access policy from 4 to 10 attributes and then from 10 to 20 attributes, respectively. Although the two systems outsource partial operations to the cloud server, the performance evaluation results presented in **Figure 4.9 B** illustrate that our scheme consumes less time, where the scheme of PU-CP-ABE takes 96% and 80% more time than our policy update process in the cases of 10 and 20 attributes, respectively. The main reason for this is that the compared system delegates just affordable operations to the cloud server, while a data owner carries out the expensive ones.



	4	10	20
PU-CP-ABE	495	1282	2454
Our Scheme	210	494	843

Figure 4.9 A: Encryption performance comparison between our scheme and PU-CP-ABE.

In terms of a decryption operation, the time consumed in a CP-ABE scheme mainly depends on the complexity of the access structure and the attributes set involved (i.e. the maximum number of attributes in a user's decryption key, which satisfy the access policy embedded in a ciphertext). Our scheme offers performance slightly lower than the compared system, as shown in **Figure 4.9** C. For instance, our scheme consumes 44% more time than the PU-CP-ABE scheme when the access policy consists of 4 attributes, and takes 33% and 34% when the numbers of attributes in the access policy are 10 and 20, respectively. The variations in percentages are affected by changing the number of the involved attributes. Note that even if 20 attributes are used, not all the attributes are involved in an access policy, which reflects a small time increase from 10 to 20 attributes. The reason for our scheme performance degradation is that unlike the compared system, our scheme needs additional cryptographic operations to solve the user revocation problem that is not considered in the compared system, leading to more computation. Alternatively, the compared system assigns heavy operations to a data owner, who has to stay online, to avoid colluding between the revoked users and the cloud server. In this case, the compared system does not need to address this property.

Finally, the re-encryption operation in our scheme and the compared one is carried out by the cloud server. For this reason, there is no need to illustrate any comparisons between them, because evaluating the efficiency of our proposed scheme is mainly on measuring the burden on data owners and users due to their limited computing resources. Although the overall performance evaluation of our scheme is higher than the compared one, we intend to improve the time consumed by a decryption operation in our future work.

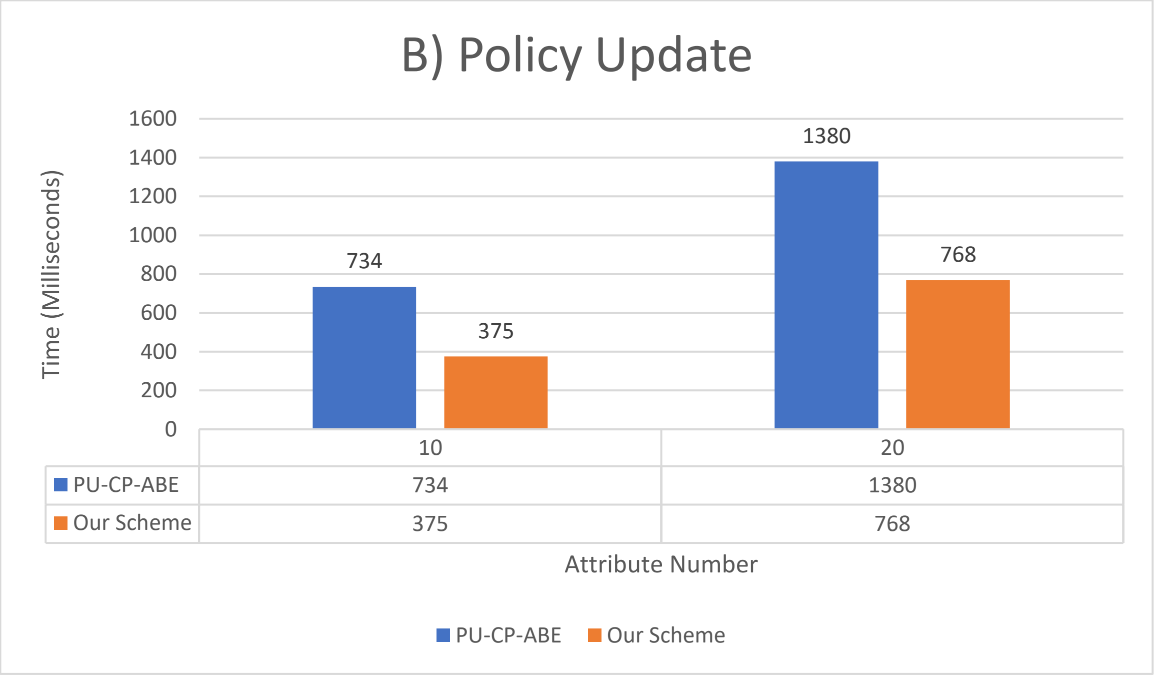


Figure 4.9 B: Policy update performance comparison between our scheme and PU-CP-ABE

Despite the expensive pairing operations of CP-ABE, many of the existing systems [116, 117] prove the feasibility of CP-ABE that suits many IoT applications. Due to the efficiency of our scheme that already enhances the efficiency of CP-ABE, our proposed scheme is practical. We tried to implement our scheme on a mobile phone. However, the JPBC Library that our proposed scheme is built on, is inefficient on mobile phones. It can only be run on Android 2.2 or lower. Moreover, only a few elliptic curve cryptography libraries support pairing operations written in Java. Most of these libraries are not compatible for mobile phone operating systems (such as Android) and only available for desktop-based applications [118, 119]. We intend to perform the above experiment when the facilities required become available.

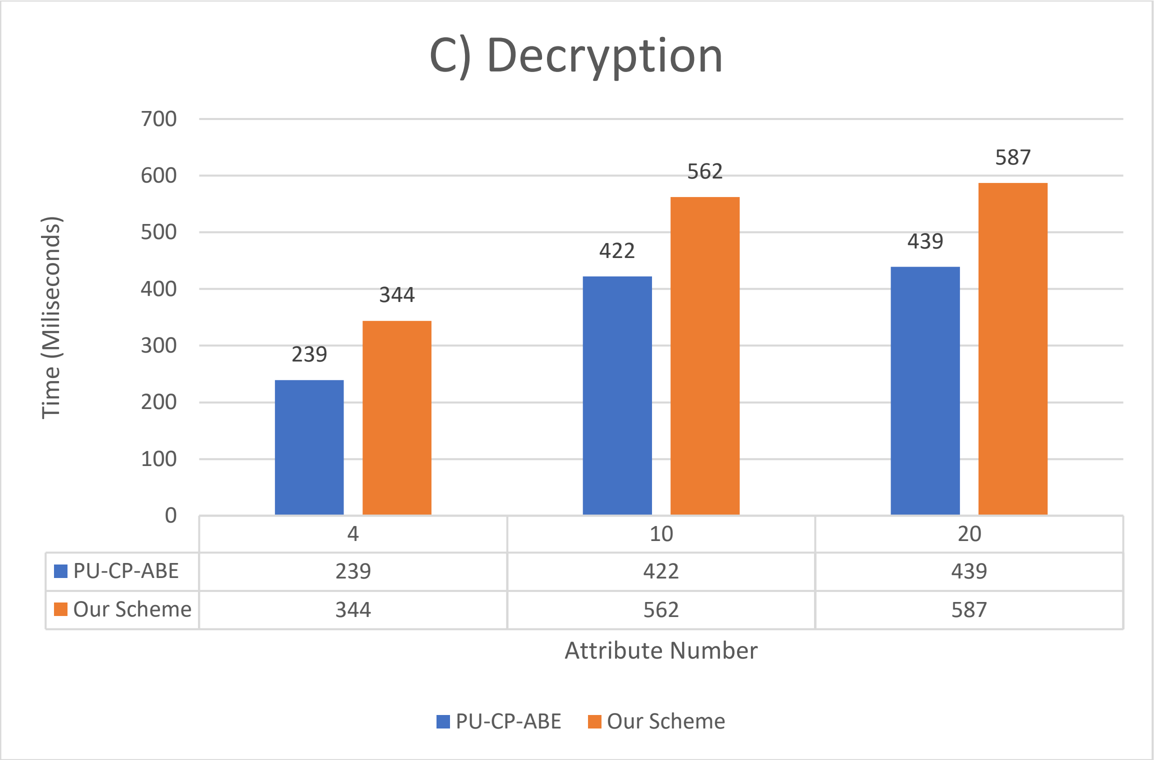


Figure 4.9 C: Decryption performance comparison between our scheme and PU-CP-ABE

4.5 Conclusions

In this chapter, some essential issues have been considered, including outsourcing computation to the cloud in a secure manner, constructing a scheme that resists against collusion attacks, and addressing the revocation problem. The consideration of these issues is due to the lack of appropriate solutions in the existing work. We have thus exploited the merits of some existing systems to introduce a novel technique that efficiently processes attribute revocation by updating access policies dynamically. In particular, the tasks for re-encrypting ciphertext and invalidating the secret keys for the revoked users are distributed between a cloud server and a proxy server, respectively, to support attribute and user revocations. In our proposed scheme, a light load is placed on the attribute authority to grant minimal control privileges to the cloud server without revealing any useful information when access privilege customization takes place. The scheme analysis demonstrates that our proposed scheme is secure and more practical. Further work on extending our scheme to a multi-authority access control scheme and outsourcing part of the decryption operation to the cloud server will be covered in the next chapter.

Chapter 5

The Proposed Multi-authority, Revocable Access Control Scheme

Chapter 5: The Proposed Multi-authority, Revocable Access Control Scheme

5 Introduction

For secure, public cloud storage, an access control scheme is critical, which ought to be carefully designed to achieve fine-grained access control and support outsourced-data confidentiality. CP-ABE is introduced as one of the most beneficial, powerful techniques that can be leveraged to construct a secure access control system. However, this type of technique mainly supports storing data only on a private cloud storage system in which the service is managed by only one single authority. In addition, CP-ABE does not properly consider revocation issues to address changes to policy attributes and users. These two issues have motivated many researchers to develop more suitable schemes with limited success.

By leveraging the existing work, in this chapter, we propose a new CP-ABE scheme that extends the scheme in Chapter 4, tackles most of the existing work's limitations and allows storing data on a public cloud storage system securely by employing multiple authorities that manage a joint set of attributes. Furthermore, the proposed scheme efficiently addresses the revocation issue by presenting two techniques that allow policy update and invalidate a user's secret key to eliminate collusion attacks. In terms of computation overhead, the proposed scheme outsources expensive operations of encryption and decryption to a cloud server to mitigate the burden on a data owner and data users, respectively. Our security and performance analysis of the scheme demonstrates that our scheme is practical and secure.

The rest of this chapter is organised as follows. The requirements and security assumptions that our scheme relies on are presented in Section 5.1. Our proposed multi-authority scheme, its entities and algorithms are introduced in Section 5.2. The scheme security and performance are analysed and discussed in Section 5.3. Some experimental results are demonstrated in Section 5.4. Finally, our work is concluded in Section 5.5.

5.1 Scheme Security

The main security assumptions, security requirements and the security model in our proposed multi-authority access control scheme are presented in this section.

5.1.1 Security Requirements and Assumptions

There are several security assumptions and requirements for our proposed multi-authority scheme. In addition to those for our single-authority scheme in **Section 4.1**, it also follows the assumption and requirement stated below:

1- A central authority CAA is assumed to be a semi-trusted entity, but it can be attacked by an adversary. Due to the decentralised setting, it ought not to be involved in generating secret keys for users.

2- Minimising the trust level of attribute authorities and tackling the key escrow problem are considered by hiding part of the master key from all the scheme entities including the attribute authorities that use the implicit value of the master key to generate users' secret keys. Also, each attribute authority generate share of a user's secret key not the whole secret key.

5.1.2 Security Model

In this part, the security model in a multi-authority access control system is considered in two phases. In the first phase, the adversary $\mathcal{A}$ attempts to compromise the attribute authorities to obtain the master secret key. The second phase is similar to a system in [42], where an attempt has been made to decrypt a ciphertext encrypted over an access policy, by the adversary with secret keys that do not have the attributes satisfying the access policy (the details are similar to those stated in **section 3.6**).

Definition 5.1. Our proposed multi-authority scheme is secure against chosen ciphertext attacks if all polynomial time adversaries have at most a negligible advantage in the selective security game.

5.2 Our Proposed Scheme

In this section, we present the scheme model, which involves six entities that run eight scheme algorithms. Some details of these entities (as shown in **Figure 5.1**) and algorithms are described below.

5.2.1 Scheme Entities

Our multi-authority scheme consists of one central authority, multiple attribute authorities, data owners, data users, a cloud server, and a proxy server:

Central authority (CAA). This entity establishes the scheme public parameters, part of the master key and a public key for each attribute in the scheme as well as assigning the identities to authorized data users and attribute authorities. Moreover, CAA identifies the threshold number of required attribute authorities involved in generating users' secret keys. In addition, once a revocation event occurs, the central authority issues a proxy key to the proxy server according to the list of revoked users, which will be introduced later.

Attribute authority (AA). This entity is one of multiple authorities involved in key generation as well as jointly managing the scheme attribute set. All attribute authorities (AAs) in the scheme are responsible for dealing with joint attribute sets and sharing a part of the scheme master secret key, where only a threshold number of these authorities are able to generate users' secret keys. Here, each AA communicates with all the other authorities when they need to share the scheme master key, but no communication is required among the authorities to generate a user's secret key.

Data users. Each user receives its identity from the central authority and its secret key, which contains its attributes, from a threshold number of attribute authorities. This secret key contains the attribute set that the user possesses. Once this set satisfies the embedded access policy, the user can gain access and recover the original data. Otherwise, the ciphertext cannot be decrypted.

Data owners. This party has the same role as the data owner in our proposed single-authority scheme in **sub-section 4.2.1**.

Cloud server. This server plays the same role as the one in **sub-section 4.2.1** for the multi-authority model to leverage its computing and storage resources. We assign heavy missions to the cloud to partially encrypt data that is uploaded by a data owner, partially decrypt a ciphertext and securely update the ciphertext according to the new access policy after a policy update process. These tasks are assigned with a guarantee that no information is revealed to the cloud server.

Proxy server. The proxy server is a semi-trusted server which is provided with a proxy key by the central authority CAA when each revocation process takes place. This server deals with this key as in **sub-section 4.2.1.**

The structure of our scheme is summarised in **Figure 5.1**. Firstly, in the scheme initialisation phase, each authorised user and authority interacts with CAA to obtain their identities. Then

CAA and AAs work together in a specific way to initialise the scheme parameters. To compute a user's secret key, the user communicates with any m out of n authorities that run the key generation algorithm, to gain its secret key SK. Then the data owners, who intend to outsource their data to a cloud server, execute the encryption algorithm to encrypt their data CT, and run the policy update algorithm when their access policy is changed. Any authorised user can access the encrypted data and recover the plaintext unless its identity is in the revocation list or its attributes do not satisfy the access policy embedded in CT. When a user revocation event occurs, CAA generates a proxy key and sends it to the proxy server which in turn updates each authorized user's secret key unless such user's identity is in the revocation list.

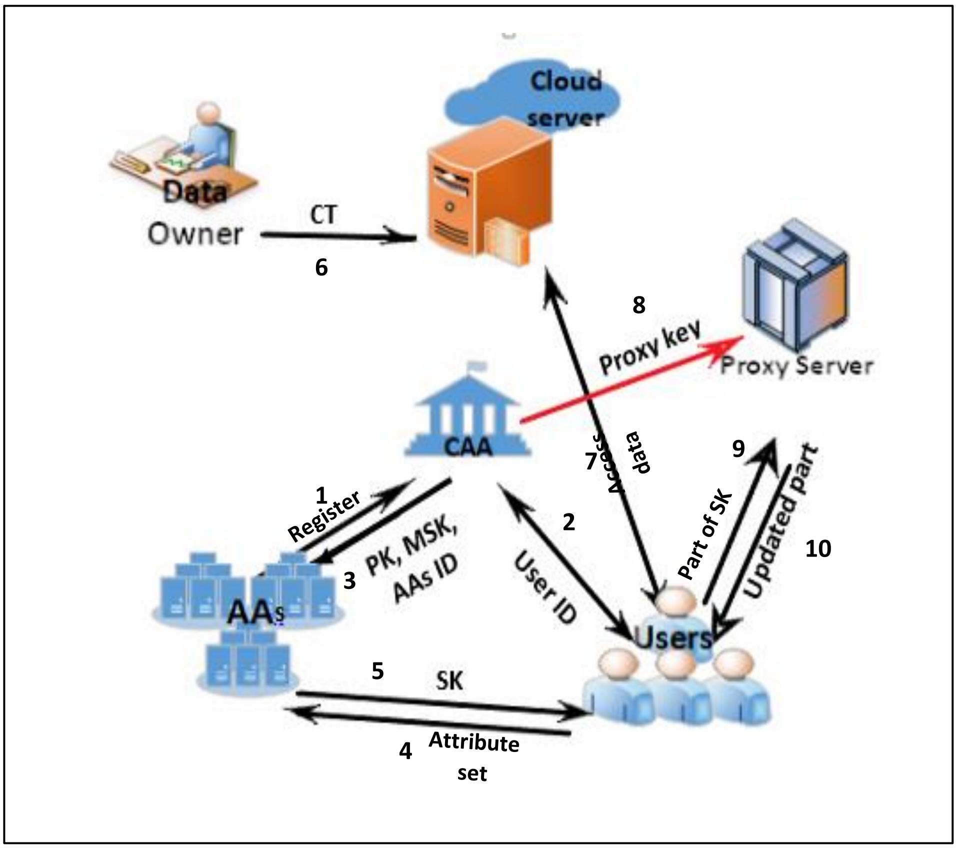


Figure 5.1: Interactions between the multi-authority scheme entities

5.2.2 Scheme Algorithms

The scheme consists of eight algorithms illustrated in **Figure 5.2**. These algorithms are described as below. All main scheme notations used in this chapter are indicated in **Table 5.1** and **Table 4.1**.

Setup: This algorithm contains three sub-algorithms. These algorithms are Setup1CAA, Setup2AA, and Setup3CAA. Where, the scheme contains n attribute authorities AA_i ($i = 1,2, \ldots., n$). The establishment of the scheme parameters and the registration of users and the attribute authorities AAs are carried out in the Setup1CAA algorithm by central authority CAA. In the second algorithm Setup2AA, generating the master secret key is run securely by AAs that contact each other to share the master key MSK. The third algorithm Setup3CAA is responsible for reconstructing the scheme public key PK that corresponds to the master secret key MSK. This algorithm is executed by central authority CAA. The details of these sub-algorithms are presented below:

Table 5.1: Main notations used in Chapter 5

Symbol	Description
$\boldsymbol{P}$	A secret polynomial of degree c which is selected by the central authority.
$\boldsymbol{n}$	The number of attribute authorities.
$\boldsymbol{AA_i}$	An attribute authority i (where, $i = 1,2, \ldots., n$).
$\boldsymbol{a}$	Randomly selected exponent belonging to the finite field Z_p.
$\boldsymbol{u_k}, \boldsymbol{a_{id}}$	Random elements in Z_p that are chosen by the central authority as unique identities of a user and an authority, respectively.
$\boldsymbol{m}$	The threshold that represents the required number of the attribute authorities to generate each user's secret key.
$\boldsymbol{\alpha_i}$	A random number chosen by the attribute authority $AA_{\boldsymbol{i}}$ to be its secret key where $\alpha_i \in Z_p$ for $i = 1,2, \ldots., n$.
$\boldsymbol{\alpha}$	The summation of α_i for all attribute authorities will be $p2MSK = \sum_{i=1}^{n} \alpha_i = \alpha$, where α will be implicitly part of the master key of the scheme.
$\boldsymbol{MSK_i}$	The master secret share of attribute authority $AA_{\boldsymbol{i}}$ ($i = 1,2, \ldots., n$).
$\boldsymbol{P(u_i)}$	The random share which is extracted from polynomial $\boldsymbol{P}$ selected by the central authority for each user $\boldsymbol{u_i}$. This share is used later to de-activate the key when user $\boldsymbol{u_i}$ is revoked.
$\boldsymbol{\bar{\lambda}_i}$	A plain secret share.
$\boldsymbol{\hat{\lambda}_i}$	The encrypted version of $\bar{\lambda}_i$.

$\boldsymbol{Setup1CAA(U)} \rightarrow (\boldsymbol{pMSK}, \boldsymbol{pPK})$: CAA selects two bilinear, cyclic groups of prime order p, G_0 and G_T where $|G| = p$, and g as a generator of G_0. A map $e : G_0 \times G_0 \rightarrow G_T$ denotes a bilinear map. Then CAA randomly chooses a polynomial P of a degree c over Z_p to later use it for blinding each user's secret key by AAs, which will facilitate revocation of a set of users. P is securely sent to each AA in the scheme. Moreover, CAA randomly selects $a \in Z_p$ as a part of the master key, and publishes the corresponding public key part g^a. The input to this algorithm is U which represents the number of attributes in the scheme. CAA generates and publishes $h_1, \ldots, h_U \in G$ which are random group elements to be associated with the U attributes in the scheme as attribute public keys. In the scheme, each user and attribute authority are identified by identities u_k and a_{id} respectively, which are chosen as random elements in Z_p and assigned to the user and authority by CAA. In addition, CAA publishes the threshold m that represents the required number of AAs to generate each user's secret key. **Figure 5.3** illustrates the procedure of this sub-algorithm.

The outputs of this sub-algorithm include (a) a part of the public key, $\boldsymbol{pPK} = (g,\ g^a, \mathrm{n}, \mathrm{m}\ , h_1, \ldots, h_U)$, which will be published for access by all the scheme parties, and (b) a part of the master secret key, $\boldsymbol{pMSK} = (a, P)$, that will be kept secret by CAA.

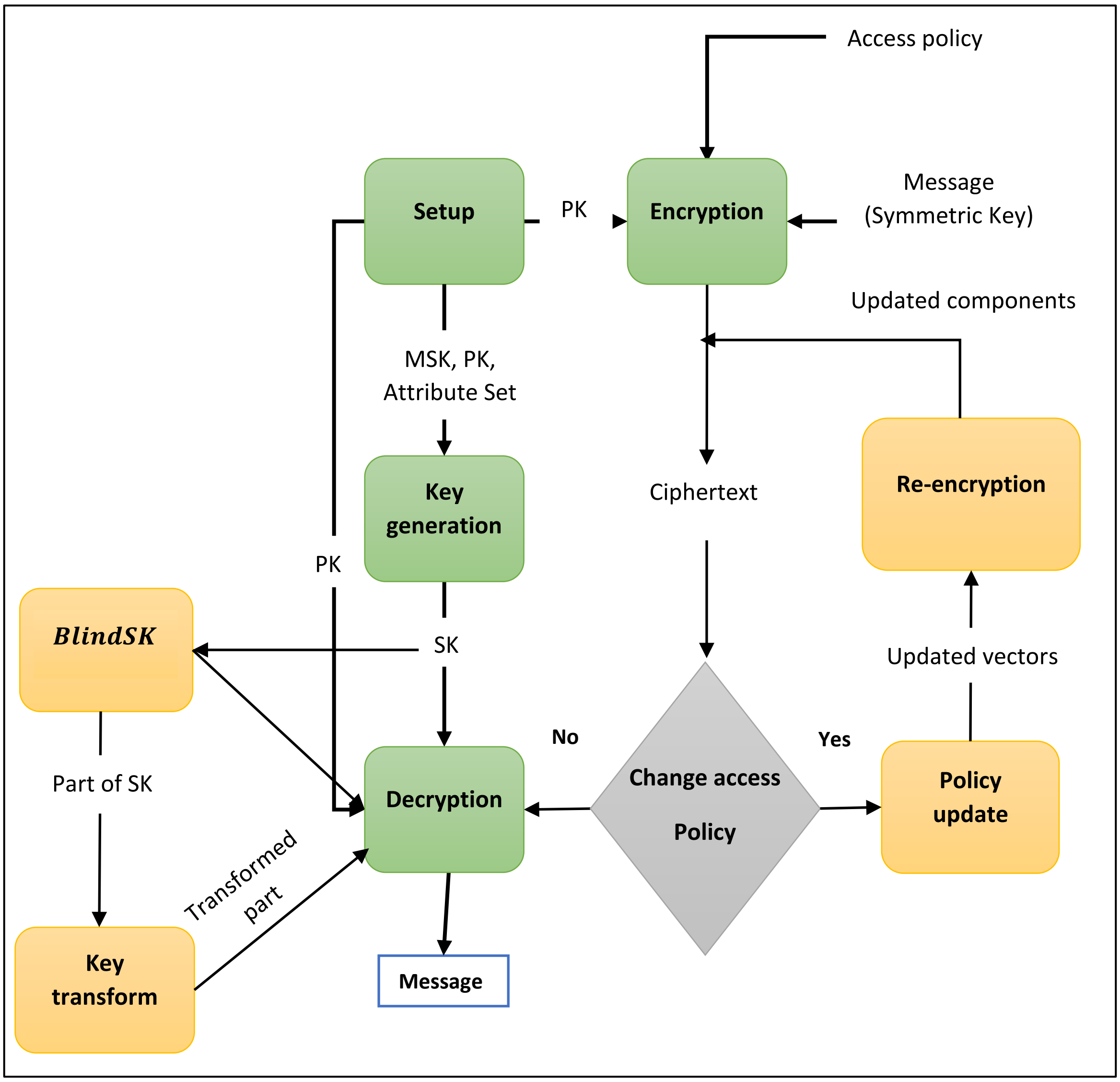


Figure 5.2: The proposed multi-authority system algorithms

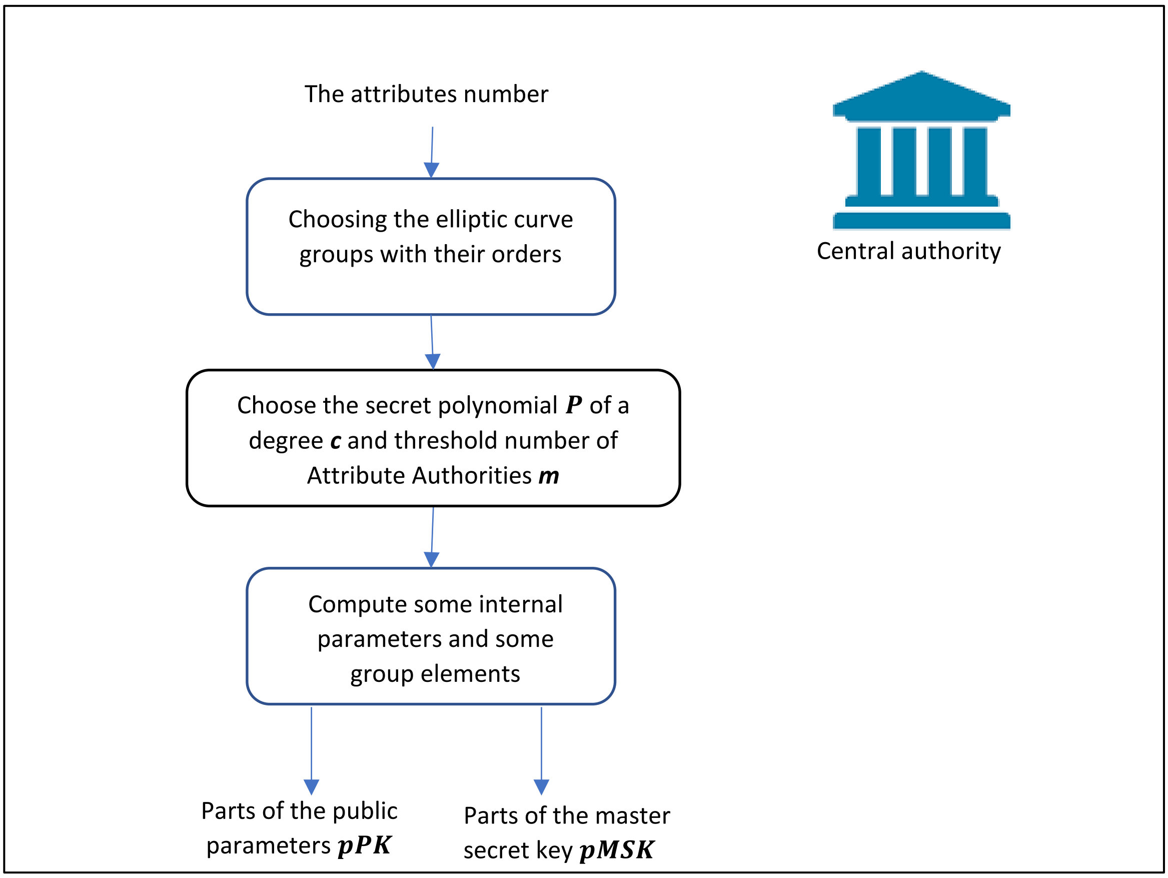

The attributes number
Choosing the elliptic curve groups with their orders
Choose the secret polynomial ***P*** of a degree ***c*** and threshold number of Attribute Authorities ***m***
Compute some internal parameters and some group elements
Parts of the public parameters ***pPK***
Parts of the master secret key ***pMSK***
Central authority

Figure 5.3: The steps of Setup1CAA sub-algorithm.

$\boldsymbol{Setup2AA(n,m)} \rightarrow \boldsymbol{(PK_i,\ MSK_i)}$: In this algorithm, the secret sharing scheme (SSSS) is used by all the AAs involved to share the second part of the master secret key, denoted as $\boldsymbol{p2MSK}$**.** To generate $p2MSK$, each AA_i $(i = 1,2, \dots, n)$ randomly chooses a number α_i to be its secret key, where $\alpha_i \in Z_p$ and the summation of numbers α_i chosen by all the AAs is defined as $p2MSK = \sum_{i=1}^{n} \alpha_i = \alpha$. After that, each AA_i creates a random polynomial $f_i(x)$ of degree $(m-1)$ where $f_i(0) = \alpha_i$. Then, each AA_i computes a share msk_{ii} for itself and msk_{ij} for every other AA_j $(j = 1,2, \dots, n$ but $j \neq i)$. These shares $msk_{ij} = f_i(a_{id_j})$ are securely sent to each corresponding AA_j, while AA_i will keep $msk_{ii} = f_i(a_{id_i})$ secret for itself. Once each AA_i receives $(n-1)$ shares from all other AAs, it calculates its master secret share $MSK_i = \sum_{j=1}^{n} msk_{ji}$ and the corresponding public key share $p2PK = e(g,g)^{MSK_i = PK_i}$ as the outputs of this sub-algorithm, where AA_i publishes PK_i as its public key. **Figure 5.4** shows the steps of this sub-algorithm.

$\boldsymbol{Setup3CAA(PK_i)} \rightarrow \boldsymbol{PK}$**:** This algorithm is executed by CAA that selects m out of n shares of the AAs' public key to generate the scheme public key as follows:

$$e(g,g)^{\alpha} \;=\; e(g,g)^{\sum_{i=1}^{m}(MSK_i \prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}})}$$

$$= \prod_{i=1}^{m} e(g,g)^{MSK_i \prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}}}$$

$$= \prod_{i=1}^{m} PK_i^{\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}}}$$

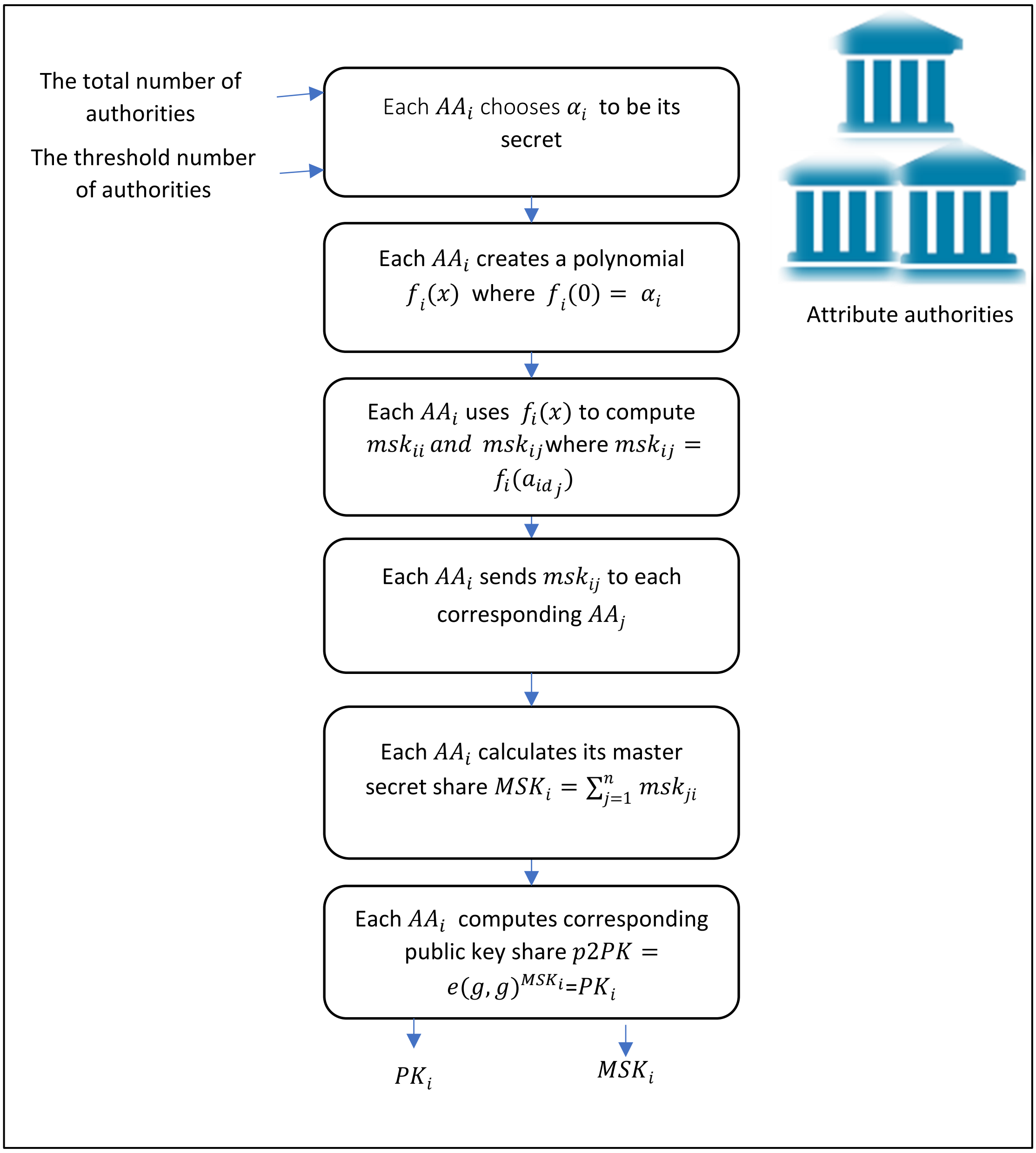

The total number of authorities
The threshold number of authorities
Each AA_i chooses α_i to be its secret
Each AA_i creates a polynomial $f_i(x)$ where $f_i(0) = \alpha_i$
Each AA_i uses $f_i(x)$ to compute msk_{ii} and msk_{ij} where $msk_{ij} = f_i(a_{id_j})$
Each AA_i sends msk_{ij} to each corresponding AA_j
Each AA_i calculates its master secret share $MSK_i = \sum_{j=1}^{n} msk_{ji}$
Each AA_i computes corresponding public key share $p2PK = e(g,g)^{MSK_i}$=PK_i
PK_i
MSK_i
Attribute authorities

Figure 5.4: The procedure of the Setup2AA sub-algorithm.

Eventually, the following scheme parameters are published by CAA as the output of this sub-algorithm:

$$PK = g, e(g,g)^{\alpha},\ g^{a}, n, m, h_1, \ldots, h_U.$$

Here, part of the master key, α, is not gained by any of the system entities, i.e. it exists implicitly. Now the scheme master key is defined as $MSK = (a, \alpha, P)$.

$\boldsymbol{Encrypt}(\boldsymbol{PK}, (\boldsymbol{W}, \boldsymbol{\rho}), \boldsymbol{M}) \rightarrow \boldsymbol{CT}$: A data owner, before migrating its data M ($M \in G_T$) to the cloud server, encrypts M using public key PK and LSSS access structure (W, ρ) specified in **Section 4.2.4**. To execute the encryption, the owner selects a random vector $\overrightarrow{v} = (s, y_2, \ldots, y_n)$ where $y_2, \ldots, y_n \in Z_p$ are used to share secret s. The algorithm computes each value $\bar{\lambda}_i$ as $\bar{\lambda}_i = \vec{v} \cdot W_i$, where W_i is the i^{th} row vector of W $(1 \leq i \leq l)$. These shares $\{\bar{\lambda}_i\}_{i \in l}$ will be encrypted by the owner before outsourcing them to the cloud. Thus, the ciphertext is outsourced to the cloud server as: $CT_1 = ((W, \rho), C = Me(g,g)^{\alpha s}, C_0 = g^{s})$, together with two other vectors. The first one is $\overrightarrow{\tilde{v}} = (\hat{\lambda}_1, \ldots, \hat{\lambda}_l)$, which represents the vector of the encrypted secret shares that will be embedded by the cloud server in the ciphertext components associated with the attributes. Here, an encrypted share $\hat{\lambda}_i$ is yielded from $\hat{\lambda}_i = (\bar{\lambda}_i + F_i) \bmod p$ for $i = 1$ to l. Here, $\{\mathrm{F}_i \in Z_p\}_{i=1}^{l}$ are randomly selected by the data owner for encrypting the plain-shares $\{\bar{\lambda}_i\}_{i=1}^{l}$ in the vector $\overrightarrow{v.W}$. These components are then published by the cloud as $\{\grave{C}_\iota = g^{a\hat{\lambda}_i} h_{\rho(i)}^{-r_i}, \grave{D}_\iota = g^{r_i}\}_{i=1}^{l}$ where the blind numbers $r_1, \ldots, r_l \in Z_p$ are chosen randomly by the cloud server.

The second vector is $\overrightarrow{Q} = (g^{-aF_1}, \ldots, g^{-aF_l})$, in which its elements correspond to the encrypted shares in $\overrightarrow{\tilde{v}}$. Both $\overrightarrow{\tilde{v}}$ and $\overrightarrow{Q}$ are used to recover the plain-shares $\{\bar{\lambda}_i\}_{i=1}^{l}$ by authorized users. For instance, the first element g^{-aF_1} in $\overrightarrow{Q}$ is used with the first encrypted share $\hat{\lambda}_1$ in $\overrightarrow{\tilde{v}}$ to recover the first plain share $\bar{\lambda}_1$. The elements of $\overrightarrow{Q}$ are included as ciphertext components. Finally, the ciphertext is published by the cloud server as:

$$CT = (C = Me(g,g)^{\alpha s}, C_0 = g^{s}, \left\{\grave{C}_\iota = g^{a\hat{\lambda}_i} h_{\rho(i)}^{-r_i}, \grave{D}_\iota = g^{r_i}, T_i = g^{-aF_i}\right\}_{i=1}^{l})$$

Since the ciphertext components associated with attributes are partially encrypted by the cloud server, the outsourcing of computation offered by the cloud is efficiently exploited.

$\boldsymbol{KeyGen}((m \text{ out of } n)\boldsymbol{MSK_i}, \boldsymbol{S}) \rightarrow (\boldsymbol{SK})$**:** This algorithm is run by m AAs which are selected randomly by a user u_k to contact separately to obtain their secret key shares SK_i that then allow u_k to compute its secret key $\boldsymbol{SK}$. Each secret key share SK_i is generated by AA_i $(i = 1,2,\dots,m)$ based on u_k's attribute set $\boldsymbol{S}$, and it is securely sent to u_k by AA_i. SK_i is defined as:

$$SK_i = (K_i = g^{\,MSK_i} g^{\,at_i P(0)} , L_i = g^{at_i},\ \acute{L}_{1_\imath} = g^{t_i P(u_k)},\ \forall x \in S \quad K_{x_i} = h_x^{t_i P(u_k)})$$

Here, the exponent $t_i \in Z_p$ is randomly chosen by each AA_i, and the exponent $P(u_k)$ represents a share calculated by AA_i $(i = 1,2,\dots,m\,)$ for user u_k to obtain secret $P(0)$ that needs c + 1 shares to recover.

Once u_k collects the m secret key shares SK_i, it computes its secret key SK as follows:

$$\boldsymbol{K} = \prod_{i=1}^{m} K_i^{\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}}}$$

$$\boldsymbol{K} = \prod_{i=1}^{m}(g^{\,MSK_i} g^{\,at_i P(0)})^{\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}}}$$

$$\boldsymbol{K} = g^{\sum_{i=1}^{m}(MSk_i.\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}})} . g^{\sum_{i=1}^{m}(aP(0)t_i.\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}})}$$

$$\boldsymbol{K} = g^{\alpha}. g^{aP(0)\sum_{i=1}^{m}(t_i.\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}})}$$

$$\boldsymbol{L} = \prod_{i=1}^{m} L_i^{\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}}}$$

$$\boldsymbol{L} = g^{\sum_{i=1}^{m}(at_i.\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}})}$$

$$\boldsymbol{L} = g^{a\sum_{i=1}^{m}(t_i.\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}})}$$

$$\acute{\boldsymbol{L}}_{\mathbf{1}} = \prod_{i=1}^{m} \grave{L}_{1_i}^{\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}}}$$

$$\acute{\boldsymbol{L}}_{\mathbf{1}} = g^{\sum_{i=1}^{m}(P(u_k)t_i.\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}})}$$

$$\acute{\boldsymbol{L}}_{\mathbf{1}} = g^{P(u_k)\sum_{i=1}^{m}(t_i.\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}})}$$

For all $x \in S$:

$$\boldsymbol{K_x} = \prod_{i=1}^{m} K_{x_i}^{\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}}}$$

$$\boldsymbol{K_x} = h_x^{\sum_{i=1}^{m}(P(u_k)t_i.\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}})}$$

$$\boldsymbol{K_x} = h_x^{P(u_k)\sum_{i=1}^{m}(t_i.\prod_{j=1,j\neq i}^{m}\frac{a_{id_j}}{a_{id_j}-a_{id_i}})}$$

For simplicity, let t be equal to:

$$\boldsymbol{t} = \sum_{i=1}^{m}(t_i. \prod_{j=1,j\neq i}^{m} \frac{a_{id_j}}{a_{id_j} - a_{id_i}})$$

As a result, u_k's secret key is defined as:

$$SK_{u_k} = (K = g^{\alpha} g^{atP(0)}, L = g^{at}, \acute{L}_1 = g^{tP(u_k)}, \forall x \in S \quad K_x = h_x^{tP(u_k)})$$

To ensure the collusion resistance property, upon each user revocation event, CAA generates a proxy key that contains a set of pairs of c secret shares (i.e. points of the secret polynomial) and the corresponding identities of the revoked users u_i ($i \in \{1,2, \dots, c\}$). These shares are embedded by the proxy server in a piece of the user's secret key to transform it for decryption. Therefore, the proxy key will be as follows:

$Proxy_{\ Key} = \{< u_i, P(u_i) >\}_{u_i \in R}$, where R is the set of revoked users.

$\boldsymbol{BlindSK}$ ($\boldsymbol{SK}$) → $\boldsymbol{BSK}$: This algorithm is executed by data user u_k to blind its key before sending it to the cloud server to securely decrypt the ciphertext. The input of this algorithm is u_k's secret key *SK*. In this algorithm, u_k selects a random exponent $Y \in Z_p$ to compute the blinded secret key:

$$BK = (K)^{1/Y} = g^{\alpha/Y} g^{atP(0)/Y},$$

$$BL = (L)^{1/Y} = g^{at/Y},$$

$$B\acute{L}_1 = (\acute{L}_1)^{1/Y} = g^{tP(u_k)/Y},$$

$$\forall x \in S \quad BK_x = (K_x)^{1/Y} = h_x^{tP(u_k)/Y}$$

Finally, the blinded secret key is set as $\boldsymbol{BSK} = (BK, BL, B\acute{L}_1, \{ BK_x\}_{x\in S})$.

$\boldsymbol{KeyTransform}(\boldsymbol{BL}, \boldsymbol{Proxy}_{\ \boldsymbol{Key}}) \rightarrow \boldsymbol{BL}^{\boldsymbol{Proxy_{key}}}$: The proxy server takes the component BL of user u_k's blinded secret key and then transforms it by embedding the shares of the revoked users in that piece using its $Proxy_{\ Key}$. The use of the revoked shares will prevent their associated users from using these shares to recover the related plaintext. In addition, the proxy uses its information and the identity of non-revoked user u_k to compute:

$$\forall i, j \in \{1, \dots, c\}, k \notin \{1,\dots,c\}, \qquad \lambda_i = \frac{u_k}{u_k - u_i} . \prod_{j \neq i} \frac{u_j}{u_j - u_i} \qquad (5.1)$$

For every non-revoked user u_k, the proxy calculates λ_k using the equation (5.1) and then sends it to u_k. In this case, the collusion attack happens only when the revoked users collude with the cloud and proxy servers together.

$$\forall u_k \notin R, (BL_{u_k})^{Proxy_{key}} = \left(BL_{u_k} \right)^{\sum_{i=1}^{c} \lambda_i P(u_i)} = \left(g^{at/_{Y}} \right)^{\sum_{i=1}^{c} \lambda_i P(u_i)} = g^{\frac{at \sum_{i=1}^{c} \lambda_i P(u_i)}{Y}}$$

$\boldsymbol{Decrypt} \, (\boldsymbol{CT}, \boldsymbol{BSK}, \boldsymbol{BL}^{\boldsymbol{Proxy_{key}}}) \rightarrow \boldsymbol{M}$: The decryption algorithm inputs are a ciphertext CT, the blinded user's private key BSK and the output $BL^{Proxy_{key}}$ of the $KeyTransform$ algorithm. Once the user's attributes satisfy the access structure in the ciphertext and it is not on the revocation list, it can recover the message M as detailed below. The values $\{\omega_i \in Z_p\}_{i \in I}$ are supposed to be a set of constants, where $I = \{i : \rho(i) \in S\}$ and $I \subset \{1,2,..,l\}$ such that when $\{\bar{\lambda}_i\}_{i \in I}$ are valid shares of a secret s which correspond to W_i, then $\sum_{i \in I} \omega_i \bar{\lambda}_i = s$. Part of this algorithm, which includes heavy operations, is run by the cloud server. The decryption algorithm computes M as follows:

$$B_1 = \frac{e(C_0 \, , BK)}{\left(\prod_{i \in I} \left(e\left(\grave{C}_\iota, B\acute{L}_1{}^{\lambda_k} \right) e\left(\grave{D}_\iota, BK_{\rho(i)}{}^{\lambda_k} \right) e\left(\, T_i, B\acute{L}_1{}^{\lambda_k} \right) \right)^{\omega_i} \right). e(C_0, BL^{Proxy_{key}})}$$

$$= \frac{e\left(\, g^s \, , g^{\,\alpha/_{Y}} g^{\,atP(0)/_{Y}} \right)}{\left(\prod_{i \in I} \left(e\left(g^{a\widehat{\lambda_\iota}} . h_{\rho(i)}^{-r_i}, g^{\,tP(u_k)\lambda_k/_{Y}} \right) e\left(g^{\,r_i}, \;\; h_{\rho(i)}{}^{\,tP(u_k)\lambda_k/_{Y}} \right) e(g^{-aF_i}, g^{\,tP(u_k)\lambda_k/_{Y}}) \right)^{\omega_i} \right). e(g^s, g^{\,at \sum_{i=1}^{c} \lambda_i P(u_i)/_{Y}})}$$

$$= \frac{e(g,g)^{\alpha\, s/_{Y}} e(g,g)^{\,a\, stP(0)/_{Y}}}{\left(\prod_{i \in I} \left(e(g,g)^{a\widehat{\lambda}_\iota tP(u_k)\lambda_k/_{Y}} e(g, h_{\rho(i)})^{-r_i tP(u_k)\lambda_k/_{Y}} e(g, \; h_{\rho(i)})^{r_i tP(u_k)\lambda_k/_{Y}} e(g,g)^{-aF_i tP(u_k)\lambda_k/_{Y}} \right)^{\omega_i} \right). e(g,g)^{ast \sum_{i=1}^{c} \lambda_i P(u_i)/_{Y}}}$$

$$= \frac{e(g,g)^{\alpha\, s/_{Y}} e(g,g)^{\,a\, stP(0)/_{Y}}}{\left(\prod_{i \in I} \left(e(g,g)^{a\widehat{\lambda_\iota} tP(u_k)\lambda_k/_{Y}} e(g,g)^{-aF_i tP(u_k)\lambda_k/_{Y}} \right)^{\omega_i} \right). e(g,g)^{ast \sum_{i=1}^{c} \lambda_i P(u_i)/_{Y}}}$$

$$= \frac{e(g,g)^{\alpha\, s/_{Y}} e(g,g)^{\,a\, stP(0)/_{Y}}}{\left(\prod_{i \in I} \left(e(g,g)^{ta(\widehat{\lambda_\iota} - F_i)P(u_k)\lambda_k/_{Y}} \right)^{\omega_i} \right). e(g,g)^{ast \sum_{i=1}^{c} \lambda_i P(u_i)/_{Y}}}$$

$$= \frac{e(g,g)^{{\alpha\, s}/{Y}} e(g,g)^{{a\, stP(0)}/{Y}}}{e(g,g)^{{taP(u_k)\lambda_k \sum_{i\in \grave{I}}(\widehat{\lambda_\imath}-F_i)\omega_i}/{Y}} . e(g,g)^{{ast\sum_{i=1}^{c}\lambda_i P(u_i)}/{Y}}}$$

Where $\{\bar{\lambda}_i = (\widehat{\lambda}_i - F_i) \text{ mod p}\}_{i\in l}$

$$B_1 = \frac{e(g,g)^{{\alpha\, s}/{Y}} e(g,g)^{{a\, stP(0)}/{Y}}}{e(g,g)^{{tasP(u_k)\lambda_k}/{Y}} . e(g,g)^{{ast\sum_{i=1}^{c}\lambda_i P(u_i)}/{Y}}}$$

$$B_1 = \frac{e(g,g)^{{\alpha\, s}/{Y}} e(g,g)^{{a\, stP(0)}/{Y}}}{e(g,g)^{{ast(\sum_{i=1}^{c}\lambda_i P(u_i)+P(u_k)\lambda_k)}/{Y}}}$$

$$B_1 = \frac{e(g,g)^{{\alpha\, s}/{Y}} e(g,g)^{{a\, stP(0)}/{Y}}}{e(g,g)^{{astP(0)}/{Y}}}$$

$$B_1 = e(g,g)^{{\alpha\, s}/{Y}}$$

Then the value of B_1 is sent by the cloud to the user who uses the value of exponent Y to compute $UB_1 = (B_1)^Y = (e(g,g)^{{\alpha\, s}/{Y}})^Y = e(g,g)^{\alpha\, s}$. The decryption algorithm can calculate the message M as:

$$M = \frac{C}{UB_1} = \frac{Me(g,g)^{\alpha s}}{e(g,g)^{\alpha s}}$$

$\boldsymbol{PolicyUpdate}\big(\boldsymbol{PK}, \boldsymbol{KOwner}, \boldsymbol{A}, \boldsymbol{\acute{A}}\big) \rightarrow (\boldsymbol{ES}, \overrightarrow{\boldsymbol{\grave{Q}}})$ and **Re-Encryption**$(\boldsymbol{PK}, \boldsymbol{ES}, \overrightarrow{\boldsymbol{\grave{Q}}}, \boldsymbol{\acute{A}}) \rightarrow$ $\boldsymbol{\acute{C}T}$: These two algorithms are similar to the corresponding ones in **Section 4.2.4**.

5.3 Scheme Analysis

In this section, the analysis of our proposed scheme is presented in terms of security and performance.

5.3.1 Security Analysis

In our distributed environment, and based on the mentioned security assumptions and requirements, we analyse the security of the proposed scheme in terms of *data confidentiality*, *collusion resistance*, *fine-grained access control* and *forward security*.

Data Confidentiality. We use the following theorem which can be proved in the same way as the one in Waters' system [42] (see **Section 3.7.1)**. The reason for this is that our scheme uses Waters' system as a basic scheme. In addition, in our proposed multi-authority access control scheme, we need to prove that sharing the master secret key in the setup phase, the processes of policy update and user revocation, and the operations of proxy and outsourced decryption are secure.

Theorem 5.1. Once the decisional q-Parallel BDHE assumption holds, all polynomial time adversaries have negligible time to selectively break our proposed CP-ABE scheme, where the challenge LSSS matrix is $W^*(l^* \times n^*)$ with $l^*, n^* \leq q$.

Proof. Let $\mathcal{A}$ be an adversary with a non-negligible advantage against our proposed scheme in the selective security game. This adversary selects W^* as a challenge matrix where each of l^* (the number of its rows) and n^* (the number of its columns) is less than or equal to q. In our proposed scheme, in the setup phase, the security vulnerability is reduced as a consequence of using SSSS by all the attribute authorities in the scheme to implicitly reconstruct a part of the master secret key (i.e. α). Therefore, no entity in the scheme knows the value of α. That means the master secret key is totally secure unless a threshold number of attribute authorities collude with each other to reconstruct its value, which can be prevented by choosing an appropriate threshold number. On the case when the CAA attacked, the other scheme entities keep offering services because part of the master secret key is still hidden as well as the CAA does not involve in generating a secret key to each user. However, the scheme cannot revoke users or registering new users until the scheme manages the issue and CAA returns to work.

In this game, any secret keys can be queried by $\mathcal{A}$ unless those keys are able to decrypt the simulator $\mathcal{B}$'s ciphertext or the identity u_k of adversary $\mathcal{A}$ does not belong to the revocation list R^*, where $\mathcal{B}$ has all the secret details about AAs' secret shares and the secret master key and hides them from $\mathcal{A}$. In this case, we can complete the proof in the same way as the single authority system in [42] and our single-authority scheme in **Section 3.7.1** and **Section 4.3.1**. Therefore, if this adversary wins the selective security game, then $\mathcal{B}$ can break the security of Waters' system [42].

Collusion Resistance. Each user in our proposed scheme is assigned with a unique identity u_k, which is implicitly used to invalidate the main part of the user's secret key when that user is revoked. The approach used for this purpose is SSSS. In addition, the cloud server does not take part in the processes of sharing the master key among *AA*s and the secret key generation. Based on the previous considerations, the revoked users will be prevented from colluding with the cloud server to gain unauthorized information. On the other hand, using CP-ABE meets the needs of preventing the authorised, malicious users from combining their attributes together to access unauthorised information at a higher security level.

Fine Grained Access Control as well as Forward Security. These requirements are also achieved as shown in **Section 4.3.1**.

5.3.2 Performance Analysis

We have chosen the multi-authority scheme in [85] to compare with our proposed scheme. Most of the existing systems utilize an access tree or AND gate as an access structure, the authorities in such systems manage disjoint attribute sets, or these systems assume that the cloud is trusted, which are different from the security assumptions and access structure used to develop our scheme. Only the scheme in [85] is the most appropriate one for the comparison, as it is based on the same security assumptions and supports the LSSS access structure as well. The comparison will be in terms of computation and communication overheads. **Table 5.2** illustrates the capability of our scheme against the system in [85].

Communication overhead. The required communications in our scheme, which are the same as those in [85], are threefold. First, communications between a user and a threshold number of AAs are needed for the user to gain its possessed secret key parts for the secret key computation. Secondly, communications among all the AAs in the system are required to share the master key in the $SetupAA$ phase. Thirdly, the communication between the AAs and central authority CAA is necessary for the attribute authority to gain its identity. In addition, our scheme needs extra communications among authorized users and the proxy server upon each user revocation event, to update their keys. It also requires communications between a data owner and the cloud server from one side, and between a user and the cloud server from the other side, to outsource the encryption and decryption operations, respectively.

Computation overhead. In our scheme and the system in [85], although a user needs to do a lot of computations to compute the secret key from the shares collected from a threshold number of AAs, these operations are run only once, and after that the user can store and reuse the secret key. Furthermore, although our proposed scheme addresses more issues (as illustrated in **Table 5.2**) such as policy update and user revocation than the compared system, our scheme in general incurs no more computation to data owners and users than the compared one, which will be supported by our experiment in the next section. This is because our scheme considers outsourcing the heavy computational operations (such as partial encryption and decryption operations) to the cloud server. That makes our proposed scheme more practical, particularly to mobile users.

Table 5.2: Comparative summary of the capability of our scheme against related work

scheme	Policy Update	User Revocation	Outsourcing decryption	Outsourcing encryption
Li et al. [85]	×	×	×	×
Our scheme	√	√	√	√

5.4 The Experimental Results and Evaluation

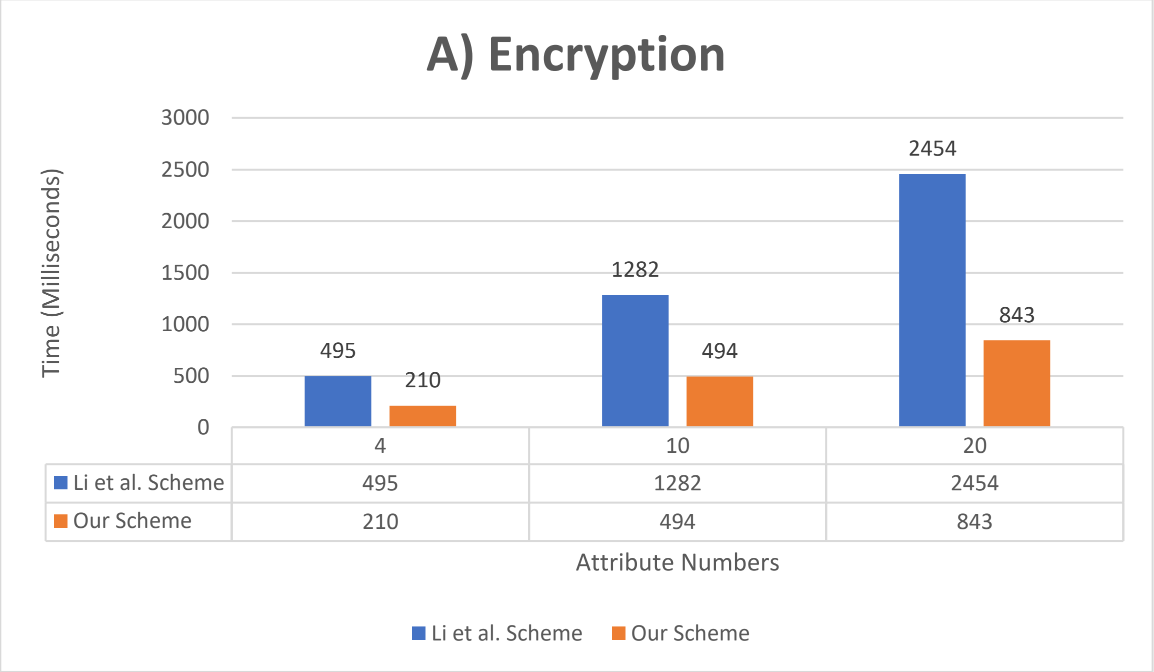
We have implemented our proposed scheme. The CP-ABE system in [42] was taken as the base and adapted to our scheme. The experiment supports our dynamic policy update and user revocation processes and relieves the encryption and decryption burden on data owners and users, respectively.

The implementation uses the Java Pairing Based Cryptography (JPBC) library [115]. The elliptic curve group of type (A) curves of JPBC is used with a 512-bit base field. The experiment is executed on Intel(R) Core (TM) i7- 2006, 3.40 GH CPU, where pairing takes about 31 milliseconds. The scheme is implemented with five attribute authorities and the threshold number is three.

Similar to our proposed single authority scheme in **Chapter 4**, the execution time for a data owner to carry out part of an encryption operation will increase linearly with the number of attributes in the access structure. Where, we have tested different access policies with 4,10 and 20 attributes, respectively, for comparing the performance of our proposed multi-authority scheme with the performance of Li et al. [85]. The results indicate that our scheme is more efficient as demonstrated in **Figure 5.5 A**. The efficiency of our scheme comes from outsourcing part of the encryption operation related to creating the components associated with attributes, to the cloud server. Consequently, the data owner just needs to calculate the vector of shares, encrypt them, and then send the encrypted vector to the cloud. However, the data owner in the compared system has to compute the shares and all the ciphertext components including the common ones and those associated with attributes.

Notably, **Figure 5.5 A** and **Figure 4.9 A** are the same. The reason for this is that the compared systems Li et al. [85] and PU-CP-ABE [65] are similar to our scheme in terms of using the CP-ABE system in [42] as the base. Whereas Li et al.'s scheme [85] develops the setup algorithm to employ multiple authorities instead of the single authority in [42], the PU-CP-ABE system [65] enhances the CP-ABE system in [42] to support policy update. Therefore, the encryption and the decryption operations are almost the same. Consequently, the improvement percentage of our proposed encryption operation is the same as in **Section 4.4**.

In terms of policy update, as mentioned earlier, the scheme designed by Li et al. [85] does not consider the process of policy update. Despite that, we have measured the execution time of our scheme to be aware of the time needed by the data owner to run this algorithm, as illustrated in **Figure 5.5 B**. After the first implementation with 4 attributes, the data owner runs the policy update algorithm to update its access policy by increasing the number of attributes to 10 and 20, respectively. The dashed line indicates that the execution time increases linearly with the number of attributes in the access policy.



	4	10	20
Li et al. Scheme	495	1282	2454
Our Scheme	210	494	843

Figure 5.5 A: The experimental results of the Encryption algorithm of our proposed multi-authority system compared with Li et al.'s scheme [85].

To build a feasible scheme that suits many limited-resources IoT devices, we alleviate the burden on users by securely outsourcing the heavy computational part of the decryption process to the cloud server and assign light, constant operations which have not been affected linearly with the complexity of the access policy, to the data users. Based on the theoretical model, our expectation indicates high performance findings.

The above expectation is also supported by our experiment showing that the execution time of the decryption operation of our proposed scheme is more competitive compared with the scheme of Li et al. [85] where their time needed for the decryption operation increases with the complexity of the access policy and the number of attributes involved in the decryption key. That is obvious in **Figure 5.5 C**, where Li et al.'s system [85] takes 139% more time than our technique when 4 attributes are used and this percentage increases to 254% and 264% when the utilised attributes are 10 and 20, respectively.

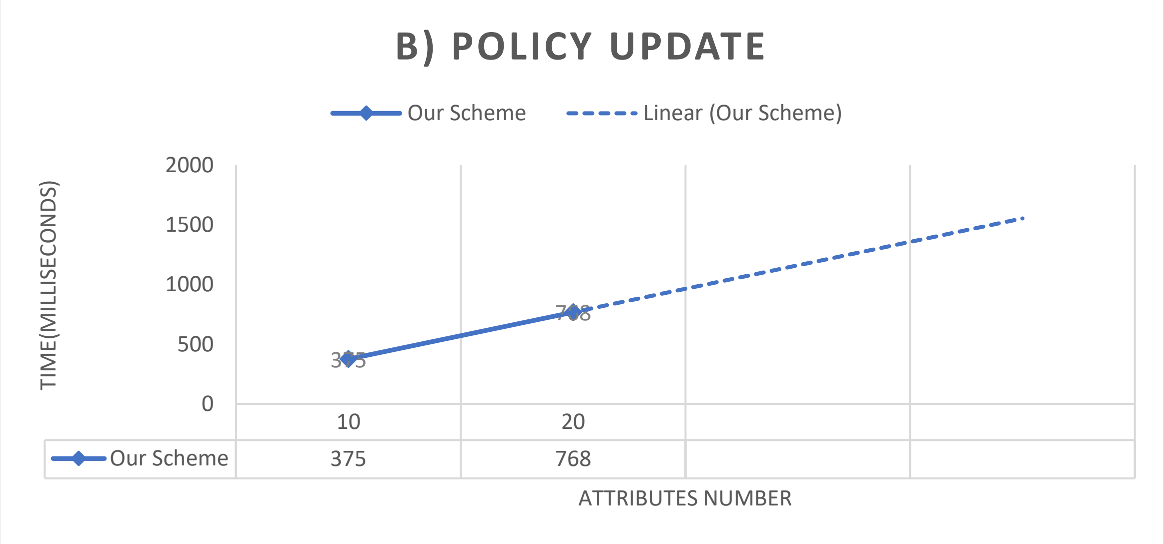
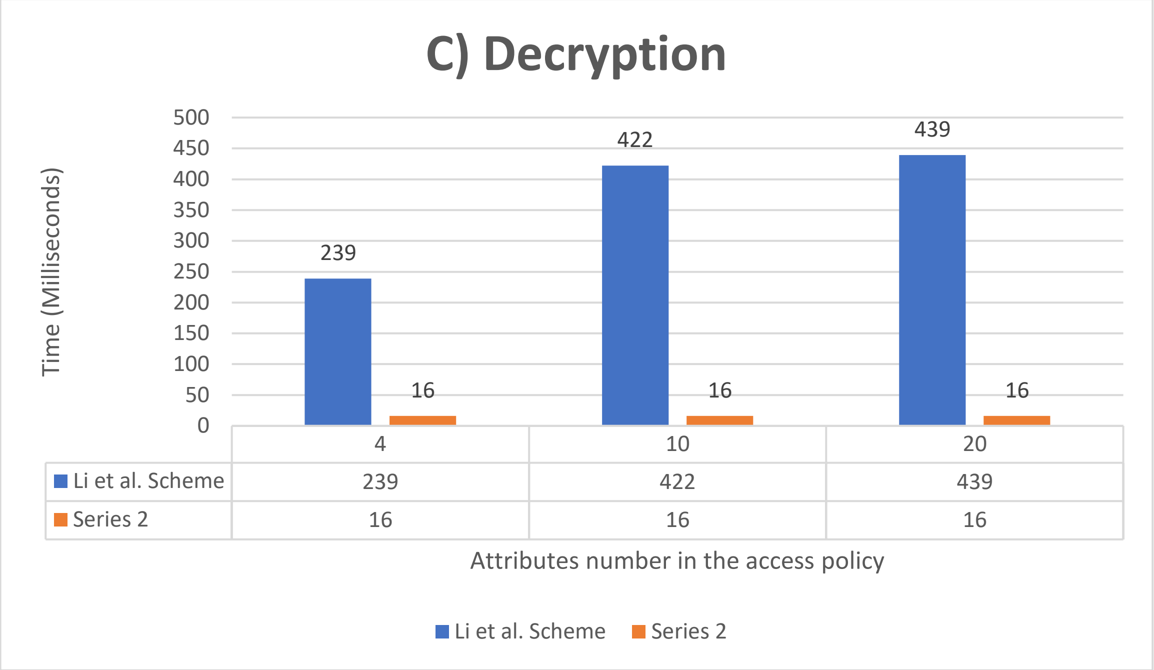


Figure 5.5 B: The experimental results of the Policy Update algorithm of our proposed multi-authority scheme.

In terms of the blind secret key algorithm, in our scheme, the user only carries out some constant, exponential operations that are considerably lighter than pairing operations. These operations include blinding its secret key and then sending it to the cloud server to compute internal results. These results are used by the data user to un-blind them to recover the plaintext which is the symmetric key of the encrypted data file.

The overall results state that our proposed multi-authority scheme has higher efficiency, functionality and practicality against the compared system.



	4	10	20
🟦 Li et al. Scheme	239	422	439
🟧 Series 2	16	16	16

Figure 5.5 C: The experimental results of the Decryption algorithm of our proposed multi-authority scheme compared with Li et al.’s scheme [85].

5.5 Conclusion

In this chapter, a new multi-authority CP-ABE scheme is proposed to enhance the system security and performance and resolve the problem of a single point bottleneck. Unlike other relevant existing systems, we have considered many current critical issues. These include a novel policy update process, user revocation and securely outsourcing expensive computational operations to the cloud without revealing any unauthorized information while preventing collusion attacks. The proposed scheme can provide fine-grained access control for public cloud storage. Our literature review, scheme analysis and the experimental results show that our scheme is superior over the compared existing multi-authority system and traditional single authority CP-ABE. For future work, we intend to minimise the communication overhead and develop the scheme to support dynamic attributes.

Chapter 6

Conclusions and Future Work

Chapter 6: Conclusions and Future Work

6.1 Conclusions

In this thesis, some fundamental issues of storing and sharing sensitive data in an untrusted environment have been considered by offering fine-grained access control services while protecting the security of such data. In our proposed work, although the system cryptographically enforces the data owners' access policies, it is able to update these policies dynamically without incurring a high computational cost.

Therefore, the major findings of this thesis can be concluded as follows:

1- Hosting data on the cloud can be vulnerable to loss, breach, or leakage.

2- Using CP-ABE can provide secure storing large amounts of sensitive data on cloud storage with one-to-many access control and embedding access policies. However, it needs some refinements.

3- Any system that does not address the revocation problem, needs to rebuild from the beginning to address this issue otherwise, it will not suit storing data on a dynamic environment such as cloud computing.

4- Testing and evaluating our proposed schemes showed that the encryption operation scaled linearly with the number of attributes, while in our single authority scheme there is just moderate drop in the performance of decryption operation. However, this drop is managed in our multi-authority scheme.

In our project, two access control schemes have been constructed, where most of the expensive operations are outsourced to the cloud server to alleviate the burden on the data owners and the scheme users, while protecting the confidentiality of their data. In addition, the cloud server is prevented from obtaining any unauthorised information.

In this chapter, the research contributions of the thesis and some future orientation are summarised below:

Single-Authority Access Control Scheme. The main contributions of our scheme that rectifies the weaknesses of the existing work are three-fold:

- Addressing the crucial challenge of dynamically updating policies that are already embedded in the ciphertext stored on a cloud server, while mitigating the computational cost. Where, this issue is considered complicated in most existing schemes due to the complexity of providing outsourced, cryptographic data management with a reasonable computational cost. Handling attributes efficiently occurs by setting the users' access privileges that are customised by policy updating and adapting them easily to new circumstances with low-cost communication and computation.

- Building a scheme that is resistant against any collusion attacks between the cloud service provider and revoked users. Most of the existing systems assume that the cloud server is semi trusted, so they assign some secret information to the server and maximise its control privileges. In contrast, our scheme enforces constraints to invalidate the secret keys known to revoked-users for data decryption and prevent such users and even the cloud server from having useful information. Hence, our scheme offers stronger security and easier implementation on the cloud server.
- Outsourcing core operations of encryption and policy update to the cloud server while protecting the confidentiality of data and without revealing any unauthorised information. This mitigates the computational burden on the data owners and thus rectifies the problems of considerable computation overheads incurred by existing systems.

Multi-Authority Access Control Scheme. In this scheme, we have extended the single attribute authority scheme in Chapter 4 to build a multi-authority scheme that enhances the scheme security by resolving the key escrow problem, eliminating the trust from one entity (i.e. attribute authority) and improving the scheme performance by addressing the single point of failure. This scheme leverages and extends some of the existing techniques to provide a number of desirable features described as follows:

- Dealing with the single point of security failure that all single authority systems and some multi-authority schemes suffer from. Furthermore, no one entity in our scheme has full control of all the information. This resolves the existing work's weaknesses by minimising the trust level of the authority and strengthening the privacy of user data.
- Extending the technique in [85] for the management of joint attribute sets to efficiently generate users' secret keys. The reason for this is that if each authority manages a different set of attributes, compromising or crashing an authority makes the unavailability of the whole system, which presents a performance bottleneck. Therefore, all attribute sets in our scheme are managed by each system attribute

authority individually. This extension overcomes most of the aforementioned existing systems' shortcomings (such as a system performance bottleneck). In addition, it also addresses the problem of the system in [85] in terms of its inability to deal with dynamic attribute changes.

- Adjusting the desirable properties of our proposed single-authority scheme in Chapter 4 to adapt to our multi-authority scheme. Such properties include efficient policy update and user revocation that enable the dynamicity and flexibility of customising and managing users' access privileges as well as protecting the scheme from collusion attacks. In addition, part of the encryption and policy update operations are securely moved to cloud servers. Although addressing the revocation issues is already a difficult mission in single-authority systems, it is considered as an even more significantly complicated task in decentralised multi-authority schemes. Since in such relevant existing schemes, the authorities are usually responsible for revoking users/attributes, the complexity of addressing the revocation problem is due to no connections among them in a decentralised-setting environment.
- Outsourcing additional expensive decryption operations to cloud servers, which alleviates the computational burden on scheme users. This is in contrast to the existing systems that incur high computational costs.

Consequently, our proposed revocable, decentralized access control scheme with multiple authorities efficiently deals with dynamic changes to access credentials and eliminates a single point of failure. We propose this scheme to allow securely storing data on a public cloud storage system and jointly administrating the system attribute set, where in a distributed setting, the most significant problem is revoking a key efficiently. The reason for this issue is that it is hard to inform all authorised, administrative entities when a key revocation event happens without management by a centralised point.

Due to the complexity of managing the revocation issues in any system with multiple authorities, to the best of our knowledge, our scheme is the first scheme that efficiently addresses the revocation issue. It presents two techniques that allow policy update and invalidate a user's secret key to deal with frequent changes to attributes and to prevent collusion attacks respectively, while outsourcing heavy computational operations to the cloud without revealing any useful information about the data. These techniques are managed by the data owner and the proxy server with few efforts. Although it is hard to compare our scheme with other systems due to the high capability of our scheme that differs from others, and the differences in security assumptions, the security and performance evaluations conducted in this project show that our scheme is secure and more efficient than the related work.

6.2 Future work

For further work, several directions can be followed to extend the research work in this thesis:

1- **Achieving Accountability.** In an untrusted environment, one of the potential research directions is to make our access control scheme accountable for protection against key exposure. Such exposure could occur when the secret decryption key of an authorized user is leaked. Since this key is valid, the decryption of the corresponding ciphertext is possible. Therefore, an effective mechanism is needed to protect the scheme from this threat.

2- **Reducing Communication Overhead.** In our proposed scheme, the main limitation is that many communications are needed. The resulting communications are due to outsourcing the complex, expensive computational operations to the cloud server to leverage its powerful, computational resources and to mitigate the burden on data owners and users. Investigating an approach to reduce these communications would be one of the key aspects to improve the efficiency of this scheme.

3- **Hiding Access Policies.** For sensitive policies, one of our substantial future research tasks is to explore techniques to enforce the access policies in a ciphertext form but hide such policies so that their private information is protected from disclosure during policy deployment.

4- **Using Dynamic Attributes.** Another core property that the CP-ABE technique needs to be extended with is dynamic attributes (e.g. location or time). It is an open challenge that would improve the dynamicity of our scheme by adding attributes to users' secret keys to restrict cloud data access in response to attribute changes and to enable a dynamic adaptation scheme.

5- **Implementing our Proposed Schemes with A Large Number of Attributes.**

References

[1] M. Bilal, "A review of internet of things architecture, technologies and analysis smartphone-based attacks against 3D printers," *arXiv preprint arXiv:1708.04560,* 2017.

[2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future generation computer systems,* vol. 29, no. 7, pp. 1645-1660, 2013.

[3] T. Bouabana-Tebibel and A. Kaci, "Parallel search over encrypted data under attribute based encryption on the Cloud Computing," *Computers & security,* vol. 54, no. October pp. 77-91, 2015.

[4] J. Geelan, "Twenty one experts define cloud computing. Virtualization, Electronic Magazine," Electronic Magazine, http://virtualization.sys-con. com/node/612375, 2008.

[5] L. Badger, T. Grance, R. Patt-Corner, and J. Voas, "Cloud computing synopsis and recommendations," *NIST special publication,* vol. 800, no. May p. 146, 2012.

[6] S. N. Samreen, N. Khatri-Valmik, S. M. Salve, and M. P. N. Khan, "Introduction to Cloud Computing," *International Research Journal of Engineering and Technology (IRJET),* vol. 5, no. 2, pp. 785-788, 2018.

[7] G. Kiryakova, N. Angelova, and L. Yordanova, "Application of cloud computing services in business," *Trakia Journal of Sciences,* vol. 1, pp. 392-396, 2015.

[8] S. Kamara and K. Lauter, "Cryptographic cloud storage," in *International Conference on Financial Cryptography and Data Security*, 2010: Springer, pp. 136-149.

[9] X. Li and X. Zhao, "Survey on access control model in cloud computing environment," in *In 2013 International Conference on Cloud Computing and Big Data*, (CloudCom-Asia), 2013: IEEE, pp. 340-345.

[10] A. R. Khan, "Access control in cloud computing environment," *ARPN Journal of Engineering and Applied Sciences,* vol. 7, no. 5, pp. 613-615, 2012.

[11] C.-W. Liu, W.-F. Hsien, C. C. Yang, and M.-S. Hwang, "A survey of attribute-based access control with user revocation in cloud data storage," *IJ Network Security,* vol. 18, no. 5, pp. 900-916, 2016.

[12] M. Ahmadi, M. Chizari, M. Eslami, M. J. Golkar, and M. Vali, "Access control and user authentication concerns in cloud computing environments," in *Telematics and Future Generation Networks (TAFGEN), 2015 1st International Conference on*, 2015: IEEE, pp. 39-43.

[13] S. Ruj, "Attribute based access control in clouds: A survey," in *Signal Processing and Communications (SPCOM), 2014 International Conference on*, 2014: IEEE, pp. 1-6.

[14] V. C. Hu *et al.*, "Guide to attribute based access control (ABAC) definition and considerations (draft)," *NIST special publication,* vol. 800, no. 162, 2013.

[15] M. Thangavel and P. Varalakshmi, "A survey on security over data outsourcing," in *2014 Sixth International Conference on Advanced Computing (ICoAC)*, 2014: IEEE, pp. 341-349.

[16] S. Ruj, M. Stojmenovic, and A. Nayak, "Decentralized access control with anonymous authentication of data stored in clouds," *IEEE transactions on parallel and distributed systems,* vol. 25, no. 2, pp. 384-394, 2014.

[17] K. Yang and X. Jia, "DAC-MACS: Effective data access control for multi-authority cloud storage systems," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 11, pp. 59-83, 2013.

[18] S. M. Khan and K. W. Hamlen, "AnonymousCloud: A data ownership privacy provider framework in cloud computing," in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, 2012: IEEE, pp. 170-176.

[19] A. A. Yassin, H. Jin, A. Ibrahim, W. Qiang, and D. Zou, "A practical privacy-preserving password authentication scheme for cloud computing," in *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International*, 2012: IEEE, pp. 1210-1217.

[20] K. Yang, Z. Liu, Z. Cao, X. Jia, D. S. Wong, and K. Ren, "TAAC: Temporal attribute-based access control for multi-authority cloud storage systems," *IACR Cryptology EPrint Archive,* vol. 2012, p. 651, 2012.

[21] Y. Zhu, H. Hu, G.-J. Ahn, D. Huang, and S. Wang, "Towards temporal access control in cloud computing," in *INFOCOM, 2012 Proceedings IEEE*, 2012: IEEE, pp. 2576-2580.

[22] D. Kulkarni and A. Tripathi, "Context-aware role-based access control in pervasive computing systems," in *Proceedings of the 13th ACM symposium on Access control models and technologies*, 2008: ACM, pp. 113-122.

[23] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," in *International Symposium on Handheld and Ubiquitous Computing*, 1999: Springer, pp. 304-307.

[24] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," *IEEE Communications Surveys & Tutorials,* vol. 16, no. 1, pp. 414-454, 2014.

[25] M. Giannikos, K. Kokoli, N. Fotiou, G. F. Marias, and G. C. Polyzos, "Towards secure and context-aware information lookup for the Internet of Things," in *Computing, Networking and Communications (ICNC), 2013 International Conference on*, 2013: IEEE, pp. 632-636.

[26] S. Mohammadi, K. Zamanifar, and S. M. Sharafi, "Management of context-aware software resources deployed in a cloud enviroment for improving quality of mobile cloud services," *International Journal of Distributed and Parallel Systems (IJDPS)* vol. 5, no. September, pp. 1-11, 2014.

[27] S.-K. Choi and J. Kwak, "Context-Aware Information-Based Access Restriction Scheme for Cloud Data," *International Journal of Multimedia & Ubiquitous Engineering,* vol. 8, no. 6, pp. 97-104, 2013.

[28] C. Gravier, J. Subercaze, A. Najjar, F. Laforest, X. Serpaggi, and O. Boissier, "Context awareness as a service for cloud resource optimization," *IEEE Internet Computing,* vol. 19, no. 1, pp. 28-34, 2015.

[29] Z. Zhou *et al.*, "Context-aware access control model for cloud computing," *International Journal of Grid and Distributed Computing,* vol. 6, no. 6, pp. 1-12, 2013.

[30] D. S. Kasunde and A. Manjrekar, "Verification of multi-owner shared data with collusion resistant user revocation in cloud," in *Computational Techniques in Information and Communication Technologies (ICCTICT), 2016 International Conference on*, 2016: IEEE, pp. 182-185.

[31] V. I. Mete and M. D. B. Gothawal, "Cipher text policy attribute based encryption for secure data retrieval in DTNs," *International Research Journal of Engineering and Technology (IRJET),* vol. 3, no. 6, pp. 1740-1745, 2016.

[32] S. Mathew, G. T. Vadakkumcheril, and T. J. Jose, "Decentralized firewall for attribute-based encryption with verifiable and revocable cloud access control."

[33] J. Hur, "Improving security and efficiency in attribute-based data sharing," *IEEE transactions on knowledge and data engineering,* vol. 25, no. 10, pp. 2271-2282, 2013.

[34] P. Zhang, Z. Chen, J. K. Liu, K. Liang, and H. Liu, "An efficient access control scheme with outsourcing capability and attribute update for fog computing," *Future Generation Computer Systems,* vol. 78, no. January, pp. 753-762, 2018.

[35] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: a survey," *Future Generation Computer Systems,* vol. 56, no. March, pp. 684-700, 2016.

[36] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," *Future Generation computer systems,* vol. 28, no. 3, pp. 583-592, 2012.

[37] A. Toninelli, R. Montanari, L. Kagal, and O. Lassila, "A semantic context-aware access control framework for secure collaborations in pervasive computing environments," in *International semantic web conference*, 2006: Springer, pp. 473-486.

[38] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*, 2006: Acm, pp. 89-98.

[39] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Security and Privacy, 2007. SP'07. IEEE Symposium on*, 2007: IEEE, pp. 321-334.

[40] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2005: Springer, pp. 457-473.

[41] M. Horváth, "Attribute-based encryption optimized for cloud computing," in *SOFSEM 2015: Theory and Practice of Computer Science*: Springer, 2015, pp. 566-577.

[42] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *International Workshop on Public Key Cryptography*, 2011: Springer, pp. 53-70.

[43] G. Wang, Q. Liu, and J. Wu, "Hierarchical attribute-based encryption for fine-grained access control in cloud storage services," in *Proceedings of the 17th ACM conference on Computer and communications security*, 2010: ACM, pp. 735-737.

[44] C.-C. Lee, P.-S. Chung, and M.-S. Hwang, "A Survey on attribute-based encryption schemes of access control in cloud environments," *IJ Network Security,* vol. 15, no. 4, pp. 231-240, 2013.

[45] Y. Li, J. Zhu, X. Wang, Y. Chai, and S. Shao, "Optimized ciphertext-policy attribute-based encryption with efficient revocation," *International Journal of Security & Its Applications,* vol. 7, No.6, no. 6, 2013.

[46] S. A. Wagh and B. Padmavathi, "Control cloud data access privilege with abe scheme and efficient user revocation," *International Journal of Engineering Science,* vol. 6, no. 5, pp.5941-5943, 2016. DOI 10.4010/2016.1441

[47] V. Kumar and P. V. Kumar, "A lterature survey on revocable multiauthority cipher text-policy attribute-based encryption (CP-ABE) scheme for cloud storage," *International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE),* vol. 3, no. 12, pp. 1723-1728, 2014.

[48] R. M. Abraham and P. Sriramya, "Efficient and secure attribute revocation of data in multi-authority cloud storage " *ARPN Journal of Engineering and Applied Sciences,* vol. 10, no. 13, pp. 5588-5592, 2015.

[49] K. Sridhar and V. Srinivas, "Revocable data access control for multi-authority cloud storage using cipher text-policy attribute based encryption," *International Journal of Research,* vol. 3, no. 10, pp. 427-435, 2016.

[50] K. Yang, 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*, 2013: ACM, pp. 523-528.

[51] C.-W. Liu, W.-F. Hsien, C.-C. Yang, and M.-S. Hwang, "A Survey of attribute-based access control with user revocation in cloud data storage," *International Journal of Network Security,* vol. 18, no. 5, pp. 900-916, 2016.

[52] Z. Wan, J. e. Liu, and R. H. Deng, "HASBE: a hierarchical attribute-based solution for flexible and scalable access control in cloud computing," *IEEE transactions on information forensics and security,* vol. 7, no. 2, pp. 743-754, 2012.

[53] L. Touati and Y. Challal, "Efficient CP-ABE attribute/key management for IoT applications," in I*EEE International Conference on Computer and Information Technology, Liverpool, United Kingdom, Oct ,* 2015: IEEE, pp. 343-350.

[54] K. Yang and X. Jia, "Expressive, efficient, and revocable data access control for multi-authority cloud storage," *IEEE transactions on parallel and distributed systems,* vol. 25, no. 7, pp. 1735-1744, 2014.

[55] J. Chen and H. Ma, "Efficient decentralized attribute-based access control for cloud storage with user revocation," in *Communications (ICC), 2014 IEEE International Conference on*, 2014: IEEE, pp. 3782-3787.

[56] S. Jahid, 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*, 2011: ACM, pp. 411-415.

[57] S. Jahid and N. Borisov, "Piratte: Proxy-based immediate revocation of attribute-based encryption," *arXiv preprint arXiv:1208.4877,* 2012.

[58] S. Kuragod, P. Nayak, and M. Kotari, "Implementation of IBE with outsourced revocation technique in cloud computing," *International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering,* vol. 4, no. 5, pp. 190-193, 2016.

[59] S. Govindappa and K. R. Shylaja, "An outsourced key revocation technique for secret key management in decentralized identity based encryption scheme," *International Journal of Advanced Research in Computer and Communication Engineering,* vol. 5, no. 4, pp. 787-791, 2016.

[60] Z. Xu and K. M. Martin, "Dynamic user revocation and key refreshing for attribute-based encryption in cloud storage," in *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, 2012: IEEE, pp. 844-849.

[61] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in *International Conference on Applied Cryptography and Network Security*, *Berlin, Heidelberg,* 2007: Springer, pp. 288-306.

[62] L. Zu, Z. Liu, and J. Li, "New ciphertext-policy attribute-based encryption with efficient revocation," in *Computer and Information Technology (CIT), 2014 IEEE International Conference on*, 2014: IEEE, pp. 281-287.

[63] Y. Cheng, Z.-y. Wang, J. Ma, J.-j. Wu, S.-z. Mei, and J.-c. Ren, "Efficient revocation in ciphertext-policy attribute-based encryption based cryptographic cloud storage," *Journal of Zhejiang University SCIENCE C,* vol. 14, no. 2, pp. 85-97, 2013.

[64] G. Wang and J. Wang, "Research on ciphertext-policy attribute-based encryption with attribute level user revocation in cloud storage," *Mathematical Problems in Engineering,* vol. 2017, 2017.

[65] W. Yuan, "Dynamic policy update for ciphertext-policy attribute-based encryption," *IACR Cryptology ePrint Archive,* vol. 2016, p. 457, 2016.

[66] V. H. Kalmani, D. Goyal, and S. Singla, "An efficient and secure solution for attribute revocation problem utilizing cp-abe scheme in mobile cloud computing," *nternational Journal of Computer Applications (0975 –8887),* vol. 129, no. November pp. 16-21, 2015.

[67] V. Odelu, A. K. Das, Y. S. Rao, S. Kumari, M. K. Khan, and K.-K. R. Choo, "Pairing-based CP-ABE with constant-size ciphertexts and secret keys for cloud environment," *Computer Standards & Interfaces,* vol. 54, no. September pp. 3-9, 2017.

[68] X. Fu, X. Nie, T. Wu, and F. Li, "Large universe attribute based access control with efficient decryption in cloud storage system," *Journal of Systems and Software,* vol. 135, no. January pp. 157-164, 2018.

[69] Y. Liu, Y. Zhang, J. Ling, and Z. Liu, "Secure and fine-grained access control on e-healthcare records in mobile cloud computing," *Future Generation Computer Systems,* vol. 78, no. January pp. 1020-1026, 2018.

[70] J. Li, Y. Zhang, X. Chen, and Y. Xiang, "Secure attribute-based data sharing for resource-limited users in cloud computing," *Computers & Security,* vol. 72, no. January pp. 1-12, 2018.

[71] R. Zhang, H. Ma, and Y. Lu, "Fine-grained access control system based on fully outsourced attribute-based encryption," *Journal of Systems and Software,* vol. 125, no. March pp. 344-353, 2017.

[72] A. Karati, R. Amin, and G. Biswas, "Provably secure threshold-based abe scheme without bilinear map," *Arabian Journal for Science and Engineering,* vol. 41, no. 8, pp. 3201-3213, 2016.

[73] H. Hong and Z. Sun, "High efficient key-insulated attribute based encryption scheme without bilinear pairing operations," *SpringerPlus,* vol. 5, no. 1, p. 131, 2016.

[74] C. Kumar, D. Kumar, and A. K. Reddy, "Concrete attribute-based encryption scheme with verifiable outsourced decryption," *International Journal of Engineering Trends and Technology (IJETT),* vol. 12, no. 9, pp.421-426, 2014.

[75] Y. Zhang, D. Zheng, and R. H. Deng, "Security and privacy in smart health: efficient policy-hiding attribute-based access control," *IEEE Internet of Things Journal,* vol. 5, no. 3, pp. 2130-2145, 2018.

[76] H. Cui, R. H. Deng, J. Lai, X. Yi, and S. Nepal, "An efficient and expressive ciphertext-policy attribute-based encryption scheme with partially hidden access structures, revisited," *Computer Networks,* vol. 133, no. March, pp. 157-165, 2018.

[77] Q. Han, Y. Zhang, and H. Li, "Efficient and robust attribute-based encryption supporting access policy hiding in Internet of Things," *Future Generation Computer Systems,* vol. 83, no. June, pp. 269-277, 2018.

[78] D. M. Freeman, "Converting pairing-based cryptosystems from composite-order groups to prime-order groups," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2010: Springer, pp. 44-61.

[79] K. Rajeshwari, "Multi-authority attribute based encryption in cloud computing for agriculture." *International Journal of Modern Trends in Engineering and Science,* vol.3, no. 8, pp.30-34, 2016.

[80] Y. Yang, X. Chen, H. Chen, and X. Du, "Improving privacy and security in decentralizing multi-authority attribute-based encryption in cloud computing," *IEEE Access,* vol. 6, no. March, pp. 18009-18021, 2018.

[81] J. Han, W. Susilo, Y. Mu, J. Zhou, and M. H. A. Au, "Improving privacy and security in decentralized ciphertext-policy attribute-based encryption," *IEEE transactions on information forensics and security,* vol. 10, no. 3, pp. 665-678, 2015.

[82] Z. Liu, Z. Cao, Q. Huang, D. S. Wong, and T. H. Yuen, "Fully secure multi-authority ciphertext-policy attribute-based encryption without random oracles," in *European Symposium on Research in Computer Security*, 2011: Springer, pp. 278-297.

[83] S. Ruj, A. Nayak, and I. Stojmenovic, "DACC: Distributed access control in clouds," in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on*, 2011: IEEE, pp. 91-98.

[84] M. Chase, "Multi-authority attribute based encryption," in *Theory of Cryptography Conference*, 2007: Springer, pp. 515-534.

[85] W. Li, K. Xue, Y. Xue, and J. Hong, "TMACS: A robust and verifiable threshold multi-authority access control system in public cloud storage," *IEEE Transactions on parallel and distributed systems,* vol. 27, no. 5, pp. 1484-1496, 2016.

[86] S. Kattimani and S. Pachouly, "A robust and verifiable threshold multi-authority access control system in public cloud storage," in *Computing Communication Control and automation (ICCUBEA), 2016 International Conference on*, 2016: IEEE, pp. 1-4.

[87] M. More and S. Y. Gaikwad, "A robust and verifiable threshold multi-authority access control system in public cloud storage," *International Journal of Advance Research, Ideas and Innovations in Technology, vol.3, pp.1220-1223,* 2017.

[88] K. Xue *et al.*, "RAAC: Robust and auditable access control with multiple attribute authorities for public cloud storage," *IEEE Transactions on Information Forensics and Security,* vol. 12, no. 4, pp. 953-967, 2017.

[89] F. Khan, H. Li, and L. Zhang, "Owner specified excessive access control for attribute based encryption," *IEEE Access,* vol. 4, no. November pp. 8967-8976, 2016.

[90] H. Zhong, W. Zhu, Y. Xu, and J. Cui, "Multi-authority attribute-based encryption access control scheme with policy hidden for cloud storage," *Soft Computing,* vol. 22, no. 1, pp. 243-251, 2018.

[91] J. Li, X. Chen, S. S. Chow, Q. Huang, D. S. Wong, and Z. Liu, "Multi-authority fine-grained access control with accountability and its application in cloud," *Journal of Network and Computer Applications,* vol. 112, no.June pp. 89-96, 2018.

[92] J. Ning, Z. Cao, X. Dong, K. Liang, L. Wei, and K.-K. R. Choo, "CryptCloud+: Secure and expressive data access control for cloud storage," *IEEE Transactions on Services Computing,* 2018, Accepted for publication, available at: https://ieeexplore.ieee.org/document/8252795/.

[93] D. V. Rudra Gowda M Patil, "Cloud++: A secure and timed data access control scheme for cloud," *International Journal of Innovative Research in Science, Engineering and Technology* vol. 7, no. 6, pp. 90-95, 2018.

[94] J. Ling and A. Weng, "A scheme of hidden-structure attribute-based encryption with multiple authorities," in *IOP Conference Series: Materials Science and Engineering*, 2018, vol. 359, no. 1: IOP Publishing, p. 012005.

[95] H. Lin, Z. Cao, X. Liang, and J. Shao, "Secure threshold multi authority attribute based encryption without a central authority," in *International Conference on Cryptology in India*, 2008: Springer, pp. 426-436.

[96] X. Li, S. Tang, L. Xu, H. Wang, and J. Chen, "Two-factor data access control with efficient revocation for multi-authority cloud storage systems," *IEEE Access,* vol. 5, no. September pp. 393-405, 2017.

[97] J. S. Milne, "Group theory," *Course Notes,* Available at http://www.jmilne.org/math/ CourseNotes/gt.html, version 3.13 ed., 2003.

[98] N. P. Smart, "The discrete logarithm problem on elliptic curves of trace one," *Journal of cryptology,* vol. 12, no. 3, pp. 193-196, 1999.

[99] D. Boneh, "The decision diffie-hellman problem," in *International Algorithmic Number Theory Symposium*, 1998: Springer, pp. 48-63.

[100] A. Menezes, "An introduction to pairing-based cryptography," *Recent trends in cryptography,* vol. 477, no. January pp. 47-65, 2009.

[101] R. Schoof, "Elliptic curves over finite fields and the computation of square roots mod " *Mathematics of computation,* vol. 44, no. 170, pp. 483-494, 1985.

[102] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in Cryptology—CRYPTO 2001*, 2001: Springer, pp. 213-229.

[103] A. Shamir, "How to share a secret," *Communications of the ACM,* vol. 22, no. 11, pp. 612-613, 1979.

[104] R. Ostrovsky, A. Sahai, and B. Waters, "Attribute-based encryption with non-monotonic access structures," in *Proceedings of the 14th ACM conference on Computer and communications security*, 2007: ACM, pp. 195-203.

[105] A. Beimel, "*Secure schemes for secret sharing and key distribution"*. PhD thesis, Technion-Israel Institute of technology, Faculty of computer science, Haifa, Israel, 1996.

[106] T. V. X. Phuong, G. Yang, and W. Susilo, "Hidden ciphertext policy attribute-based encryption under standard assumptions," *IEEE Transactions on Information Forensics and Security,* vol. 11, no. 1, pp. 35-45, 2016.

[107] T. Nishide, K. Yoneyama, and K. Ohta, "Attribute-based encryption with partially hidden encryptor-specified access structures," in *International Conference on Applied Cryptography and Network Security*, 2008: Springer, pp. 111-129.

[108] J. Li, C. Jia, J. Li, and X. Chen, "Outsourcing encryption of attribute-based encryption with mapreduce," in *International Conference on Information and Communications Security*, 2012: Springer, pp. 191-201.

[109] J. Lai, R. H. Deng, and Y. Li, "Expressive CP-ABE with partially hidden access structures," in *Proceedings of the 7th ACM symposium on information, computer and communications security*, 2012: ACM, pp. 18-19.

[110] H. Cui, R. H. Deng, G. Wu, and J. Lai, "An efficient and expressive ciphertext-policy attribute-based encryption scheme with partially hidden access structures," in *International Conference on Provable Security*, 2016: Springer, pp. 19-38.

[111] Z. Liu, Z. Cao, and D. S. Wong, "Efficient generation of linear secret sharing scheme matrices from threshold access trees," *Cryptology ePrint Archive: Listing,* 2010.

[112] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Public Key Cryptography*, 2011, vol. 6571: Springer, pp. 53-70.

[113] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Annual international conference on the theory and applications of cryptographic techniques*, 2011: Springer, pp. 568-588.

[114] C. Gentry and A. Silverberg, "Hierarchical ID-based cryptography," in *International Conference on the Theory and Application of Cryptology and Information Security*, 2002: Springer, pp. 548-566.

[115] (2013-12-04). *The Java Pairing Based Cryptography Library (JPBC)* [Online]. Available: http://gas.dia.unisa.it/projects/jpbc/#.WqKookx2v5o [Accessed 10/1/2018].

[116] S. Moffat, M. Hammoudeh, and R. Hegarty, "A survey on ciphertext-policy attribute-based encryption (cp-abe) approaches to data security on mobile devices and its application to iot," in *Proceedings of the International Conference on Future Networks and Distributed Systems*, 2017: ACM, p. 34.

[117] M. Ambrosin *et al.*, "On the feasibility of attribute-based encryption on Internet of Things devices," *IEEE Micro,* vol. 36, no. 6, pp. 25-35, 2016.

[118] A. De Caro and V. Iovino, "jPBC: Java pairing based cryptography," in *Computers and communications (ISCC), 2011 IEEE Symposium on*, 2011: IEEE, pp. 850-855.

[119] A. M. Braga and E. N. Nascimento, "Portability evaluation of cryptographic libraries on android smartphones," in *Cyberspace Safety and Security*: Springer, 2012, pp. 459-469.